

39th Computational Complexity Conference

CCC 2024, July 22–25, 2024, Ann Arbor, MI, USA

Edited by

Rahul Santhanam



Editors

Rahul Santhanam 

University of Oxford, UK

rahul.santhanam@cs.ox.ac.uk

ACM Classification 2012

Theory of computation

ISBN 978-3-95977-331-7

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/978-3-95977-331-7>.

Publication date

July, 2024

Bibliographic information published by the Deutsche Nationalbibliothek

The Deutsche Nationalbibliothek lists this publication 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.CCC.2024.0

ISBN 978-3-95977-331-7

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

- Luca Aceto (Reykjavik University, IS and Gran Sasso Science Institute, IT)
- Christel Baier (TU Dresden, DE)
- Roberto Di Cosmo (Inria and Université Paris Cité, FR)
- Faith Ellen (University of Toronto, CA)
- Javier Esparza (TU München, DE)
- Daniel Král' (Masaryk University, Brno, CZ)
- Meena Mahajan (*Chair*, Institute of Mathematical Sciences, Chennai, IN)
- Anca Muscholl (University of Bordeaux, FR)
- Chih-Hao Luke Ong (Nanyang Technological University, SG)
- Phillip Rogaway (University of California, Davis, US)
- Eva Rotenberg (Technical University of Denmark, Lyngby, DK)
- Raimund Seidel (Universität des Saarlandes, Saarbrücken, DE and Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Wadern, DE)
- Pierre Senellart (ENS, Université PSL, Paris, FR)

ISSN 1868-8969

<https://www.dagstuhl.de/lipics>

■ Contents

Preface	
<i>Rahul Santhanam</i>	0:ix
Conference Organization	
.....	0:xi
External Reviewers	
.....	0:xiii–0:xiv

Regular Papers

A Technique for Hardness Amplification Against AC^0	
<i>William M. Hoza</i>	1:1–1:20
Streaming Zero-Knowledge Proofs	
<i>Graham Cormode, Marcel Dall’Agnol, Tom Gur, and Chris Hickey</i>	2:1–2:66
Solving Unique Games over Globally Hypercontractive Graphs	
<i>Mitali Bafna and Dor Minzer</i>	3:1–3:15
Derandomizing Logspace with a Small Shared Hard Drive	
<i>Edward Pyne</i>	4:1–4:20
Explicit Time and Space Efficient Encoders Exist Only with Random Access	
<i>Joshua Cook and Dana Moshkovitz</i>	5:1–5:54
The Entangled Quantum Polynomial Hierarchy Collapses	
<i>Sabee Grewal and Justin Yirka</i>	6:1–6:23
Polynomial Pass Semi-Streaming Lower Bounds for K-Cores and Degeneracy	
<i>Sepehr Assadi, Prantar Ghosh, Bruno Loff, Parth Mittal, and Sagnik Mukhopadhyay</i>	7:1–7:16
Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries	
<i>Gil Cohen and Tal Yankovitz</i>	8:1–8:16
Lifting Dichotomies	
<i>Yaroslav Alekseev, Yuval Filmus, and Alexander Smal</i>	9:1–9:18
Explicit Directional Affine Extractors and Improved Hardness for Linear Branching Programs	
<i>Xin Li and Yan Zhong</i>	10:1–10:14
Linear-Size Boolean Circuits for Multiselection	
<i>Justin Holmgren and Ron Rothblum</i>	11:1–11:20
A Subquadratic Upper Bound on Sum-Of-Squares Composition Formulas	
<i>Pavel Hrubeš</i>	12:1–12:11
Hard Submatrices for Non-Negative Rank and Communication Complexity	
<i>Pavel Hrubeš</i>	13:1–13:12



Complexity of Robust Orbit Problems for Torus Actions and the <i>abc</i> -Conjecture <i>Peter Bürgisser, Mahmut Levent Doğan, Visu Makam, Michael Walter, and Avi Wigderson</i>	14:1–14:48
Quantum Automating TC^0 -Frege Is LWE-Hard <i>Noel Arteche, Gaia Carenini, and Matthew Gray</i>	15:1–15:25
A Strong Direct Sum Theorem for Distributional Query Complexity <i>Guy Blanc, Caleb Koch, Carmen Strassle, and Li-Yang Tan</i>	16:1–16:30
Local Enumeration and Majority Lower Bounds <i>Mohit Gurumukhani, Ramamohan Paturi, Pavel Pudlák, Michael Saks, and Navid Talebanfard</i>	17:1–17:25
Pseudorandomness, Symmetry, Smoothing: I <i>Harm Derksen, Peter Ivanov, Chin Ho Lee, and Emanuele Viola</i>	18:1–18:27
Information Dissemination via Broadcasts in the Presence of Adversarial Noise <i>Klim Efremenko, Gillat Kol, Dmitry Paramonov, Ran Raz, and Raghuvansh R. Saxena</i>	19:1–19:33
Lower Bounds for Set-Multilinear Branching Programs <i>Prerona Chatterjee, Deepanshu Kush, Shubhangi Saraf, and Amir Shpilka</i>	20:1–20:20
Public-Key Pseudoentanglement and the Hardness of Learning Ground State Entanglement Structure <i>Adam Bouland, Bill Fefferman, Soumik Ghosh, Tony Metger, Umesh Vazirani, Chenyi Zhang, and Zixin Zhou</i>	21:1–21:23
Depth- d Frege Systems Are Not Automatable Unless $P = NP$ <i>Theodoros Papamakarios</i>	22:1–22:17
Exponential Separation Between Powers of Regular and General Resolution over Parities <i>Sreejata Kishor Bhattacharya, Arkadev Chattopadhyay, and Pavel Dvořák</i>	23:1–23:32
Distribution-Free Proofs of Proximity <i>Hugo Aaronson, Tom Gur, Ninad Rajgopal, and Ron D. Rothblum</i>	24:1–24:18
On the Degree of Polynomials Computing Square Roots Mod p <i>Kiran S. Kedlaya and Swastik Kopparty</i>	25:1–25:14
Dimension Independent Disentangler from Unentanglement and Applications <i>Fernando Granha Jeronimo and Pei Wu</i>	26:1–26:28
Baby PIH: Parameterized Inapproximability of Min CSP <i>Venkatesan Guruswami, Xuandi Ren, and Sai Sandeep</i>	27:1–27:17
Finding Missing Items Requires Strong Forms of Randomness <i>Amit Chakrabarti and Manuel Stoeckl</i>	28:1–28:20
Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity <i>Shuichi Hirahara, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira</i>	29:1–29:56
The Computational Advantage of MIP^* Vanishes in the Presence of Noise <i>Yangjing Dong, Honghao Fu, Anand Natarajan, Minglong Qin, Haochen Xu, and Penghui Yao</i>	30:1–30:71

Low-Depth Algebraic Circuit Lower Bounds over Any Field <i>Michael A. Forbes</i>	31:1–31:16
BPL \subseteq L-AC ¹ <i>Kuan Cheng and Yichuan Wang</i>	32:1–32:14
Failure of Feasible Disjunction Property for k -DNF Resolution and NP-Hardness of Automating It <i>Michal Garlik</i>	33:1–33:23
Search-To-Decision Reductions for Kolmogorov Complexity <i>Noam Mazon and Rafael Pass</i>	34:1–34:20
Finer-Grained Hardness of Kernel Density Estimation <i>Josh Alman and Yunfeng Guan</i>	35:1–35:21
Gap MCSP Is Not (Levin) NP-Complete in Obfuscopia <i>Noam Mazon and Rafael Pass</i>	36:1–36:21

■ Preface

The papers for this volume were accepted for presentation at the 39th Computational Complexity Conference (CCC 2024), held from July 22-25, 2024, in Ann Arbor, USA. The conference is organised by the Computational Complexity Foundation (CCF) in cooperation with the ACM Special Interest Group on Algorithms and Computation Theory (SIGACT) and the European Association for Theoretical Computer Science (EATCS).

The call for papers sought original research papers in all areas of complexity theory. Of the 105 submissions, the program committee selected 36 for presentation at the conference.

The program committee would like to thank everyone involved in the conference, including all those who submitted papers for consideration as well as the reviewers (listed separately) for their scientific contributions; the board of trustees of the Computational Complexity Foundation for their advice and assistance; the Local Arrangements Committee chair Mahdi Cheraghchi; Adam Bouland, Nutan Limaye and Toniann Pitassi for their invited talks; and Michael Wagner for coordinating the production of the proceedings.

Rahul Santhanam

Program Committee Chair, on behalf of the Program Committee



■ Conference Organization

Program Committee

Scott Aaronson, University of Texas at Austin
Marshall Ball, New York University
Mark Bun, Boston University
Dean Doron, Ben-Gurion University
Ankit Garg, Microsoft Research India
Alexander Golovnev, Georgetown University
Troy Lee, University of Sydney
Robert Robere, McGill University
Noga Ron-Zewi, University of Haifa
Rahul Santhanam (Chair), University of Oxford
Srikanth Srinivasan, Copenhagen University and University of Aarhus and IIT Bombay
Madhu Sudan, Harvard University
Iddo Tzameret, Imperial College

Local Arrangements Committee

Mahdi Cheraghchi (Chair), University of Michigan

Board of Trustees

Amit Chakrabarti, Dartmouth College
Mahdi Cheraghchi, University of Michigan
Valentine Kabanets (President), Simon Fraser University
Nutan Limaye, IT University of Copenhagen
Meena Mahajan, The Institute of Mathematical Sciences
Pierre McKenzie, Université de Montréal
Susanna de Rezende, Lund University
Benjamin Rossman, Duke University
Shubhangi Saraf, University of Toronto



■ External Reviewers

Prashanth Amireddy
Gal Arnon
Noel Arteche
Srinivasan Arunachalam
Roozbeh Bassirian
Amik Raj Behera
Alexander Belovs
Shalev Ben-David
Omri Ben-Eliezer
Siddarth Bhaskar
Eric Blais
Andrej Bogdanov
John Bostanci
Marco Carmosino
Diptarka Chakraborty
Sourav Chakraborty
Prerona Chatterjee
Arkadev Chattopadhyay
Eshan Chattopadhyay
Yu Chen
James Cook
Peter Crawford-Kahrl
Samir Datta
Ronald de Wolf
Yotam Dikstein
Feyza Duman Keles
Arnaud Durand
Pranjal Dutta
Pavel Dvorak
Julian Dorfler
Christian Engels
Saroja Erabelli
Bill Fefferman
Noah Fleming
Karthik Gajulapalli
Michael Garlik
Dmitry Gavinsky
Alexandru Gheorghiu
Prantar Ghosh
Suprovat Ghoshal
Eli Goldin
Jesse Goodman
Mike Goos
Joshua Grochow
Stefan Grosser
Svyatoslav Gryaznov
Jiaxin Guan
Zeyu Guo
Tuomas Hakoniemi
Peter Hall
Kristoffer Arnsfelt Hansen
Prahlahd Harsha
Pooya Hatami
Edward A.Hirsch
Kaave Hosseini
William Hoza
Rahul Ilango
Dmitry Itsykson
Siddharth Iyer
Siddhartha Jain
Stacey Jeffery
Valentine Kabanets
Neeraj Kayal
Alexander Kelley
Samuel King
Alexander Knop
Tamara Kohler
Leszek Kolodziejczyk
William Kretschmer
Vaibhav Krishnan
Alexander Kulikov
Vinayak Kumar
Massimo Lauria
Victor Lecomte
Chin Ho Lee
Eunou Lee
Sihyun Lee
Jiatu Li
Jiawei Li
Zeyong Li
Nutan Limaye
Wei-Kai Lin
Qipeng Liu
Siqi Liu
Yanyi Liu
Yipan Liu
Bruno Loff
Xin Lyu
Pasin Manurangsi
Kunal Marwaha



Alex May
Gilbert Maystre
Noam Mazon
Ian Mertz
Ivan Mikhailin
Alexey Milovanov
Milan Mosse
Hmoon Mousavi
Saachi Mutreja
Satyajeet Nagargoje
Naoto Ohsaka
Rafael Oliveira
Shuo Pang
Aduri Pavan
Jan Pich
Vladimir Podolskii
Pavel Pudlak
Edward Pyne
Youing Qiao
Hanlin Ren
Artur Riazanov
Kilian Risse
Sushant Sachdeva
Shay Sapir
Sidhant Saraogi
Tselil Schramm
Adrian She
Suhail Sherif
Jamie Sikora
Amit Sinhababu
Alexander Smal
Anastasia Sofronova
Dmitry Sokolov
Carmen Strassle
Sathyawageeswar Subramanian
Navid Talebanfard
Raghunath Tewari
Bhargav Thankey
Neil Thapen
Thomas Thierauf
Santhoshini Velusamy
Marc Vinyals
Ben Lee Volk
Nadezhda Voronova
Erik Waingarten
Jordi Weggemans
Huacheng Yu
Henry Yuen

Wei Zhan
Jiapeng Zhang
Rachel Zhang

A Technique for Hardness Amplification Against AC^0

William M. Hoza   

Department of Computer Science, The University of Chicago, IL, USA

Abstract

We study hardness amplification in the context of two well-known “moderate” average-case hardness results for AC^0 circuits. First, we investigate the extent to which AC^0 circuits of depth d can approximate AC^0 circuits of some larger depth $d+k$. The case $k=1$ is resolved by Håstad, Rossman, Servedio, and Tan’s celebrated average-case depth hierarchy theorem (JACM 2017). Our contribution is a significantly stronger correlation bound when $k \geq 3$. Specifically, we show that there exists a linear-size AC_{d+k}^0 circuit $h: \{0,1\}^n \rightarrow \{0,1\}$ such that for every AC_d^0 circuit g , either g has size $\exp(n^{\Omega(1/d)})$, or else g agrees with h on at most a $(1/2 + \varepsilon)$ -fraction of inputs where $\varepsilon = \exp(-(1/d) \cdot \Omega(\log n)^{k-1})$. For comparison, Håstad, Rossman, Servedio, and Tan’s result has $\varepsilon = n^{-\Theta(1/d)}$. Second, we consider the majority function. It is well known that the majority function is moderately hard for AC^0 circuits (and stronger classes). Our contribution is a stronger correlation bound for the XOR of t copies of the n -bit majority function, denoted $MAJ_n^{\oplus t}$. We show that if g is an AC_d^0 circuit of size S , then g agrees with $MAJ_n^{\oplus t}$ on at most a $(1/2 + \varepsilon)$ -fraction of inputs, where $\varepsilon = (O(\log S)^{d-1}/\sqrt{n})^t$.

To prove these results, we develop a hardness amplification technique that is tailored to a specific type of circuit lower bound proof. In particular, one way to show that a function h is moderately hard for AC^0 circuits is to (a) design some distribution over random restrictions or random projections, (b) show that AC^0 circuits simplify to shallow decision trees under these restrictions/projections, and finally (c) show that after applying the restriction/projection, h is moderately hard for shallow decision trees with respect to an appropriate distribution. We show that (roughly speaking) if h can be proven to be moderately hard by a proof with that structure, then XORing multiple copies of h amplifies its hardness. Our analysis involves a new kind of XOR lemma for decision trees, which might be of independent interest.

2012 ACM Subject Classification Theory of computation \rightarrow Circuit complexity

Keywords and phrases Bounded-depth circuits, average-case lower bounds, hardness amplification, XOR lemmas

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.1

Related Version *Full Version*: <https://ecc.weizmann.ac.il/report/2023/176/> [41]

Funding Part of this work was done while the author was visiting the Simons Institute for the Theory of Computing.

Acknowledgements I thank Avishay Tal for collaboration at an early stage of this project. I thank Li-Yang Tan for encouragement. I thank Pooya Hatami for a helpful conversation.

1 Introduction

1.1 Average-Case Circuit Lower Bounds

Circuit lower bounds are at the heart of computational complexity theory. To understand the limitations of (extremely) efficient computation, we seek to prove that certain explicit functions cannot be computed by certain interesting classes of Boolean circuits. In fact, ideally, we want to prove *average-case* circuit lower bounds, also known as *correlation bounds*. That is, we would like to prove that circuits in some class \mathcal{C} cannot compute some function $h: \{0,1\}^n \rightarrow \{0,1\}$ on more than a $(1/2 + \varepsilon)$ -fraction of inputs for some small value $\varepsilon > 0$:

$$\text{For every } g \in \mathcal{C}, \quad \Pr_{\mathbf{x} \in \{0,1\}^n} [g(\mathbf{x}) = h(\mathbf{x})] \leq \frac{1}{2} + \varepsilon. \quad (1)$$

We would like ε to be as small as possible. For example, one motivation for trying to minimize ε comes from the Nisan-Wigderson framework for converting correlation bounds into pseudorandom generators (PRGs) [57]. In this framework, a bound of the form (1) implies a PRG with error εn , and in particular, the framework requires $\varepsilon < 1/n$.

In this work, we focus on the case that \mathcal{C} consists of AC^0 circuits, i.e., circuits made up of AND and OR gates of unbounded fan-in, with literals and constants at the bottom. The *size* of the circuit is the number of AND and OR gates, and the *depth* of the circuit is the length of the longest path from an input gate to the output gate. We refer to an AC^0 circuit of depth d as an “ AC_d^0 circuit.” We are especially interested in the constant-depth regime; this class of circuits can be viewed as a model of constant-time parallel computation. Some of the most celebrated theorems in circuit complexity are lower bounds on the size of AC^0 circuits computing various explicit functions. For example, if g is an AC_d^0 circuit, then g famously cannot compute the parity function on n bits or the majority function on n bits, unless g has size at least $\exp(c_d \cdot n^{1/(d-1)})$ [28, 1, 80, 35, 36].

1.2 Hardness Amplification and Yao’s XOR Lemma

One appealing approach for proving strong correlation bounds is to first construct a function h that is “moderately hard” (e.g., maybe we have $\varepsilon = 1/\sqrt{n}$), and then apply some kind of *hardness amplification* scheme that converts h into a “very hard” function (e.g., maybe now we can take $\varepsilon = n^{-\omega(1)}$). The most famous method for hardness amplification is Yao’s XOR Lemma [79, 53, 43, 29]. Starting from a hard function $h: \{0,1\}^n \rightarrow \{0,1\}$, this lemma considers the new hard function $h^{\oplus t}: \{0,1\}^{nt} \rightarrow \{0,1\}$ defined by $h^{\oplus t}(x^{(1)}, \dots, x^{(t)}) = \bigoplus_{i=1}^t h(x^{(i)})$. One well-known version¹ of Yao’s XOR Lemma says that if h is moderately hard for $\text{MAJ} \circ \mathcal{C}$ circuits, where MAJ denotes the majority function, then $h^{\oplus t}$ is very hard for \mathcal{C} circuits.

In the context of relatively weak classes such as AC^0 , the distinction between \mathcal{C} and $\text{MAJ} \circ \mathcal{C}$ is extremely important. Proving lower bounds on the size of $\text{MAJ} \circ \mathcal{C}$ circuits is generally much more difficult than proving lower bounds on the size of \mathcal{C} circuits. For this reason, there is a great deal of interest in “removing the majority gate” from Yao’s XOR Lemma. For example, we can ask the following.

► **Question 1.1** (Does XORing amplify hardness for AC^0 ?). *Let $h: \{0,1\}^n \rightarrow \{0,1\}$ and let $t = \log n$. Assume that every constant-depth subexponential-size AC^0 circuit g satisfies*

$$\Pr_{\mathbf{x} \in \{0,1\}^n} [g(\mathbf{x}) = h(\mathbf{x})] \leq \frac{1}{2} + n^{-\Omega(1)}.$$

Does it follow that every constant-depth polynomial-size AC^0 circuit g satisfies

$$\Pr_{\mathbf{x} \in \{0,1\}^{nt}} [g(\mathbf{x}) = h^{\oplus t}(\mathbf{x})] \leq \frac{1}{2} + n^{-\omega(1)}?$$

Several recent papers have developed and applied a refined version of Yao’s XOR Lemma featuring an “approximate linear sum” gate instead of the traditional majority gate [22, 21, 20, 42, 19, 25]. This clever approach has been fruitful, but it is still not applicable if we

¹ See, for example, Viola’s work [77].

start with a function that is hard merely for AC^0 circuits. Unfortunately, there are strong *barrier results* saying that every “black-box” hardness amplification scheme must involve *some* nontrivial computational overhead [74, 32, 63, 31, 62]. As a special case, this line of work implies that Theorem 1.1 cannot be resolved affirmatively via a “black-box” hardness amplification scheme. Thus, we have an ironic state of affairs: we have a rich toolkit for proving lower bounds on the size of AC^0 circuits, because we are able to exploit these circuits’ weaknesses, but at the same time, *specifically because these circuits are too weak*, we cannot use Yao’s XOR Lemma to amplify our lower bounds.²

1.3 Our Contributions

In this work, we develop a non-black-box method for hardness amplification, applicable to some (but not all) moderate hardness results for AC^0 circuits. We use our method to amplify two well-known average-case hardness results, discussed next.

1.3.1 Correlation Bounds for Depth Reduction Within AC^0

Our first application of our hardness amplification technique concerns the role of depth in circuit complexity. To what extent are deeper circuits more powerful than shallower circuits? In other words, what is the *marginal utility of time* for parallel computation?

Surprisingly, it turns out that in many contexts, circuits can be generically and nontrivially simulated by shallower circuits. For example:

- NC^1 circuits (i.e., circuits of depth $O(\log n)$ with bounded fan-in) can be simulated by AC_d^0 circuits of size $\exp(n^{O(1/d)})$ [73, 75, 76, 71].
- ACC_d^0 circuits (i.e., AC_d^0 circuits augmented with MOD_m gates) of size S can be simulated by $\text{SYM} \circ \text{AND}$ circuits of size $\exp((\log S)^{O(d)})$ [72, 2, 4, 81, 3, 11, 78, 24].
- AC^0 circuits can be approximated in various ways by low-degree polynomials [60, 66, 67, 10, 70, 55, 15, 37, 8, 59, 16, 69, 51, 34], which can be viewed as a “depth-two” model of computation.

In light of these remarkable “depth reduction” results and their numerous applications, we would like to know precisely when, and to what extent, depth reduction is possible. Indeed, there is a longstanding interest in thoroughly understanding the *hardness of circuit depth reduction* within AC^0 . Early work shows that there exists a linear-size AC_{d+1}^0 circuit $h: \{0, 1\}^n \rightarrow \{0, 1\}$ such that every AC_d^0 circuit computing h must have size $\exp(n^{\Omega(1/d)})$ [65, 80, 35]. For several decades, it was a stubborn open problem to prove a similar hierarchy theorem in the average-case setting. O’Donnell and Wimmer essentially resolved the depth-2 vs. depth-3 case [58], and then finally Håstad, Rossman, Servedio, and Tan resolved the general depth- d vs. depth- $(d + 1)$ case in a breakthrough last decade [39]:

► **Theorem 1.2** (The average-case depth hierarchy theorem [39]). *Let $n, d \in \mathbb{N}$ with $d \leq \frac{\alpha \log n}{\log \log n}$, where $\alpha > 0$ is a suitable constant. There is an explicit³ AC_{d+1}^0 circuit $h: \{0, 1\}^n \rightarrow \{0, 1\}$ of size $O(n)$ such that for every AC_d^0 circuit $g: \{0, 1\}^n \rightarrow \{0, 1\}$, either g has size $\exp(n^{\Omega(1/d)})$, or else the following correlation bound holds:*

$$\Pr_{\mathbf{x} \in \{0, 1\}^n} [g(\mathbf{x}) = h(\mathbf{x})] \leq \frac{1}{2} + n^{-\Omega(1/d)}. \quad (2)$$

² The exception, of course, is if we start from a lower bound against a stronger class such as $\text{MAJ} \circ \text{AC}^0$. See Klivans’ work [49].

³ I.e., the circuit h can be constructed in $\text{poly}(n)$ time, given the parameters n and d .

1:4 A Technique for Hardness Amplification Against AC^0

Theorem 1.2 asserts that h is *moderately* hard for AC_d^0 circuits. Håstad, Rossman, Servedio, and Tan identified two obstacles preventing significant improvement of the $n^{-\Omega(1/d)}$ correlation bound in (2):

- The “hard function” h in Theorem 1.2 is monotone. By the Kahn-Kalai-Linial theorem [47], every monotone Boolean function can be approximated by a constant or a variable with success probability $1/2 + \omega(1/n)$.
- By the discriminator lemma [33], every linear-size AC_{d+1}^0 circuit h , whether monotone or not, can be approximated by a linear-size AC_d^0 circuit with success probability $1/2 + \Omega(1/n)$. (See Hatami, Hoza, Tal, and Tell’s work for further details of these two arguments [40, Appendix A].)

In this work, we overcome both obstacles by using a different, non-monotone hard function h with depth slightly greater than $d + 1$. We prove an average-case lower bound for the task of simulating AC_{d+k}^0 circuits using AC_d^0 circuits, with a correlation bound that gets significantly stronger as k gets larger.

► **Theorem 1.3** (AC_d^0 circuits cannot approximate AC_{d+k}^0 circuits). *Let $n, d, k \in \mathbb{N}$ with $k \geq 3$ and $dk \leq \frac{\alpha \log n}{\log \log n}$, where $\alpha > 0$ is a suitable constant. There is an explicit AC_{d+k}^0 circuit $h: \{0, 1\}^n \rightarrow \{0, 1\}$ of size $O(n)$ such that for every AC_d^0 circuit $g: \{0, 1\}^n \rightarrow \{0, 1\}$, either g has size $\exp(n^{\Omega(1/d)})$, or else the following correlation bound holds:*

$$\Pr_{\mathbf{x} \in \{0,1\}^n} [g(\mathbf{x}) = h(\mathbf{x})] \leq \frac{1}{2} + \exp\left(-\frac{1}{d} \cdot \Omega(\log n)^{k-1}\right).$$

Our hard function h is the XOR of approximately $\log^{k-2} n$ many copies of Håstad, Rossman, Servedio, and Tan’s hard function [39]. By combining Theorem 1.3 with the Nisan-Wigderson framework [57] and a reduction due to Li and Zuckerman [54], we obtain new constructions of *seedless randomness extractors* that are computable by small $AC_{d+O(1)}^0$ circuits and that can extract from sources that are “recognizable” by large AC_d^0 circuits. See the full version of this paper for details [41].

1.3.2 Correlation Bounds for XOR of Majority

Our second application of our hardness amplification technique concerns the n -bit majority function (MAJ_n). It is well known that the majority function is moderately hard for AC^0 circuits and more generally for $AC^0[\oplus]$ circuits, i.e., AC^0 circuits augmented with parity gates.⁴ Specifically, based on the seminal works of Razborov and Smolensky [60, 66, 67], we have the following correlation bound.

► **Theorem 1.4** (Majority is moderately hard for $AC_d^0[\oplus]$ circuits). *Let $n, d, S \in \mathbb{N}$ with $S \geq n$. Let $g: \{0, 1\}^n \rightarrow \{0, 1\}$ be an $AC_d^0[\oplus]$ circuit of size S . Then*

$$\Pr_{\mathbf{x} \in \{0,1\}^n} [g(\mathbf{x}) = MAJ_n(\mathbf{x})] \leq \frac{1}{2} + \frac{O(\log S)^{d-1}}{\sqrt{n}}.$$

We emphasize that we are considering the problem of computing the majority function on a $(1/2 + \varepsilon)$ -fraction of n -bit inputs, which is distinct from the perhaps more famous “promise majority” problem in which we wish to compute the majority function on all inputs with relative Hamming weight outside the interval $1/2 \pm \varepsilon$. It seems that O’Donnell and Wimmer were the first to explicitly consider correlation bounds for the majority function [58].

⁴ Even more generally, we can consider MOD_q gates where q is a power of a prime – but let us focus on parity gates for simplicity.

The specific quantitative bound in Theorem 1.4 is actually a log-factor improvement over what was known before, to the best of our knowledge. We therefore include a proof of Theorem 1.4 in the full version of this paper [41, Appendix A]. (We also present a matching AC^0 construction based on prior work, showing that Theorem 1.4 is tight.) That being said, our main focus is on the qualitative distinction between functions that are “moderately hard” and functions that are “very hard.” The fact that the majority function is moderately hard for $\text{AC}^0[\oplus]$ circuits – for example, the correlation bound above is $\tilde{\Theta}(1/\sqrt{n})$ in the constant-depth polynomial-size regime – was already well-understood prior to this work.

Remarkably, this weak correlation bound is the best bound known on the correlation between $\text{AC}^0[\oplus]$ circuits and any hard function in NP .⁵ It is a major open problem to construct an explicit function that is provably “very hard” for $\text{AC}^0[\oplus]$ circuits. The function $\text{MAJ}_n^{\oplus t}$, perhaps with $t = \text{polylog}(n)$, seems like a reasonable candidate.

Chattopadhyay, Hatami, Hosseini, Lovett, and Zuckerman recently proved that XORing amplifies the hardness of MAJ_n for constant-degree \mathbb{F}_2 -polynomials [18], which can be considered a special case of polynomial-size $\text{AC}_2^0[\oplus]$ circuits. In this work, we consider a different special case of $\text{AC}^0[\oplus]$ circuits, namely AC^0 circuits. Our contribution is a proof that XORing amplifies the hardness of MAJ_n for AC^0 circuits.

► **Theorem 1.5** ($\text{MAJ}_n^{\oplus t}$ is hard for AC_d^0 circuits). *Let $n, t, d, S \in \mathbb{N}$ and let $g: \{0, 1\}^{nt} \rightarrow \{0, 1\}$ be an AC_d^0 circuit of size S . Then*

$$\Pr_{\mathbf{x} \in \{0, 1\}^{nt}} [g(\mathbf{x}) = \text{MAJ}_n^{\oplus t}(\mathbf{x})] \leq \frac{1}{2} + \left(\frac{O(\log S)^{d-1}}{\sqrt{n}} \right)^t.$$

1.4 Our Technique

1.4.1 XOR Lemmas for Decision Trees

Our correlation bounds are based on XOR lemmas for *decision trees*. Before explaining the connection between AC^0 circuits and decision trees, let us discuss the XOR lemmas for decision trees themselves – a fascinating subject in its own right. Let h be a Boolean function that is moderately hard for shallow decision trees: every depth- D decision tree agrees with h on at most a $(1/2 + \varepsilon)$ -fraction of inputs.

It is not hard to show that decision trees *of that same depth D* can compute $h^{\oplus t}$ on at most a $(1/2 + \varepsilon')$ -fraction of inputs, where $\varepsilon' = \frac{1}{2} \cdot (2\varepsilon)^t$. (For example, this is a special case of Shaltiel’s analysis of “fair” decision trees [61].) It turns out that a slight generalization of that simple analysis suffices for proving our correlation bound for depth reduction within AC^0 (Theorem 1.3).

On the other hand, to get the best parameters in Theorem 1.5 (on the hardness of $\text{MAJ}_n^{\oplus t}$), it turns out that we need a more sophisticated XOR lemma for decision trees, in which we allow the tree attempting to compute $h^{\oplus t}$ to have depth significantly larger than D .

This problem has been previously studied by Drucker [26]. Focusing on one setting of parameters, Drucker showed that for every constant $\alpha > 0$, there is a value $D' = \Omega(Dt)$ such that trees of depth D' cannot compute $h^{\oplus t}$ on more than a $(1/2 + \varepsilon')$ -fraction of inputs, where $\varepsilon' = O(\varepsilon)^{(1-\alpha) \cdot t}$ [26]. Although it comes close, this result is not quite sufficient to prove Theorem 1.5 because of the $(1 - \alpha)$ -factor loss in the exponent. Furthermore, unfortunately, the $(1 - \alpha)$ -factor loss is unavoidable in general, due to counterexamples

⁵ If we permit hard functions that satisfy less stringent explicitness conditions, then better correlation bounds are known against $\text{AC}^0[\oplus]$ and even stronger classes [77, 23, 22, 19].

identified by Shaltiel [61]. The idea behind these counterexamples is that although h is hard for decision trees of depth D , it might nevertheless be easy for decision trees of depth $D + 1$. In this case, for any constant $c > 0$, a decision tree of depth cDt can successfully compute h on $\Omega(t)$ independent instances.

To circumvent Shaltiel’s counterexamples [61], we strengthen the assumption. We assume that h is moderately hard for depth- D decision trees *for all D simultaneously*, with a correlation bound ε that scales with the depth D according to some log-concave function $\varepsilon(D)$. Under this assumption, we prove the decision trees of depth $\Omega(Dt)$ have correlation at most $O(\varepsilon)^t$ with $h^{\oplus t}$.

► **Lemma 1.6** (XOR lemma for decision trees under a robust hardness assumption). *Let $h: \{0, 1\}^n \rightarrow \{0, 1\}$ be a function and let $\varepsilon: [0, \infty) \rightarrow (0, \infty)$ be a log-concave function. Assume that for every $D \in \mathbb{N}$ and every decision tree $T: \{0, 1\}^n \rightarrow \{0, 1\}$ of depth at most D , we have*

$$\Pr_{\mathbf{x} \in \{0, 1\}^n} [T(\mathbf{x}) = h(\mathbf{x})] \leq \frac{1}{2} + \varepsilon(D).$$

Then for every $D, t \in \mathbb{N}$ and every decision tree $T: \{0, 1\}^{nt} \rightarrow \{0, 1\}$ of depth at most $Dt/2$, we have

$$\Pr_{\mathbf{x} \in \{0, 1\}^{nt}} [T(\mathbf{x}) = h^{\oplus t}(\mathbf{x})] \leq \frac{1}{2} + O(\varepsilon(D))^t.$$

(See Lemma 3.2 for a more general statement.)

1.4.2 Amplifying the Average-Case Depth Hierarchy Theorem

Now we briefly explain how we use an XOR lemma for decision trees to prove Theorem 1.3 (our correlation bound for depth reduction within AC^0). Our analysis builds on Håstad, Rossman, Servedio, and Tan’s proof of the average-case depth hierarchy theorem [39]. Recall that their lower bound proof is based on the concept of *random projections*, which generalize traditional random restrictions. (A traditional *restriction* assigns values to some input variables while keeping others “alive.” A *projection* can additionally merge living variables.) To prove that their hard function h is moderately hard for AC_d^0 circuits, Håstad, Rossman, Servedio, and Tan carefully designed a distribution \mathcal{R} over projections and a distribution μ over inputs and showed the following [39].

1. (Completion to the uniform distribution.) For every function $f: \{0, 1\}^n \rightarrow \{0, 1\}$, plugging a uniform random $\mathbf{x} \in \{0, 1\}^n$ into f is equivalent to first sampling a projection $\pi \sim \mathcal{R}$, then independently sampling an input $\mathbf{y} \sim \mu$, and finally plugging \mathbf{y} into $f|_{\pi}$.
2. (Simplification.) For every AC_d^0 circuit g , either g has size $\exp(n^{\Omega(1/d)})$, or else with high probability over $\pi \sim \mathcal{R}$, the circuit g *simplifies* under π in the sense that $g|_{\pi}$ can be computed by a shallow decision tree.
3. (Maintaining structure.) With high probability over $\pi \sim \mathcal{R}$, the hard function h *maintains structure* in the sense that $h|_{\pi}$ is moderately hard for shallow decision trees with respect to μ .

Taken together, the three steps above imply that h is moderately hard for AC_d^0 circuits with respect to a uniform random input. We call this proof structure the *random simplification method* for proving correlation bounds.

As mentioned previously, our hard function is $h^{\oplus t}$, where h is Håstad, Rossman, Servedio, and Tan’s hard function and $t \approx \log^{k-2} n$. To prove that $h^{\oplus t}$ is very hard for AC_d^0 circuits, we use the random simplification method. We apply \mathcal{R} to each of the t input blocks of $h^{\oplus t}$ independently. By Håstad, Rossman, Servedio, and Tan’s analysis [39], each copy of h is

likely to be moderately hard for shallow decision trees after the projection. Therefore, by a suitable XOR lemma for decision trees, $h^{\oplus t}$ is likely to be *very* hard for shallow decision trees after the projection. Meanwhile, Håstad, Rossman, Servedio, and Tan’s simplification arguments [39] extend to the case of several independent copies of \mathcal{R} , completing the proof.

1.4.3 Amplifying the Hardness of the Majority Function

There are at least three known proofs that the majority function is moderately hard for AC^0 circuits: one using the Razborov-Smolensky method [27, 50, 41], one due to O’Donnell and Wimmer [58], and one due to Tal [69]. However, none of these proofs fits into our framework of “random simplification arguments,” so it is not clear how to combine them with our amplification technique. (The latter two proofs do use switching lemmas, but only in an indirect Fourier-analytic way.) For this reason, in the full version of this paper [41, §5.1], we present yet another proof that the majority function is moderately hard for AC_d^0 circuits. Our new proof does fit into our “random simplification argument” framework, and furthermore, the “robust hardness assumption” of Lemma 1.6 is satisfied in our proof. These features of our proof enable us to apply our new XOR lemma for decision trees to complete our analysis of $MAJ_n^{\oplus t}$.

1.5 Related Work

Goldwasser, Gutfreund, Healy, Kaufman, and Rothblum designed a method for converting worst-case hardness into moderate average-case hardness in the context of weak circuit classes [30], which complements our work in some ways. One contrast between their work and ours is that they merely construct a hard function with a very weak explicitness guarantee, namely membership in EXP, whereas we study an extremely explicit hardness amplification method, namely XORing. More recently, Chen, Lu, Lyu, and Oliveira developed a method for constructing very hard functions for weak circuit classes starting from relatively weak assumptions [20] – but once again, their hard functions only satisfy weak explicitness guarantees such as membership in E.

A long sequence of works has established strong bounds on the correlation between the parity function and AC^0 circuits [28, 1, 80, 35, 36, 7, 49, 75, 9, 44, 38]. One of these works, by Klivans [49], is especially relevant for us. Klivans’ proof is based on a result by Aspnes, Beigel, Furst, and Rudich, who showed that if g is a $MAJ \circ AC_d^0$ circuit, then either g has size $\exp(n^{\Omega(1/d)})$, or else g disagrees with the parity function on a constant fraction of inputs [6]. Klivans combined this result with Yao’s XOR Lemma to re-prove a strong (albeit not optimal) bound on the correlation between AC_d^0 circuits and the parity function [49]. Klivans’ proof is the only prior work we are aware of that uses hardness amplification methods to prove an unconditional AC^0 circuit lower bound.

Many prior works have studied XOR lemmas for various types of decision trees, along with the closely related “direct product” and “direct sum” problems [45, 12, 56, 61, 48, 68, 5, 46, 26, 64, 52, 13, 14, 17]. However, as far as we are aware, we are the first to consider the case that we have hardness for all depths simultaneously.

1.6 Organization

After some preliminaries, we present our XOR lemma for decision trees (Lemma 1.6) in Section 3. Then, in Section 4, we present general lemmas showing that XORing amplifies hardness whenever the hardness is proved via the random simplification method. The proofs of our main results (Theorem 1.3 and Theorem 1.5) are omitted from this extended abstract, but they can be found in the full version of this paper [41].

2 Preliminaries

We write \mathbb{N} to denote the set of non-negative integers.

2.1 Boolean Functions

In the introduction, we worked with functions $f: \{0, 1\}^n \rightarrow \{0, 1\}$. Going forward, it will be more convenient to encode a bit $b \in \{0, 1\}$ as the value $(-1)^b$. Thus, we will work with functions $f: \{\pm 1\}^n \rightarrow \{\pm 1\}$. We continue to use the notation $f^{\oplus t}$, but now $f^{\oplus t}$ denotes the product of t copies of f on independent inputs.

We use the following notation to describe decision trees.

► **Definition 2.1** (Decision trees). *For a function $f: \{\pm 1\}^n \rightarrow \{\pm 1\}$, we define $\text{DTDepth}(f)$ to be the minimum depth of a decision tree computing f . In the other direction, for a parameter $D \in \mathbb{N}$, we define $\text{DTDepth}[D]$ to be the class of all functions $f: \{\pm 1\}^n \rightarrow \{\pm 1\}$ that can be computed by depth- D decision trees. (The parameter n will always be clear from context.)*

2.2 Probability and Correlation

We denote random variables using boldface. We write $\mathbf{x} \sim \mu$ to indicate that the random variable \mathbf{x} is sampled from the distribution μ . If $\mu, \tilde{\mu}$ are discrete probability distributions over some set Ω , then we consider the “total variation distance” between μ and $\tilde{\mu}$ to be

$$\max_{S \subseteq \Omega} (|\Pr[\mathbf{x} \in S] - \Pr[\tilde{\mathbf{x}} \in S]|),$$

where $\mathbf{x} \sim \mu$ and $\tilde{\mathbf{x}} \sim \tilde{\mu}$. We also rely on the following alternative notion of “distance” between probability distributions.

► **Definition 2.2** (Max-divergence). *Let μ and $\tilde{\mu}$ be discrete probability distributions over some set Ω . The max-divergence of $\tilde{\mu}$ from μ is defined by*

$$D_\infty(\tilde{\mu} \parallel \mu) = \ln \left(\max_{x \in \Omega} \left(\frac{\Pr[\tilde{\mathbf{x}} = x]}{\Pr[\mathbf{x} = x]} \right) \right),$$

where $\mathbf{x} \sim \mu$ and $\tilde{\mathbf{x}} \sim \tilde{\mu}$.

Max-divergence and total variation distance are related by the following lemma.

► **Lemma 2.3** (Low max-divergence \Rightarrow low total variation distance). *Let μ and $\tilde{\mu}$ be discrete probability distributions over the same set Ω . Let $\varepsilon = D_\infty(\tilde{\mu} \parallel \mu)$. There exists a probability distribution μ' such that μ can be written as a convex combination $\mu = (1 - \varepsilon) \cdot \tilde{\mu} + \varepsilon \cdot \mu'$. Moreover, the total variation distance between μ and $\tilde{\mu}$ is at most ε .*

Proof. If $\varepsilon = 0$, the lemma is trivial, so assume $\varepsilon > 0$. For each $x \in \Omega$, define

$$p(x) = \frac{\Pr[\mathbf{x} = x] - (1 - \varepsilon) \Pr[\tilde{\mathbf{x}} = x]}{\varepsilon},$$

where $\mathbf{x} \sim \mu$ and $\tilde{\mathbf{x}} \sim \tilde{\mu}$. Then $\sum_{x \in \Omega} p(x) = 1$. Furthermore, $p(x) \geq 0$, because

$$(1 - \varepsilon) \Pr[\tilde{\mathbf{x}} = x] \leq (1 - \varepsilon) \cdot e^\varepsilon \cdot \Pr[\mathbf{x} = x] \leq \Pr[\mathbf{x} = x].$$

Therefore, $p(\cdot)$ is a probability mass function, and we can let μ' be the corresponding probability distribution. For the “moreover” part, observe that for any $S \subseteq \Omega$, we have

$$\Pr[\mathbf{x} \in S] = (1 - \varepsilon) \cdot \Pr[\tilde{\mathbf{x}} \in S] + \varepsilon \cdot \Pr[\mathbf{x}' \in S],$$

where $\mathbf{x}' \sim \mu'$. Therefore,

$$\Pr[\mathbf{x} \in S] \leq \Pr[\tilde{\mathbf{x}} \in S] + \varepsilon \cdot \Pr[\mathbf{x}' \in S] \leq \Pr[\tilde{\mathbf{x}} \in S] + \varepsilon,$$

and

$$\Pr[\mathbf{x} \in S] \geq (1 - \varepsilon) \cdot \Pr[\tilde{\mathbf{x}} \in S] \geq \Pr[\tilde{\mathbf{x}} \in S] - \varepsilon. \quad \blacktriangleleft$$

We use the following notation for product distributions.

► **Definition 2.4** (Tensor product of probability distributions). *Let μ_1, \dots, μ_t be probability distributions over the spaces $\Omega_1, \dots, \Omega_t$. Sample $\mathbf{x}_1 \sim \mu_1, \dots, \mathbf{x}_t \sim \mu_t$ independently. The tensor product $\mu_1 \otimes \dots \otimes \mu_t$ is the probability distribution of $(\mathbf{x}_1, \dots, \mathbf{x}_t)$. As a special case, we define*

$$\mu^{\otimes t} = \underbrace{\mu \otimes \mu \otimes \dots \otimes \mu}_{t \text{ copies}}.$$

We use the following standard definition to reason about average-case hardness of $\{\pm 1\}$ -valued functions.

► **Definition 2.5** (Correlation). *Let $g, h: \{\pm 1\}^n \rightarrow \mathbb{R}$ be functions and let μ be a distribution over $\{\pm 1\}^n$. We define*

$$\text{Corr}_\mu(g, h) = \mathbb{E}_{\mathbf{x} \sim \mu} [g(\mathbf{x}) \cdot h(\mathbf{x})].$$

More generally, if \mathcal{C} is a class of functions $g: \{\pm 1\}^n \rightarrow \mathbb{R}$, then we define

$$\text{Corr}_\mu(\mathcal{C}, h) = \max_{g \in \mathcal{C}} \text{Corr}_\mu(g, h).$$

If μ is omitted, then by default it is assumed to be the uniform distribution over $\{\pm 1\}^n$.

If g and h are $\{\pm 1\}$ -valued, then a bound $|\text{Corr}(g, h)| \leq \varepsilon$ is equivalent to the statement that g agrees with h on at most a $(1/2 + \varepsilon/2)$ -fraction of inputs, because for any two $\{0, 1\}$ -valued random variables \mathbf{a}, \mathbf{b} , we have $\Pr[\mathbf{a} = \mathbf{b}] = \frac{1}{2} + \frac{1}{2} \mathbb{E}[(-1)^{\mathbf{a}} \cdot (-1)^{\mathbf{b}}]$.

2.3 Generalized Restrictions

To formulate our hardness amplification technique in the clearest and most general way possible, we work with a notion of *generalized restrictions* that includes restrictions and projections as special cases. A generalized restriction, formally defined below, consists of an arbitrary “preprocessing” step that can be applied to a Boolean function of interest.

► **Definition 2.6** (Generalized restriction). *A generalized restriction is a function $\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}^n$. If $f: \{\pm 1\}^n \rightarrow \{\pm 1\}$ is a Boolean function, then we define $g|_\pi$ to be the composition $g \circ \pi$. That is, $g|_\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}$ is given by $g|_\pi(x) = g(\pi(x))$.*

Traditional restrictions can be viewed as a special case of generalized restrictions as follows.

1:10 A Technique for Hardness Amplification Against AC^0

► **Definition 2.7** (Traditional restrictions as generalized restrictions). A restriction is a string $\rho \in \{+1, -1, \star\}^n$. We identify ρ with a generalized restriction $\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}^n$, where $r = |\rho^{-1}(\star)|$, as follows. Given $y \in \{\pm 1\}^r$, we let $\pi(y)$ be ρ , except that the i -th star is replaced with y_i for every $i \in [r]$.

Next, we consider *distributions* over generalized restrictions, and we explain how to interpret the tensor product of such distributions.

► **Definition 2.8** (Tensor product of generalized restriction distributions). Let $r, n \in \mathbb{N}$, and let \mathcal{R} be a distribution over generalized restrictions $\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}^n$. Let π_1, \dots, π_t be independent samples from \mathcal{R} , and define $\vec{\pi}: \{\pm 1\}^{rt} \rightarrow \{\pm 1\}^{nt}$ by concatenating, i.e.,

$$\vec{\pi}(y^{(1)}, \dots, y^{(t)}) = (\pi_1(y^{(1)}), \dots, \pi_t(y^{(t)})).$$

Then the tensor product $\mathcal{R}^{\otimes t}$ is the distribution of the random variable $\vec{\pi}$.

2.4 Logarithmic Concavity

We recall the following standard definition.

► **Definition 2.9** (Log-concave). A function $f: [0, \infty) \rightarrow (0, \infty)$ is log-concave if $\log f$ is concave, i.e., for every $x, y \in [0, \infty)$ and $\lambda \in (0, 1)$, we have $f(x)^\lambda \cdot f(y)^{1-\lambda} \leq f(\lambda x + (1-\lambda)y)$.

If f is log-concave, then by induction on t , we have $\prod_{i=1}^t f(x_i) \leq f(\bar{x})^t$ where $\bar{x} = \frac{1}{t} \sum_{i=1}^t x_i$.

3 XOR Lemmas for Decision Trees

In this section, we present our XOR lemma for decision trees. We begin by stating a simple XOR lemma, in which the decision tree attempting to compute $h^{\oplus t}$ has the same depth as the decision tree attempting to compute h .

► **Lemma 3.1** (Basic XOR lemma for decision trees). Let $h_1, \dots, h_t: \{\pm 1\}^r \rightarrow \{\pm 1\}$ be functions, and define $h(y^{(1)}, \dots, y^{(t)}) = \prod_{i=1}^t h_i(y^{(i)})$. Let μ be a distribution over $\{\pm 1\}^r$. For every $D \in \mathbb{N}$, we have

$$\text{Corr}_{\mu^{\otimes t}}(h, \text{DTDepth}[D]) \leq \prod_{i=1}^t \text{Corr}_{\mu}(h_i, \text{DTDepth}[D]).$$

We were unable to find a reference for the specific statement of Lemma 3.1, but it has no significant novelty. It is closely related to Shaltiel's analysis of "fair" decision trees [61]. It can also be viewed as a special case of Claim 3.6 that we prove below. As discussed in Subsection 1.4, Lemma 3.1 is sufficient for our analysis of depth- d approximators to AC_{d+k}^0 circuits (Theorem 1.3). However, for our analysis of $\text{MAJ}_n^{\oplus t}$ (Theorem 1.5), we need a more sophisticated XOR lemma, stated next.

► **Lemma 3.2** (XOR lemma for decision trees under robust hardness assumptions, general version). Let $h_1, \dots, h_t: \{\pm 1\}^r \rightarrow \{\pm 1\}$ be functions, and define $h(y^{(1)}, \dots, y^{(t)}) = \prod_{i=1}^t h_i(y^{(i)})$. Let μ_1, \dots, μ_t be distributions over $\{\pm 1\}^r$, and define $\mu = \mu_1 \otimes \dots \otimes \mu_t$. Let $\varepsilon: [0, \infty) \rightarrow (0, \infty)$ be a log-concave function, and assume that for every $i \in [t]$ and every $D \in \mathbb{N}$, we have

$$\text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D]) \leq \varepsilon(D).$$

Then for every $D \in \mathbb{N}$, we have

$$\text{Corr}_{\mu}(h, \text{DTDepth}[Dt/2]) \leq O(\varepsilon(D))^t.$$

The first step of the proof of Lemma 3.2 is the following claim, which enables us to relate the success probability of a tree to the success probabilities of its subtrees.

▷ **Claim 3.3 (Law of total correlation).** Let $h, T, E: \{\pm 1\}^r \rightarrow \{\pm 1\}$. Let μ be a distribution over $\{0, 1\}^r$. For each $b \in \{\pm 1\}$, let $p_b = \Pr_{\mathbf{y} \sim \mu}[E(\mathbf{y}) = b]$, and let μ^b be the conditional distribution ($\mathbf{y} \sim \mu \mid E(\mathbf{y}) = b$). Suppose that T can be decomposed in the form

$$T(y) = \begin{cases} T_{+1}(y) & \text{if } E(y) = +1 \\ T_{-1}(y) & \text{if } E(y) = -1 \end{cases}$$

for some $T_{+1}, T_{-1}: \{\pm 1\}^r \rightarrow \{\pm 1\}$. Then

$$\text{Corr}_\mu(h, T) = \sum_{b \in \{\pm 1\}} p_b \cdot \text{Corr}_{\mu^b}(h, T_b).$$

Proof.

$$\begin{aligned} \text{Corr}_\mu(h, T) &= \mathbb{E}_{\mathbf{y} \sim \mu} [h(\mathbf{y}) \cdot T(\mathbf{y})] \\ &= \sum_{b \in \{\pm 1\}} p_b \cdot \mathbb{E}_{\mathbf{y} \sim \mu} [h(\mathbf{y}) \cdot T(\mathbf{y}) \mid E(\mathbf{y}) = b] \quad (\text{Law of total expectation}) \\ &= \sum_{b \in \{\pm 1\}} p_b \mathbb{E}_{\mathbf{y} \sim \mu^b} [h(\mathbf{y}) \cdot T_b(\mathbf{y})]. \quad \triangleleft \end{aligned}$$

Next, we consider the following notion of “fair” decision trees due to Shaltiel [61].

▶ **Definition 3.4** ((D_1, \dots, D_t) -fair decision trees [61]). Let $T: \{\pm 1\}^{rt} \rightarrow \{\pm 1\}$ be a decision tree and let $D_1, \dots, D_t \in \mathbb{N}$. We say that T is (D_1, \dots, D_t) -fair if for every input $\vec{y} = (y^{(1)}, \dots, y^{(t)}) \in (\{\pm 1\}^r)^t$, for every $i \in [t]$, the computation $T(\vec{y})$ makes at most D_i queries to $y^{(i)}$.

The key to proving Lemma 3.2 is to generalize Definition 3.4 to the case of a set of tuples (D_1, \dots, D_t) .

▶ **Definition 3.5** (Q -fair decision trees). Let $T: \{\pm 1\}^{rt} \rightarrow \{\pm 1\}$ be a decision tree and let $Q \subseteq \mathbb{N}^t$. We say that T is Q -fair if for every input $\vec{y} = (y^{(1)}, \dots, y^{(t)}) \in (\{\pm 1\}^r)^t$, there is some tuple $(D_1, \dots, D_t) \in Q$ such that for every $i \in [t]$, the computation $T(\vec{y})$ makes at most D_i queries to $y^{(i)}$.

We emphasize that the tuple (D_1, \dots, D_t) is permitted to vary from one input \vec{y} to another. Therefore, the fact that a tree is Q -fair does not necessarily imply that there is some $(D_1, \dots, D_t) \in Q$ such that the tree is (D_1, \dots, D_t) -fair. Given the concept of Q -fairness, it is relatively straightforward to prove the following claim by induction on the depth of T . The claim generalizes the analysis by Shaltiel [61], who considered the case of (D_1, \dots, D_t) -fair decision trees and focused on the uniform distribution.

▷ **Claim 3.6 (XOR lemma for Q -fair decision trees).** Let $h_1, \dots, h_t: \{\pm 1\}^r \rightarrow \{\pm 1\}$ be functions, and define $h(y^{(1)}, \dots, y^{(t)}) = \prod_{i=1}^t h_i(y^{(i)})$. Let μ_1, \dots, μ_t be distributions over $\{\pm 1\}^r$, and define $\mu = \mu_1 \otimes \dots \otimes \mu_t$. Let $Q \subseteq \mathbb{N}^t$ and let $T: \{\pm 1\}^{rt} \rightarrow \{\pm 1\}$ be a Q -fair decision tree. Then

$$\text{Corr}_\mu(h, T) \leq \sum_{(D_1, \dots, D_t) \in Q} \prod_{i=1}^t \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i]).$$

1:12 A Technique for Hardness Amplification Against AC⁰

Proof. Assume without loss of generality that T never queries the same variable twice. For the base case, if T has depth 0, then T is a constant function, so

$$|\text{Corr}_\mu(h, T)| = \prod_{i=1}^t \left| \mathbb{E}_{\mathbf{y}^{(i)} \sim \mu_i} [h_i(y^{(i)})] \right| = \prod_{i=1}^t \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[0]).$$

Since T is Q -fair, Q must be nonempty. The lemma follows because $\text{Corr}_{\mu_i}(h_i, \text{DTDepth}[0]) \leq \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i])$ for every $D_i \in \mathbb{N}$. For the inductive step, let $y_{j_*}^{(i_*)}$ be the variable queried by the root of the tree. Let T_{+1} and T_{-1} be the children of the root, corresponding to the cases $y_{j_*}^{(i_*)} = +1$ and $y_{j_*}^{(i_*)} = -1$ respectively. Define

$$Q' = \{(D_1, \dots, D_{i_*-1}, D_{i_*} - 1, D_{i_*+1}, \dots, D_t) : (D_1, \dots, D_t) \in Q \text{ and } D_{i_*} \neq 0\}.$$

Then T_{+1} and T_{-1} are both Q' -fair.

For each $b \in \{\pm 1\}$, define

$$p_b = \Pr_{\mathbf{y}^{(i_*)} \sim \mu_{i_*}} \left[\mathbf{y}_{j_*}^{(i_*)} = b \right].$$

Let $\mu_{i_*}^b$ be the conditional distribution $(\mathbf{y}^{(i_*)} \sim \mu_{i_*} \mid \mathbf{y}_{j_*}^{(i_*)} = b)$, and for $i \neq i_*$, let $\mu_i^b = \mu_i$. Let $\mu^b = \mu_1^b \otimes \dots \otimes \mu_t^b$. By Claim 3.3 and the induction hypothesis, we have

$$\begin{aligned} \text{Corr}_\mu(h, T) &= \sum_{b \in \{\pm 1\}} p_b \cdot \text{Corr}_{\mu^b}(h, T_b) \\ &\leq \sum_{b \in \{\pm 1\}} p_b \cdot \sum_{(D_1, \dots, D_t) \in Q'} \prod_{i=1}^t \text{Corr}_{\mu_i^b}(h_i, \text{DTDepth}[D_i]) \\ &= \sum_{(D_1, \dots, D_t) \in Q'} \left(\sum_{b \in \{\pm 1\}} p_b \cdot \text{Corr}_{\mu_{i_*}^b}(h_{i_*}, \text{DTDepth}[D_{i_*}]) \right) \cdot \Pi_{\neq i_*}(D_1, \dots, D_t) \end{aligned}$$

where $\Pi_{\neq i_*}(D_1, \dots, D_t) = \prod_{i \in [t], i \neq i_*} \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i])$. Now we bound the inner sum. By Claim 3.3, for any D_{i_*} , we have

$$\text{Corr}_{\mu_{i_*}^b}(h_{i_*}, \text{DTDepth}[D_{i_*} + 1]) \geq \sum_{b \in \{\pm 1\}} p_b \cdot \text{Corr}_{\mu_{i_*}^b}(h_{i_*}, \text{DTDepth}[D_{i_*}]),$$

because we can approximate h_{i_*} with respect to μ_{i_*} by first querying $y_{j_*}^{(i_*)}$ and then using optimal subtrees of depth D_{i_*} . For every $(D_1, \dots, D_t) \in Q'$, we have $(D_1, \dots, D_{i_*-1}, D_{i_*} + 1, D_{i_*+1}, \dots, D_t) \in Q$. Therefore,

$$\text{Corr}_\mu(h, T) \leq \sum_{(D_1, \dots, D_t) \in Q} \prod_{i=1}^t \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i]). \quad \triangleleft$$

Given Claim 3.6, our XOR lemma for decision trees under a robust hardness assumption (Lemma 3.2) readily follows, as we now show.

Proof of Lemma 3.2. Let $T: \{\pm 1\}^{rt} \rightarrow \{\pm 1\}$ be a decision tree of depth at most $Dt/2$. Let Q be the set of t -tuples $(D_1, \dots, D_t) \in \mathbb{N}^t$ such that (1) $D_1 + \dots + D_t \leq Dt$ and (2) D_i is an integer multiple of $\lceil D/2 \rceil$ for every i . We claim that T is Q -fair. Indeed, let $\vec{y} = (y^{(1)}, \dots, y^{(t)})$ be any input, and let D_i be the number of queries that $T(\vec{y})$ makes to $y^{(i)}$. Let D'_i be the smallest integer multiple of $\lceil D/2 \rceil$ such that $D_i \leq D'_i$. Then $D'_i \leq D_i + (\lceil D/2 \rceil - 1)$, and hence $D'_1 + \dots + D'_t \leq Dt/2 + t \cdot (\lceil D/2 \rceil - 1) \leq Dt$, showing that $(D'_1, \dots, D'_t) \in Q$.

Therefore, by Claim 3.6,

$$\text{Corr}_\mu(h, T) \leq \sum_{(D_1, \dots, D_t) \in Q} \prod_{i=1}^t \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i]).$$

For any $(D_1, \dots, D_t) \in Q$, we can define (D'_1, \dots, D'_t) such that $D'_i \geq D_i$ and $D'_1 + \dots + D'_t$ is *exactly* Dt rather than being at most Dt . Then $\text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D_i]) \leq \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D'_i])$, so

$$\begin{aligned} \text{Corr}_\mu(h, T) &\leq \sum_{(D_1, \dots, D_t) \in Q} \prod_{i=1}^t \text{Corr}_{\mu_i}(h_i, \text{DTDepth}[D'_i]) \\ &\leq \sum_{(D_1, \dots, D_t) \in Q} \prod_{i=1}^t \varepsilon(D'_i) \\ &\leq \sum_{(D_1, \dots, D_t) \in Q} \varepsilon(D)^t && \text{(Log-concavity)} \\ &= |Q| \cdot \varepsilon(D)^t. \end{aligned}$$

To bound $|Q|$, observe that if $(D_1, \dots, D_t) \in Q$, then we can write $D_i = c_i \cdot \lceil D/2 \rceil$ for some nonnegative integers c_1, \dots, c_t . Furthermore, $Dt \geq \sum_i c_i \cdot \lceil D/2 \rceil \geq (D/2) \cdot \sum_i c_i$, so $c_1 + \dots + c_t \leq 2t$. Therefore, $|Q|$ is at most the number of ways that $2t$ can be partitioned into $t + 1$ nonnegative integers, which is precisely $\binom{3t}{t}$. Thus,

$$\text{Corr}_\mu(h, T) \leq \binom{3t}{t} \cdot \varepsilon(D)^t \leq O(\varepsilon(D))^t. \quad \blacktriangleleft$$

4 XOR Lemmas for the Random Simplification Method

In this section, we prove two general ‘‘XOR lemmas for the random simplification method,’’ which formalize our hardness amplification technique. The first and simpler version is as follows.

► **Lemma 4.1** (XOR lemma for the random simplification method, basic version). *Let $n, t, r, D \in \mathbb{N}$ and $\varepsilon, \delta > 0$. Let $h: \{\pm 1\}^n \rightarrow \{\pm 1\}$ and $g: \{\pm 1\}^{nt} \rightarrow \{\pm 1\}$ be Boolean functions, let \mathcal{R} be a distribution over generalized restrictions $\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}^n$, let μ be a distribution over $\{\pm 1\}^r$, and assume the following.*

1. *(The distribution μ completes \mathcal{R} to the uniform distribution.) If we sample $\pi \sim \mathcal{R}$ and $\mathbf{y} \sim \mu$ independently, then $\pi(\mathbf{y})$ is a uniform random element of $\{\pm 1\}^n$.*
2. *(The function g simplifies under $\mathcal{R}^{\otimes t}$.) We have*

$$\Pr_{\pi \sim \mathcal{R}^{\otimes t}} [\text{DTDepth}(g|\pi) > D] \leq \delta.$$

3. *(The function h retains structure under \mathcal{R} .) We have*

$$\mathbb{E}_{\pi \sim \mathcal{R}} [\text{Corr}_\mu(h|\pi, \text{DTDepth}[D])] \leq \varepsilon.$$

Then $\text{Corr}(g, h^{\oplus t}) \leq \varepsilon^t + \delta$.

1:14 A Technique for Hardness Amplification Against AC^0

Proof. Sample $\vec{\pi} = (\pi_1, \dots, \pi_t) \sim \mathcal{R}^{\otimes t}$ and $\vec{y} \sim \mu^{\otimes t}$ independently. Let \mathbf{T} be $g|_{\vec{\pi}}$ if $\text{DTDepth}(g|_{\vec{\pi}}) \leq D$; otherwise, let \mathbf{T} be the constant-zero function. Assumption 1 implies that $\vec{\pi}(\vec{y})$ is distributed uniformly over $\{\pm 1\}^{nt}$. Therefore,

$$\begin{aligned}
\text{Corr}(h^{\oplus t}, g) &= \mathbb{E}_{\vec{\pi}} [\text{Corr}_{\mu^{\oplus t}}(h^{\oplus t}|_{\vec{\pi}}, g|_{\vec{\pi}})] && \text{(Assumption 1)} \\
&\leq \delta + \mathbb{E}_{\vec{\pi}} [\text{Corr}_{\mu^{\oplus t}}(h^{\oplus t}|_{\vec{\pi}}, \mathbf{T})] && \text{(Assumption 2)} \\
&\leq \delta + \mathbb{E}_{\vec{\pi}} [\text{Corr}_{\mu^{\oplus t}}(h^{\oplus t}|_{\vec{\pi}}, \text{DTDepth}[D])] \\
&\leq \delta + \mathbb{E}_{\vec{\pi}} \left[\prod_{i=1}^t \text{Corr}_{\mu}(h|_{\pi_i}, \text{DTDepth}[D]) \right] && \text{(Lemma 3.1)} \\
&= \delta + \left(\mathbb{E}_{\pi \sim \mathcal{R}} [\text{Corr}_{\mu}(h|_{\pi}, \text{DTDepth}[D])] \right)^t && \text{(Independence)} \\
&\leq \delta + \varepsilon^t && \text{(Assumption 3.)} \quad \blacktriangleleft
\end{aligned}$$

In the full version of this paper [41], we review the basic structure of Håstad, Rossman, Servedio, and Tan’s proof of the average-case depth hierarchy theorem [39] and explain how it fits into the framework of Lemma 4.1. As a result, we are able to use Lemma 4.1 to prove our result about the average-case hardness of AC_{d+k}^0 circuits for AC_d^0 circuits (Theorem 1.3).

In this extended abstract, let us focus on the hardness amplification technique itself. The conclusion of Lemma 4.1 is $\text{Corr}(g, h^{\oplus t}) \leq \varepsilon^t + \delta$. The “+ δ ” term is unfortunate, since it does not improve with increasing t . To address this weakness, we now prove a more sophisticated version of Lemma 4.1 in which the correlation bound is $O(\varepsilon)^t$, with no “+ δ ” term, albeit under stronger assumptions.

► **Lemma 4.2** (Tighter XOR lemma for the random simplification method). *Let $n, t \in \mathbb{N}$ and let $h: \{\pm 1\}^n \rightarrow \{\pm 1\}$ be a Boolean function. Let \mathcal{C} be a class of Boolean functions $g: \{\pm 1\}^{nt} \rightarrow \{\pm 1\}$ that is closed under restrictions.⁶ Let $r \in \mathbb{N}$, let \mathcal{R} be a distribution over generalized restrictions $\pi: \{\pm 1\}^r \rightarrow \{\pm 1\}^n$, and let μ be a distribution over $\{\pm 1\}^r$. Let $\varepsilon > 0$, and assume the following.*

1. (The distribution μ approximately completes \mathcal{R} to the uniform distribution.) *If we sample $\pi \sim \mathcal{R}$ and $\mathbf{y} \sim \mu$ independently, and we sample $\mathbf{x} \in \{\pm 1\}^n$ uniformly at random, then $D_{\infty}(\pi(\mathbf{y}) \parallel \mathbf{x}) \leq \varepsilon$.*
2. (The class \mathcal{C} simplifies under $\mathcal{R}^{\otimes t}$.) *For every $g \in \mathcal{C}$ and every $D \in \mathbb{N}$, we have*

$$\Pr_{\vec{\pi} \sim \mathcal{R}^{\otimes t}} [\text{DTDepth}(g|_{\vec{\pi}}) \geq D] \leq 2^{t-D}.$$

3. (The function h retains structure under \mathcal{R} .) *For every $D \in \mathbb{N}$ and every $\pi \in \text{Supp}(\mathcal{R})$, we have*

$$\text{Corr}_{\mu}(h|_{\pi}, \text{DTDepth}[D]) \leq \varepsilon \cdot 2^{D/3}.$$

Then $\text{Corr}(\mathcal{C}, h^{\oplus t}) \leq O(\varepsilon)^t$.

Proof. Fix any $g \in \mathcal{C}$. Our job is to analyze the correlation between g and $h^{\oplus t}$ under a uniform random input. By Lemma 2.3, we can sample a uniform random input by the following procedure.

⁶ Every function in \mathcal{C} has domain $\{\pm 1\}^{nt}$. When we say that \mathcal{C} is closed under restrictions, we are thinking of a restriction of $g \in \mathcal{C}$ as another function on nt bits that ignores some of its input variables.

1. Sample $\vec{\pi} = (\pi_1, \dots, \pi_t) \sim \mathcal{R}^{\otimes t}$.
2. Sample $\vec{y} = (\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(t)}) \sim \mu^{\otimes t}$.
3. Sample $\vec{\mathbf{e}} = (\mathbf{e}^{(1)}, \dots, \mathbf{e}^{(t)}) \sim (\mu')^{\otimes t}$, where μ' is the distribution over $\{\pm 1\}^n$ from Lemma 2.3.
4. Sample $\mathbf{I} \subseteq [t]$ where $\Pr[i \in \mathbf{I}] = 1 - \varepsilon$ independently for every i .
5. Output the string $\vec{\mathbf{x}} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}) \in \{\pm 1\}^{nt}$, where

$$\mathbf{x}^{(i)} = \begin{cases} \pi_i(\mathbf{y}^{(i)}) & \text{if } i \in \mathbf{I} \\ \mathbf{e}^{(i)} & \text{if } i \notin \mathbf{I}. \end{cases}$$

Let $\mathbf{g}: \{\pm 1\}^{nt} \rightarrow \{\pm 1\}$ be the function obtained from g by plugging $\mathbf{e}^{(i)}$ into each block $i \notin \mathbf{I}$ and leaving the blocks in \mathbf{I} alive. Since \mathbf{g} ignores the variables in blocks outside \mathbf{I} , we have

$$g(\vec{\mathbf{x}}) = \mathbf{g}|_{\vec{\pi}}(\vec{y}).$$

Similarly, define $\mathbf{h}: \{\pm 1\}^{nt} \rightarrow \{\pm 1\}$ by the formula

$$\mathbf{h}(\vec{x}) = \left(\prod_{i \in \mathbf{I}} h_i(x^{(i)}) \right) \cdot \left(\prod_{i \notin \mathbf{I}} h_i(\mathbf{e}^{(i)}) \right),$$

so that $h^{\oplus t}(\vec{\mathbf{x}}) = \mathbf{h}|_{\vec{\pi}}(\vec{y})$. That way,

$$\text{Corr}(g, h^{\oplus t}) = \mathbb{E}[g(\vec{\mathbf{x}}) \cdot h^{\oplus t}(\vec{\mathbf{x}})] = \mathbb{E}_{\mathbf{I}, \vec{\mathbf{e}}, \vec{\pi}}[\text{Corr}_{\mu^{\otimes t}}(\mathbf{g}|_{\vec{\pi}}, \mathbf{h}|_{\vec{\pi}})].$$

Let $\mathbf{D} = \lfloor \text{DTDepth}(\mathbf{g}|_{\vec{\pi}}) / |\mathbf{I}| \rfloor$. Then

$$\mathbb{E}_{\mathbf{I}, \vec{\mathbf{e}}, \vec{\pi}}[\text{Corr}_{\mu^{\otimes t}}(\mathbf{g}|_{\vec{\pi}}, \mathbf{h}|_{\vec{\pi}})] \leq \mathbb{E}_{\mathbf{I}, \vec{\mathbf{e}}, \vec{\pi}}[\text{Corr}_{\mu^{\otimes t}}(\mathbf{h}|_{\vec{\pi}}, \text{DTDepth}[(\mathbf{D} + 1) \cdot |\mathbf{I}|])].$$

Let $\pi_{\mathbf{I}} = (\pi_i)_{i \in \mathbf{I}}$, and define $h_{\mathbf{I}}: \{\pm 1\}^{n|\mathbf{I}|} \rightarrow \{\pm 1\}$ by $h_{\mathbf{I}}((x^{(i)})_{i \in \mathbf{I}}) = \prod_{i \in \mathbf{I}} h(x^{(i)})$. Then for any fixing of $\mathbf{I}, \vec{\mathbf{e}}, \vec{\pi}$, we have

$$\text{Corr}_{\mu^{\otimes t}}(\mathbf{h}|_{\vec{\pi}}, \text{DTDepth}[(\mathbf{D} + 1) \cdot |\mathbf{I}|]) = \text{Corr}_{\mu^{\otimes |\mathbf{I}|}}(h_{\mathbf{I}}|_{\pi_{\mathbf{I}}}, \text{DTDepth}[(\mathbf{D} + 1) \cdot |\mathbf{I}|]).$$

Now we apply Lemma 3.2. For each $i \in \mathbf{I}$ and each $D \in \mathbb{N}$, we have

$$\text{Corr}_{\mu}(h|_{\pi_i}, \text{DTDepth}[D]) \leq \varepsilon \cdot 2^{D/3}.$$

Furthermore, the function $\varepsilon(D) = \varepsilon \cdot 2^{D/3}$ is log-concave. Therefore, Lemma 3.2 guarantees that

$$\text{Corr}_{\mu^{\otimes |\mathbf{I}|}}(h_{\mathbf{I}}|_{\pi_{\mathbf{I}}}, \text{DTDepth}[(\mathbf{D} + 1) \cdot |\mathbf{I}|]) \leq O(\varepsilon \cdot 2^{2 \cdot (\mathbf{D} + 1) / 3})^{|\mathbf{I}|} = O(\varepsilon \cdot 2^{2\mathbf{D}/3})^{|\mathbf{I}|}.$$

Thus, overall, we get

$$\text{Corr}(g, h^{\oplus t}) \leq \mathbb{E}_{\mathbf{I}, \vec{\mathbf{e}}, \vec{\pi}} \left[O(\varepsilon \cdot 2^{2\mathbf{D}/3})^{|\mathbf{I}|} \right].$$

Now consider any fixing of \mathbf{I} and $\vec{\mathbf{e}}$. Since \mathcal{C} is closed under restrictions, $\mathbf{g} \in \mathcal{C}$. Therefore, our simplification assumption tells us that for every $D \in \mathbb{N}$, we have

$$\Pr_{\vec{\pi}}[\mathbf{D} = D] \leq 2^{t-D \cdot |\mathbf{I}|}.$$

1:16 A Technique for Hardness Amplification Against AC^0

Consequently,

$$\begin{aligned} \mathbb{E}_{\pi} \left[O \left(\varepsilon \cdot 2^{2\mathbf{D}/3} \right)^{|\mathbf{I}|} \right] &= \sum_{D=0}^{\infty} \Pr_{\pi}[\mathbf{D} = D] \cdot O \left(\varepsilon \cdot 2^{2D/3} \right)^{|\mathbf{I}|} \\ &\leq 2^t \cdot \sum_{D=0}^{\infty} O \left(\varepsilon \cdot 2^{-D/3} \right)^{|\mathbf{I}|} \\ &\leq 2^t \cdot O \left(\sum_{D=0}^{\infty} \varepsilon \cdot 2^{-D/3} \right)^{|\mathbf{I}|} \\ &= 2^t \cdot O(\varepsilon)^{|\mathbf{I}|}. \end{aligned}$$

Therefore, our overall bound is given by

$$\begin{aligned} \text{Corr}(g, h^{\oplus t}) &\leq \mathbb{E}_{\mathbf{I}} [2^t \cdot O(\varepsilon)^{|\mathbf{I}|}] = 2^t \cdot \sum_{I \subseteq [t]} \Pr[\mathbf{I} = I] \cdot O(\varepsilon)^{|I|} \\ &= 2^t \cdot \sum_{I \subseteq [t]} (1 - \varepsilon)^{|I|} \cdot \varepsilon^{t-|I|} \cdot O(\varepsilon)^{|I|} \\ &\leq O(\varepsilon)^t. \end{aligned} \quad \blacktriangleleft$$

In the full version of this paper [41], we explain how to use Lemma 4.2 to prove our correlation bound for $\text{MAJ}_n^{\oplus t}$ (Theorem 1.5).

5 Directions for Further Research

The main open question related to our work is whether XORing always amplifies hardness for AC^0 circuits (cf. Theorem 1.1). We wish to also highlight the problem of proving *tight* correlation bounds for depth reduction within AC^0 (cf. Theorem 1.3). That is, what is the correlation between linear-size AC_{d+k}^0 circuits and near-exponential-size AC_d^0 circuits?

For simplicity, let us consider the case that d and k are both constants. As discussed previously, the extreme case $k = 1$ (i.e., using AC_d^0 circuits to approximate AC_{d+1}^0 circuits) is resolved by Håstad, Rossman, Servedio, and Tan's work [39] to within polynomial factors; the optimal correlation bound is $n^{\Theta(1)}$. Prior work also implies near-matching upper and lower bounds in the opposite extreme case $d = 1$ (i.e., using AC_1^0 circuits to approximate AC_{1+k}^0 circuits). In this case, it turns out that the optimal correlation bound is $\exp(-\tilde{\Theta}(\log^k n))$. (The approximators are based on the Linial-Nisan-Mansour theorem [55]; see the full version of this paper [41, Appendix B] for details.)

Based on those two extreme cases, it is tempting to conjecture that for all d and k , the optimal correlation bound should be $\exp(-\tilde{\Theta}(\log^k n))$, but in truth it is not at all clear that this is the best guess. Arguably the most interesting case is $k = 2$, i.e., the problem of using AC_d^0 circuits to approximate AC_{d+2}^0 circuits. On the one hand, the best method we know for constructing such an approximator is simply to use an optimal AC_1^0 approximator. On the other hand, the best correlation bound we know for this case is Håstad, Rossman, Servedio, and Tan's bound [39]. We therefore have a considerable gap between the upper and lower correlation bounds for this case, namely $n^{-\Omega(1)}$ vs. $n^{-\tilde{O}(\log^d n)}$.

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 Eric Allender. A note on the power of threshold circuits. In *Proceedings of the 30th Symposium on Foundations of Computer Science (FOCS)*, pages 580–584, 1989. doi:10.1109/SFCS.1989.63538.
- 3 Eric Allender and Vivek Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23(5):1026–1049, 1994. doi:10.1137/S0097539792233907.
- 4 Eric Allender and Ulrich Hertrampf. Depth reduction for circuits of unbounded fan-in. *Inform. and Comput.*, 112(2):217–238, 1994. doi:10.1006/inco.1994.1057.
- 5 Andris Ambainis, Robert Špalek, and Ronald de Wolf. A new quantum lower bound method, with applications to direct product theorems and time-space tradeoffs. *Algorithmica*, 55(3):422–461, 2009. doi:10.1007/s00453-007-9022-9.
- 6 James Aspnes, Richard Beigel, Merrick Furst, and Steven Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):135–148, 1994. doi:10.1007/BF01215346.
- 7 László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Inform. Process. Lett.*, 26(1):51–53, 1987. doi:10.1016/0020-0190(87)90036-6.
- 8 Louay M. J. Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM J. Comput.*, 38(6):2220–2272, 2009. doi:10.1137/070691954.
- 9 Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating AC^0 by small height decision trees and a deterministic algorithm for $\#AC^0SAT$. In *27th Conference on Computational Complexity (CCC)*, pages 117–125, 2012. doi:10.1109/CCC.2012.40.
- 10 Richard Beigel, Nick Reingold, and Daniel A. Spielman. The perceptron strikes back. In *Proceedings of the 6th Annual Structure in Complexity Theory Conference (SCT)*, pages 286–291, 1991. doi:10.1109/SCT.1991.160270.
- 11 Richard Beigel and Jun Tarui. On ACC. *Comput. Complexity*, 4(4):350–366, 1994. doi:10.1007/BF01263423.
- 12 Yosi Ben-Asher and Ilan Newman. Decision trees with and, or queries. In *Proceedings of the 10th Conference on Structure in Complexity Theory (SCT)*, pages 74–81, 1995. doi:10.1109/SCT.1995.514729.
- 13 Shalev Ben-David and Robin Kothari. Randomized query complexity of sabotaged and composed functions. *Theory Comput.*, 14:Paper No. 5, 27, 2018. doi:10.4086/toc.2018.v014a005.
- 14 Eric Blais and Joshua Brody. Optimal Separation and Strong Direct Sum for Randomized Query Complexity. In *Proceedings of the 34th Computational Complexity Conference (CCC)*, pages 29:1–29:17, 2019. doi:10.4230/LIPIcs.CCC.2019.29.
- 15 Ravi B. Boppana. The average sensitivity of bounded-depth circuits. *Information Processing Letters*, 63(5):257–261, 1997. doi:10.1016/S0020-0190(97)00131-2.
- 16 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *Journal of the ACM*, 57(5), 2010.
- 17 Joshua Brody, Jae Tak Kim, Peem Lerduptipongporn, and Hariharan Srinivasulu. A strong XOR lemma for randomized query complexity, 2020. arXiv:2007.05580.
- 18 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, Shachar Lovett, and David Zuckerman. XOR lemmas for resilient functions against polynomials. In *Proceedings of the 52nd Symposium on Theory of Computing (STOC)*, pages 234–246, 2020. doi:10.1145/3357713.3384242.
- 19 Lijie Chen. New Lower Bounds and Derandomization for ACC, and a Derandomization-Centric View on the Algorithmic Method. In *14th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 34:1–34:15, 2023. doi:10.4230/LIPIcs.ITCS.2023.34.
- 20 Lijie Chen, Zhenjian Lu, Xin Lyu, and Igor C. Oliveira. Majority vs. approximate linear sum and average-case complexity below NC^1 . In *48th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 51:1–51:20, 2021. doi:10.4230/LIPIcs.ICALP.2021.51.

- 21 Lijie Chen and Xin Lyu. Inverse-exponential correlation bounds and extremely rigid matrices from a new derandomized XOR lemma. In *Proceedings of the 53rd Symposium on Theory of Computing (STOC)*, pages 761–771, 2021. doi:10.1145/3406325.3451132.
- 22 Lijie Chen, Xin Lyu, and R. Ryan Williams. Almost-everywhere circuit lower bounds from non-trivial derandomization. In *61st Symposium on Foundations of Computer Science (FOCS)*, pages 1–12, 2020. doi:10.1109/FOCS46700.2020.00009.
- 23 Lijie Chen and Hanlin Ren. Strong average-case circuit lower bounds from nontrivial derandomization. *SIAM Journal on Computing*, 51(3):STOC20–115–STOC20–173, 2022. doi:10.1137/20M1364886.
- 24 Shiteng Chen and Periklis A. Papakonstantinou. Depth reduction for composites. *SIAM J. Comput.*, 48(2):668–686, 2019. doi:10.1137/17M1129672.
- 25 Yeyuan Chen, Yizhi Huang, Jiayu Li, and Hanlin Ren. Range avoidance, remote point, and hard partial truth table via satisfying-pairs algorithms. In *Proceedings of the 55th Symposium on Theory of Computing (STOC)*, pages 1058–1066, 2023. doi:10.1145/3564246.3585147.
- 26 Andrew Drucker. Improved direct product theorems for randomized query complexity. *Comput. Complexity*, 21(2):197–244, 2012. doi:10.1007/s00037-012-0043-7.
- 27 Yuval Filmus. Smolensky’s lower bound. Unpublished, 2010. URL: <https://yuvalfilmus.cs.technion.ac.il/Manuscripts/Smolensky.pdf>.
- 28 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.
- 29 Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao’s XOR-lemma. In *Studies in complexity and cryptography*, volume 6650 of *Lecture Notes in Comput. Sci.*, pages 273–301. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_23.
- 30 Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. Verifying and decoding in constant depth. In *Proceedings of the 39th Symposium on Theory of Computing (STOC)*, pages 440–449, 2007. doi:10.1145/1250790.1250855.
- 31 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *Proceedings of the 59th Symposium on Foundations of Computer Science (FOCS)*, pages 956–966, 2018. doi:10.1109/FOCS.2018.00094.
- 32 Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In *Proceedings of the 12th International Conference on Randomization and Computation (RANDOM)*, pages 455–468, 2008. doi:10.1007/978-3-540-85363-3_36.
- 33 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46(2):129–154, 1993. doi:10.1016/0022-0000(93)90001-D.
- 34 Prahlahd Harsha and Srikanth Srinivasan. On polynomial approximations to AC^0 . *Random Structures Algorithms*, 54(2):289–303, 2019. doi:10.1002/rsa.20786.
- 35 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Symposium on Theory of Computing (STOC)*, pages 6–20, 1986. doi:10.1145/12130.12132.
- 36 Johan Håstad. *Computational limitations for small depth circuits*. PhD thesis, Massachusetts Institute of Technology, 1986.
- 37 Johan Håstad. A slight sharpening of LMN. *Journal of Computer and System Sciences*, 63(3):498–508, 2001. doi:10.1006/jcss.2001.1803.
- 38 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014. doi:10.1137/120897432.
- 39 Johan Håstad, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for Boolean circuits. *J. ACM*, 64(5):Art. 35, 27, 2017. doi:10.1145/3095799.
- 40 Pooya Hatami, William M. Hoza, Avishay Tal, and Roei Tell. Depth- d threshold circuits vs. depth- $(d + 1)$ and-or trees. In *Proceedings of the 55th Symposium on Theory of Computing (STOC)*, pages 895–904, 2023. Full version: <https://ecc.weizmann.ac.il/report/2022/087/>. doi:10.1145/3564246.3585216.

- 41 William M. Hoza. A technique for hardness amplification against AC^0 . <https://eccc.weizmann.ac.il/report/2023/176/>, 2023.
- 42 Xuanguai Huang and Emanuele Viola. Average-case rigidity lower bounds. In *Proceedings of the 16th International Computer Science Symposium in Russia (CSR)*, pages 186–205, 2021. doi:10.1007/978-3-030-79416-3_11.
- 43 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *36th Symposium on Foundations of Computer Science (FOCS)*, pages 538–545, 1995. doi:10.1109/SFCS.1995.492584.
- 44 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0 . In *Proceedings of the 23rd Symposium on Discrete Algorithms (SODA)*, pages 961–972, 2012. doi:10.1137/1.9781611973099.77.
- 45 Russell Impagliazzo, Ran Raz, and Avi Wigderson. A direct product theorem. In *Proceedings of 9th Annual Conference on Structure in Complexity Theory (SCT)*, pages 88–96, 1994. doi:10.1109/SCT.1994.315814.
- 46 Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Inform. Process. Lett.*, 110(20):893–897, 2010. doi:10.1016/j.ipl.2010.07.020.
- 47 Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on boolean functions. In *Proceedings of the 29th Symposium on Foundations of Computer Science (FOCS)*, pages 68–80, 1988. doi:10.1109/SFCS.1988.21923.
- 48 Hartmut Klauck, Robert Špalek, and Ronald de Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. *SIAM J. Comput.*, 36(5):1472–1493, 2007. doi:10.1137/05063235X.
- 49 Adam R. Klivans. On the derandomization of constant depth circuits. In *Proceedings of the 5th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 249–260, 2001. doi:10.1007/3-540-44666-4_28.
- 50 Swastik Kopparty. Lecture 4: AC^0 lower bounds and pseudorandomness. Scribe notes by Jason Perry and Brian Garnett, 2013. URL: <https://sites.math.rutgers.edu/~sk1233/courses/topics-S13/lec4.pdf>.
- 51 Swastik Kopparty and Srikanth Srinivasan. Certifying polynomials for $AC^0[\oplus]$ circuits, with applications to lower bounds and circuit compression. *Theory of Computing*, 14(12):1–24, 2018. doi:10.4086/toc.2018.v014a012.
- 52 Troy Lee and Jérémie Roland. A strong direct product theorem for quantum query complexity. *Comput. Complexity*, 22(2):429–462, 2013. doi:10.1007/s00037-013-0066-8.
- 53 L. A. Levin. One way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987. doi:10.1007/BF02579323.
- 54 Fu Li and David Zuckerman. Improved extractors for recognizable and algebraic sources. In *Proceedings of the 23rd International Conference on Randomization and Computation (RANDOM)*, pages 72:1–72:22, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.72.
- 55 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993. doi:10.1145/174130.174138.
- 56 Noam Nisan, Steven Rudich, and Michael Saks. Products and help bits in decision trees. *SIAM J. Comput.*, 28(3):1035–1050, 1999. doi:10.1137/S0097539795282444.
- 57 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.
- 58 Ryan O’Donnell and Karl Wimmer. Approximation by DNF: examples and counterexamples. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 195–206, 2007. doi:10.1007/978-3-540-73420-8_19.
- 59 Alexander Razborov. A simple proof of Bazzi’s theorem. *ACM Transactions on Computation Theory*, 1(1), 2009. doi:10.1145/1490270.1490273.

- 60 Alexander A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Science of the USSR*, 41(4):333–338, 1987. doi:10.1007/BF01137685.
- 61 Ronen Shaltiel. Towards proving strong direct product theorems. *Comput. Complexity*, 12(1-2):1–22, 2003. doi:10.1007/s00037-003-0175-x.
- 62 Ronen Shaltiel. Is it possible to improve Yao’s XOR lemma using reductions that exploit the efficiency of their oracle? *Comput. Complexity*, 32(1):Paper No. 5, 47, 2023. doi:10.1007/s00037-023-00238-9.
- 63 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010. doi:10.1137/080735096.
- 64 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.
- 65 Michael Sipser. Borel sets and circuit complexity. In *Proceedings of the 15th Symposium on Theory of Computing*, pages 61–69, 1983. doi:10.1145/800061.808733.
- 66 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Symposium on Theory of Computing (STOC)*, pages 77–82, 1987. doi:10.1145/28395.28404.
- 67 Roman Smolensky. On representations by low-degree polynomials. In *Proceedings of 34th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 130–138, 1993. doi:10.1109/SFCS.1993.366874.
- 68 Robert Špalek. The multiplicative quantum adversary. In *Proceedings of the 23rd Conference on Computational Complexity (CCC)*, pages 237–248, 2008. doi:10.1109/CCC.2008.9.
- 69 Avishay Tal. Tight bounds on the fourier spectrum of AC^0 . In *Proceedings of the 32nd Computational Complexity Conference (CCC)*, pages 15:1–15:31, 2017. doi:10.4230/LIPIcs.CCC.2017.15.
- 70 Jun Tarui. Probabilistic polynomials, AC^0 functions and the polynomial-time hierarchy. *Theoretical Computer Science*, 113(1):167–183, 1993. doi:10.1016/0304-3975(93)90214-E.
- 71 Roei Tell. On implications of better sub-exponential lower bounds for AC^0 . <https://sites.google.com/site/roeitell/Expositions>, 2020.
- 72 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. doi:10.1137/0220053.
- 73 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *Proceedings of the 6th Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 162–176, 1977. doi:10.1007/3-540-08353-7_135.
- 74 Emanuele Viola. *The complexity of hardness amplification and derandomization*. PhD thesis, Harvard University, 2006.
- 75 Emanuele Viola. On the power of small-depth computation. *Found. Trends Theor. Comput. Sci.*, 5(1):1–72, 2009. doi:10.1561/04000000033.
- 76 Emanuele Viola. Selected challenges in computational lower bounds. *SIGACT News*, 48(1):39–45, March 2017. doi:10.1145/3061640.3061648.
- 77 Emanuele Viola. New lower bounds for probabilistic degree and ac^0 with parity gates. <https://eccc.weizmann.ac.il/report/2020/015/>, 2020.
- 78 Ryan Williams. Nonuniform acc circuit lower bounds. *J. ACM*, 61(1), January 2014. doi:10.1145/2559903.
- 79 Andrew C. Yao. Theory and application of trapdoor functions. In *Proceedings of the 23rd Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982. doi:10.1109/SFCS.1982.45.
- 80 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *26th Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 1985. doi:10.1109/SFCS.1985.49.
- 81 Andrew Chi-Chih Yao. On ACC and threshold circuits. In *Proceedings of the 31st Symposium on Foundations of Computer Science (FOCS)*, pages 619–627, 1990. doi:10.1109/SFCS.1990.89583.

Streaming Zero-Knowledge Proofs

Graham Cormode   

University of Warwick, UK

Marcel Dall’Agnol   

Princeton University, NJ, USA

Tom Gur   

University of Cambridge, UK

Chris Hickey 

University of Manchester, UK

Abstract

Streaming interactive proofs (SIPs) enable a space-bounded algorithm with one-pass access to a massive stream of data to verify a computation that requires large space, by communicating with a powerful but untrusted prover.

This work initiates the study of *zero-knowledge* proofs for data streams. We define the notion of zero-knowledge in the streaming setting and construct zero-knowledge SIPs for the two main algorithmic building blocks in the streaming interactive proofs literature: the *sumcheck* and *polynomial evaluation* protocols. To the best of our knowledge *all* known streaming interactive proofs are based on either of these tools, and indeed, this allows us to obtain zero-knowledge SIPs for central streaming problems such as index, point and range queries, median, frequency moments, and inner product.

Our protocols are efficient in terms of time and space, as well as communication: the verifier algorithm’s space complexity is $\text{polylog}(n)$ and, after a non-interactive setup that uses a random string of near-linear length, the remaining parameters are $n^{o(1)}$.

En route, we develop an algorithmic toolkit for designing zero-knowledge data stream protocols, consisting of an *algebraic streaming commitment protocol* and a *temporal commitment protocol*. Our analyses rely on delicate algebraic and information-theoretic arguments and reductions from average-case communication complexity.

2012 ACM Subject Classification Theory of computation \rightarrow Interactive proof systems; Theory of computation \rightarrow Streaming, sublinear and near linear time algorithms

Keywords and phrases Zero-knowledge proofs, streaming algorithms, computational complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.2

Related Version *Full Version*: <https://arxiv.org/abs/2301.02161>

Funding *Graham Cormode*: The work of Graham Cormode and Chris Hickey was supported by European Research Council grant ERC-2014-CoG 647557.

Tom Gur: Tom Gur is supported by UKRI Future Leaders Fellowship MR/S031545/1 and EPSRC New Horizons Grant EP/X018180/1.

Acknowledgements We thank Aditya Prakash for the proof of Claim 30, as well as Justin Thaler and Nick Spooner for fruitful discussions and careful reading of an earlier version of this manuscript.

1 Introduction

The design and analysis of algorithms in the *streaming model* is an exceptionally active area of research, particularly so in recent years (see, e.g., the surveys [46, 45, 20] and references therein). A streaming algorithm A observes a long data stream $x = (x_1, \dots, x_n)$, whose size far exceeds A ’s limited memory, one symbol at a time, and computes some pre-specified



© Graham Cormode, Marcel Dall’Agnol, Tom Gur, and Chris Hickey;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 2; pp. 2:1–2:66

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



information about x (e.g., statistics such as the number of distinct elements). To successfully do so, A maintains a summary of the stream that is both small and easily updateable. Algorithmic techniques and data structures that work under such constraints underpin both theoretical progress and real-world deployment of algorithms for massive datasets.

However, it has long been known that many natural and important problems are hard in the streaming model [2]. This motivates the study of protocols for *delegation of computation*, whereby a streaming algorithm offloads expensive operations to an untrusted party with large memory, but can still verify (in low space) that the purported result is correct. Accordingly, *interactive proofs* in the data stream model received a great deal of attention in the last decade [24, 22, 23, 54, 11, 55, 14, 26, 21, 16, 17].

Streaming interactive proofs (SIPs) are delegation-of-computation protocols where the computationally bounded party is bounded *not* in its time complexity, but rather in space and input access. More precisely, an SIP is an interactive protocol between a powerful but untrusted prover P and a space-bounded streaming verifier V which has sequential, one-pass access to a massive input as well as the prover’s messages. We note that prover and verifier observe the same stream of bits, which only the former can store in its entirety.

Remarkably, SIPs allow low-space streaming algorithms to efficiently verify key problems in the data stream model that are completely intractable without the assistance of a prover. Indeed, the aforementioned sequence of works constructed SIPs with polylogarithmic-space verifiers for a large collection of problems, many of which require linear space for a streaming algorithm alone (such as the INDEX and frequency moment problems). The underlying power that enables exponential separations between streaming algorithms and SIPs essentially boils down to two powerful protocols: *sumcheck* and *polynomial evaluation*, which can in turn be applied to a plethora of problems.

Determining the extent to which SIPs can be augmented with extra features is the natural next step to a refined understanding of the complexity landscape around them. Our work focuses on *zero-knowledge*: ensuring that the protocol reveals no information besides what it is designed to compute. We remark that this feature widens the array of computational tasks solvable by mutually distrusting parties, thus supporting numerous cryptographic protocols currently in use [6, 4, 7].

Despite the fundamental role of zero-knowledge in theoretical computer science (see, e.g., [57, 30, 56, 31] and references therein) and the extensive study of SIPs over the last decade, no zero-knowledge SIPs were known prior to this work. Indeed, it is not obvious a priori whether they are at all possible: for instance, while traditional zero-knowledge prevents leakage of information to a polynomial-time adversary about some hard computation on an input x (e.g., a witness that certifies x is in a language), in the streaming setting a space-bounded verifier must learn no additional information *about x itself* – even if its runtime is unbounded.

1.1 Zero-knowledge in the streaming model

Recall that in the traditional setting, which deals with *polynomial-time* algorithms, a protocol is zero-knowledge if, for every (possibly malicious) verifier \tilde{V} , there exists a simulator $S_{\tilde{V}}$ whose output cannot be told apart (either computationally or statistically) from a real interaction between P and \tilde{V} by any distinguisher D ; and if this holds up to negligibly small error, the protocol can be safely repeated or composed.

In the streaming model, algorithms are restricted to *one-pass sequential access* to their input and the primary resource is *space*, rather than time. Accordingly, we say that an SIP is zero-knowledge if \tilde{V} , S and D are streaming algorithms; when \tilde{V} has s bits of memory, the simulator has roughly s space and we allow the distinguisher D to have an arbitrary $\text{poly}(s)$ amount of memory. (See Section 4 for formal definitions.) Albeit similar, this notion is distinct to its poly-time analogue in two fundamental ways.

Negligible distinguishing bias is a robust notion of security in the setting of polynomial-time computation because it prevents polynomial-time adversaries from boosting their advantage by repeating (polynomially) many executions. However, in the data stream model, the *one-pass* restriction on input access precludes this strategy altogether; indeed, streaming problems often become trivial with a single additional pass. We therefore define secure protocols as those achieving $o(1)$ distinguishing bias, which ensures that the probability of information leakage tends to zero. (See Remark 14 for a more detailed discussion of alternative “hybrid” models and security bounds.)

The second crucial distinction is that the notion of zero-knowledge for SIPs is *unconditional*, i.e., does not rely on computational assumptions, faithfully to the nature of the data stream model. This differs markedly from past work on zero-knowledge protocols where the verifier is able to process incoming messages in a streaming fashion (e.g., [34, 22]), whose zero-knowledge property is still with respect to the standard setting: while the honest verifier is a streaming algorithm, the protocols are only secure against polynomial-time adversaries. In this work, *adversaries are also streaming algorithms*.

This paper explores the extent to which zero knowledge streaming interactive proofs (zkSIPs) can outperform streaming algorithms: does there exist a problem they solve more efficiently? If so, can they do so for a natural problem such as INDEX, or even more ambitiously, achieve an exponential reduction in the space complexity for key problems in the data stream model?

1.2 Main results

Our main contribution is a strong positive answer to the questions above, providing the tools to construct zero-knowledge streaming interactive proofs for essentially any problem within the reach of current (non-zero-knowledge) SIPs.

In more detail, our main results are zero-knowledge versions of the two building blocks underlying all known SIPs: the *sumcheck* and *polynomial evaluation* protocols, from which we derive zkSIPs for central streaming problems in Section 1.3. In doing so, we obtain *exponentially* smaller space complexity for the fundamental INDEX and frequency moment problems (among others) when compared to streaming algorithms alone.

We remark that all our zkSIPs are two-stage protocols with a *setup* and an *interactive* stage. The setup is non-interactive and consists merely of a random string (see Section 2.3), which can be reused in multiple interactive executions (of possibly different protocols).¹ With this simple preprocessing step, we achieve essentially optimal time and communication complexities (i.e., subpolynomial or even polylogarithmic – as do the best non-zero-knowledge SIPs – and dramatically smaller than the complexity of streaming the input) in the interactive stage.

Sumcheck Zero-Knowledge SIP

In the SUMCHECK problem, the goal is to compute the sum of evaluations of a low-degree polynomial over a large structured set (a subcube). Protocols for SUMCHECK are some of the most important building blocks for interactive proofs, and are extremely useful for SIPs in particular.

¹ We also remark that omitting the setup yields an honest-verifier (but not malicious-verifier) zkSIP with $n^{o(1)}$ communication complexity.

We state the following theorem in generality, but note that standard parameter settings imply space complexity $s = \text{polylog}(n)$ as well as $O(n^{1+\delta})$ (for any constant $\delta > 0$) and $n^{o(1)}$ communication in the setup and interactive stages, respectively. (The time complexity is of the same order as the communication in both stages.) This is the case in all of our applications.

► **Theorem 1** (Theorems 47 and 48, informally stated). *There exists a zkSIP for SUMCHECK where, for m -variate low-degree polynomials over \mathbb{F} , the verifier uses $s = O(m^2 \log |\mathbb{F}|)$ bits of space. The SIP communicates $\tilde{O}(|\mathbb{F}|^m)$ bits in its setup and $|\mathbb{F}|^{\log \log |\mathbb{F}| + O(1)}$ bits in the interactive stage.*

The round complexity (the number of messages sent or received by each party throughout the SIP) is $m + O(1)$, a small constant larger than that of the standard sumcheck protocol.

We stress that while sumcheck is traditionally used (in the polynomial-time setting) to verify exponentially large sums in polynomial time, this is *not* the goal of the streaming variant, as sums of evaluations over a large set can be obtained incrementally for functions computable in low space (a class that includes polynomials).

Nevertheless, the sumcheck protocol achieves exponential savings in space complexity for problems that require large space without interaction: it enables efficient verification of sums of polynomials implicitly defined input defines implicitly, which require *linear space* to compute otherwise.

Polynomial Evaluation Zero-Knowledge SIP

We proceed to our second main result: a zero-knowledge SIP for the *polynomial evaluation problem* PEP, which consists of computing a low-degree polynomial at a single point (revealed after the description of the polynomial). It allows a streaming algorithm to recover data that was seen but not stored, by saving a small *fingerprint* of the stream. Similarly to sumcheck, general-purpose PEP protocols are widely applicable to the design of SIPs.

► **Theorem 2** (Theorems 35 and 36, informally stated). *There exists a zkSIP for PEP where, for m -variate low-degree polynomials over \mathbb{F} , the verifier uses $O(m \log |\mathbb{F}|)$ bits of space. The communication complexity is $\tilde{O}(|\mathbb{F}|^m)$ in the setup and $\text{poly}(|\mathbb{F}|)$ bits in the interactive stage.*

As in Theorem 1, standard parameter settings imply zkSIPs with polylogarithmic space, $n^{o(1)}$ time and communication complexity (in the interactive stage)² as well as near-linear communication in the setup. The round complexity is $O(1)$.

1.2.1 Streaming commitment protocols

En route to proving Theorems 1 and 2, we construct tools for the design of zkSIPs which we find of independent interest. Namely, we provide two types of *commitment protocols* for streaming algorithms.

We remark that in the polynomial-time setting, the existence of secure commitment schemes is equivalent to the existence of one-way functions [43, 47, 42], so it may seem surprising that our results hold *unconditionally*. However, in the incomparable model of streaming algorithms, which are not time-bounded, but are instead severely constrained with respect to space and input access, we show that no cryptographic assumption is needed.³

² A nontrivial security guarantee still holds with $\text{polylog}(n)$ communication, but with $n^{o(1)}$ the protocol becomes secure against arbitrary $\text{polylog}(n)$ -space adversaries; see Remark 38.

³ We refer to commitment *protocols* rather than schemes in the streaming model to avoid ambiguity with the polynomial-time analogue; see Definition 17.

Streaming algebraic commitment protocol

The following result shows that not only does a streaming commitment protocol exist, but that it can be made *linear*; that is, the sender may commit to a sequence of messages and decommit to a linear combination thereof, with linear coefficients of the receiver’s choosing.

► **Theorem 3** (Theorem 24, informally stated). *There exists a commitment protocol whereby an unbounded-space sender commits a tuple $\alpha \in \mathbb{F}^\ell$ to a streaming receiver and decommits to a linear combination $\alpha \cdot \beta$, with linear coefficients β chosen by the receiver. The receiver’s space complexity is $O(\ell \log |\mathbb{F}|)$ and the protocol communicates $\tilde{O}(|\mathbb{F}|^{3\ell})$ bits.*

Temporal commitment protocol

The second component is a new notion of a streaming commitment, which we call *temporal*. This protocol allows a streaming *verifier* to “timestamp” its message, providing evidence that it was chosen before streaming a particular input.

► **Theorem 4** (Theorem 33, informally stated). *Let Γ be an alphabet and A a space- s streaming algorithm with $s = \text{polylog} |\Gamma|$. If A streams $z \sim \Gamma^v$ and v is large enough, the following holds: independently of its computation after z , with high probability A can output at most s symbol-certificate pairs $(\alpha, i) \in \Gamma \times [v]$ such that $\alpha = z_i$.*

In other words, $A(z)$ cannot remember more than s symbol-certificate pairs for the string z ; and the bound is unchanged if A obtains information uncorrelated with z after reading the stream.

1.3 Applications

Recall that Theorems 1 and 2 provide zero-knowledge versions of the general tools that essentially underlie all known SIPs, namely, the *sumcheck* and *polynomial evaluation* protocols. We demonstrate the power and flexibility of our tools by deriving from them explicit zkSIPs for streaming problems of fundamental importance: INDEX and FREQUENCY-MOMENT, as well as POINT-QUERY, RANGE-COUNT, SELECTION and INNER-PRODUCT.

As mentioned in the previous section, while the following statements highlight space complexities, the communication complexities are $n^{o(1)}$ in the interactive stage and $O(n^{1+\delta})$ for arbitrarily small δ in the setup stage.

In the INDEX problem, a streaming algorithm reads a length- n string x followed by an index $j \in [n]$, and its goal is to output x_j . INDEX is a hard problem for streaming algorithms, requiring *linear* space to solve [49]. By instantiating our zkSIP for polynomial evaluation with respect to the low-degree extension of the input evaluated at the index j , we obtain the following.

► **Corollary 5** (Corollary 39, informally stated). *There exists a zkSIP for INDEX with logarithmic verifier space complexity.*

Note that this matches the space complexity of the non-zero-knowledge SIP of [14, 15].

In the FREQUENCY-MOMENT $_k$ (or F_k) problem, an algorithm streams $x \in [\ell]^n$ and its task is to compute $F_k(x) = \sum_{i \in [\ell]} \varphi_i^k$, the k^{th} moment of the frequency vector $(\varphi_1, \dots, \varphi_\ell)$, where φ_i is the number of occurrences of i in x . This is a central problem in the streaming literature, which is well known to require linear space to compute [2]; by instantiating our sumcheck protocol with respect to the low-degree extension of the frequency vector, we obtain a zero knowledge protocol for the exact computation of F_k .

► **Corollary 6** (Corollary 50, informally stated). *For every $\ell \in [n]$ and k , there exists a zkSIP that computes F_k with $\text{polylog}(n)$ verifier space complexity.*

Lastly, we illustrate the flexibility of our protocols by constructing additional zkSIPs for several other problems: POINT-QUERY (where the input is a stream of integer updates to an ℓ -dimensional vector y followed by an index j and the task is to output y_j); RANGE-COUNT (where the input is a sequence of points in $[\ell]$ followed by a range $R \subseteq [\ell]$ and the task is to output the number of occurrences in R); SELECTION (which generalises the computation of the median); and INNER-PRODUCT (where the task is to output the inner product between the frequency vectors of a pair of streams).

► **Corollary 7** (Corollaries 41, 43, 45, and 52, informally stated). *There exist $\text{polylog}(n)$ -space zkSIPs for POINT-QUERY, RANGE-COUNT, SELECTION and INNER-PRODUCT.*

1.4 Related work

This work builds on the line of research on streaming interactive proofs, initiated by [12, 13] and actively investigated over the last decade [23, 24, 22, 11, 55, 38, 14, 26, 1, 21, 15, 29, 17]. These sublinear interactive proofs are also closely related to proofs of proximity [52, 36, 39, 40, 33, 51, 19, 32, 37, 25].

Indeed, our two main results can be seen as zero-knowledge versions of the main techniques in [15] and [22]: respectively, a polynomial evaluation and a sumcheck protocol. (We note that while [18] construct a zero-knowledge sumcheck protocol via an algebraic commitment scheme, their model and techniques are completely different.)

Past work has studied zero-knowledge protocols where the verifier is able to process incoming messages in a streaming fashion (e.g., [34, 22]), but their zero-knowledge property is with respect to the standard, *polynomial-time*, setting; that is, while the honest verifier is a streaming algorithm, the security of the protocol holds against polynomial-time adversaries, whereas we consider adversaries that are also streaming algorithms.

We note that while unconditional cryptographic primitives such as bit commitments and key agreement are achievable in the *bounded-storage model* (see, e.g., [35] and references therein), the security guarantees are weaker, allowing at most a quadratic, rather than arbitrary polynomial, gap between honest and malicious parties. Recent work on the streaming variant of the model [27, 28] is more closely related to ours. However, they do not construct commitment schemes and, more importantly, these results assume bounds on the space of both parties; therefore, they do not immediately apply to *statistically sound* proof systems such as those considered in this work.

We also note that while zero-knowledge proofs within sublinear models of computation have been actively explored in the last decade (e.g., [8, 44]), our work is the first to do so in the streaming model.

1.5 Open problems

This work opens several avenues for future research; in this short section, we highlight four particularly compelling directions.

Achieving zero-knowledge versions of the main building blocks in the SIP literature suggests a natural question: can *all* SIPs be endowed with zero-knowledge? That is, denoting by SIP (respectively, zkSIP) the class of languages that admit SIPs (respectively, zkSIPs) with $\text{polylog}(n)$ space complexity, we raise the following problem.

► **Open problem 1.** *Is SIP equal to zkSIP?*

In our two-stage protocols, the communication complexity is dominated by the setup (a reusable random string of near-linear length); the remainder of the protocol is extremely efficient, with $n^{o(1)}$ (or even $\text{polylog } n$) communication and time complexity. Making this parameter sublinear would be a major step towards practical applicability.

► **Open problem 2.** *Can zero-knowledge SIPs achieve sublinear communication complexity?*

Lastly, recall that the notion of security in this work is (unconditional and) *computational*, where streaming adversaries detect a simulation with at most $o(1)$ bias. It is natural to ask whether stronger notions are achievable – both with respect to an adversary’s capabilities and feasible security bounds.

► **Open problem 3.** *Are there SIPs with statistical (or even perfect) zero-knowledge?*

► **Open problem 4.** *Can security bounds of $\frac{1}{\text{poly}(n)}$ or $\frac{1}{n^{\omega(1)}}$ be obtained for computational zkSIPs?*

Organisation

The rest of the paper is organised as follows. In Section 2 we give a high-level overview of the challenges and the techniques we use to endow SIPs with zero-knowledge. We briefly discuss the preliminaries for the technical sections in Section 3, and, in Section 4, formally define the notion of streaming zero-knowledge and discuss key conceptual points. In Section 5 we construct the two commitment protocols that comprise the main components for our polynomial evaluation and sumcheck protocols. We construct the protocols, prove their zero-knowledge property and show applications for them in Sections 6 and 7, respectively.

2 Technical overview

We provide a high-level overview of the techniques we use and build upon in this paper. For concreteness, we illustrate our methodology by focusing on the construction of zero knowledge SIPs for one of the most fundamental problems in the data stream model: INDEX.

We begin with a bird’s eye view of our ideas and the challenges that arise in their implementation. The starting point of our efforts is Section 2.1, where we describe the *polynomial evaluation protocol (pep)*, from which a (non zero-knowledge) SIP for the INDEX problem follows. An attempt to make this protocol zero-knowledge faces two fundamental challenges, which we address in Sections 2.2 and 2.3 via the construction of two types of *streaming commitment protocols*.

In Section 2.4, we apply the foregoing protocols to obtain a streaming interactive proof for INDEX and provide an overview of the proof of its zero-knowledge property, which requires an involved simulator argument. Finally, Section 2.5 sketches another application of this framework that obtains an additional powerful and flexible tool: a *zero-knowledge streaming sumcheck* protocol.

2.1 A starting point: the polynomial evaluation protocol

Recall that in the INDEX problem, a streaming algorithm with s bits of memory receives a length- n string x over an alphabet Γ , followed by a coordinate $j \in [n]$, and its goal is to output $x_j \in \Gamma$. It is well-known that INDEX is maximally hard for streaming algorithms, requiring $s = \Omega(n)$ space for the output to be correct with nontrivial probability.

First, note that obtaining an efficient SIP for INDEX is non-trivial even without zero-knowledge. Indeed, the naive approach of having the prover P reveal the index j before V streams x (allowing the verifier to save x_j) fails: both parties observe *the same* stream of information, so P only learns j long after V has seen x_j . Any communication in an SIP before the input stream must therefore be *independent* of it.

Remarkably, an exponential reduction in space complexity is possible despite both prover and verifier not knowing the index j before it appears in the stream. We recall the SIP in [15], upon which we build, and argue why it is *not* zero-knowledge to begin with. Their SIP is an application of `pep`, the *polynomial evaluation protocol*, which enables a small-space algorithm to recover any element that was streamed but not stored, using only a small fingerprint of the stream.

We embed the input stream into an object with algebraic structure in a space of size much larger than n , namely, by viewing $x_i \in \mathbb{F}$, for a large enough finite field \mathbb{F} , and considering an m -variate *low-degree polynomial* \hat{x} that interpolates across all x_i ; we call the polynomial $\hat{x} : \mathbb{F}^m \rightarrow \mathbb{F}$ of individual degree $d = d(m, n)$ the low-degree extension (LDE) of x . (Usual parameter settings satisfy $d, m \leq \log n$ and $|\mathbb{F}| = \text{polylog}(n)$.)

The protocol proceeds as follows. The verifier samples a random evaluation point $\rho \sim \mathbb{F}^m$ and computes the *fingerprint* $\hat{x}(\rho)$, which can be evaluated in low space via standard online Lagrange interpolation. After V learns j , it enlists P in the recovery of x_j : it sends P a line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ incident to j (viewing this index as an element of \mathbb{F}^m) and ρ , where $L(0) = j$ and $L(\rho) = \rho$ for a random $\rho \sim \mathbb{F}$, whereupon P replies with the (low-degree) univariate polynomial $\hat{x}_{|L} = \hat{x} \circ L$.

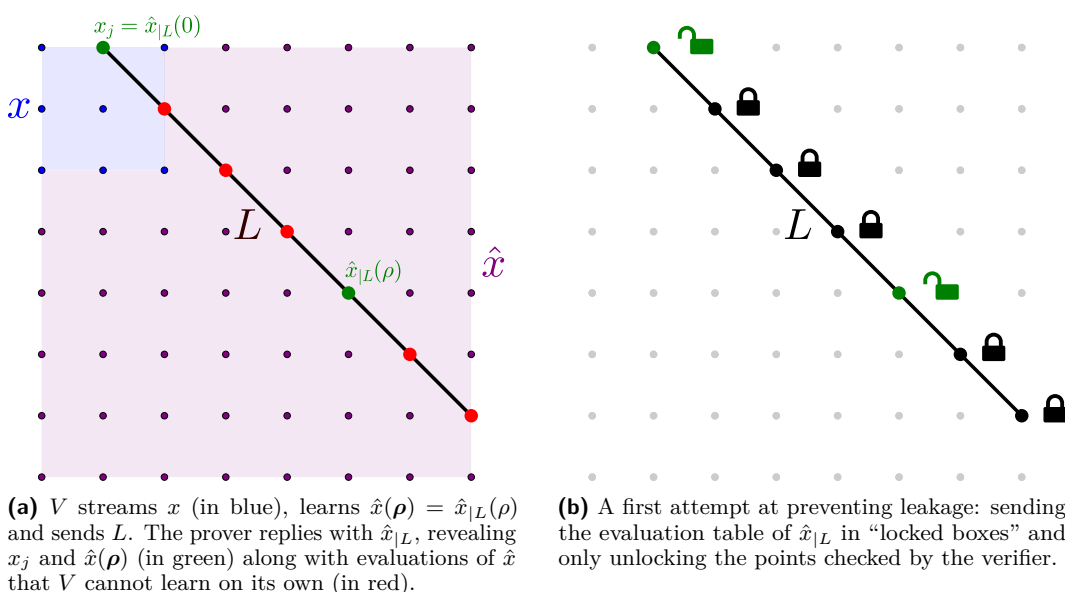
If P is honest, then V can easily recover $x_j = \hat{x}(j) = \hat{x}_{|L}(0)$. However, P could easily cheat if V made no further checks: the prover could just as well pick $\alpha \in \mathbb{F}$ arbitrarily and send any low-degree polynomial g such that $g(0) = \alpha$ to (falsely) convince V that $x_j = \alpha$. By having V only accept the prover's claim that $x_j = g(0)$ if g also agrees with the fingerprint, i.e., if $g(\rho) = \hat{x}_{|L}(\rho) = \hat{x}(\rho)$, the verifier thwarts this (and any other) attack: since both ρ and ρ are unknown to the prover, to convince the verifier of an incorrect answer $g(0) \neq \hat{x}_{|L}(0)$, the prover must send a polynomial $g \neq \hat{x}_{|L}$ that agrees with $\hat{x}_{|L}$ at a random point; and if \mathbb{F} is sufficiently large, the probability of this event (ρ being a root of the nonzero polynomial $g - \hat{x}_{|L}$) is arbitrarily small.

The protocol outlined above is, however, *not* zero-knowledge: after all, V learns not only x_j , but the restriction of \hat{x} to an entire line L through j (see Figure 1a). Note that learning the restriction of \hat{x} to (say) a random line R does not necessarily constitute leakage: V could simply compute a few evaluations (rather than only one) of $\hat{x}_{|R}$, which fully determine the polynomial. The issue is that L is a function of the coordinate j , which V does not know prior to streaming x .

In the next section we will take our first steps towards making the protocol zero-knowledge, i.e., ensuring that the verifier learns nothing beyond the value x_j . Note that the honest V only evaluates $\hat{x}_{|L}$ at two points, ρ and 0; what if P could send the evaluations of $\hat{x}_{|L}$ in “locked boxes” and only open the pair that the verifier needs?

2.2 Curtailing leakage with commitments

To make the foregoing approach more precise, let us first assume the existence of a *commitment protocol* that allows P to transmit any field element α to V in two steps: sending a string `commit`(α), from which V is unable to extract any information about α ; and later, upon the verifier's request, revealing a field element β such that, if $\beta \neq \alpha$, then V can detect that the P is being dishonest.



■ **Figure 1** Leakage in the SIP for INDEX via evaluation of the bivariate polynomial $\hat{x} : \mathbb{F}^2 \rightarrow \mathbb{F}$, and an (unsuccessful) attempt to prevent it.

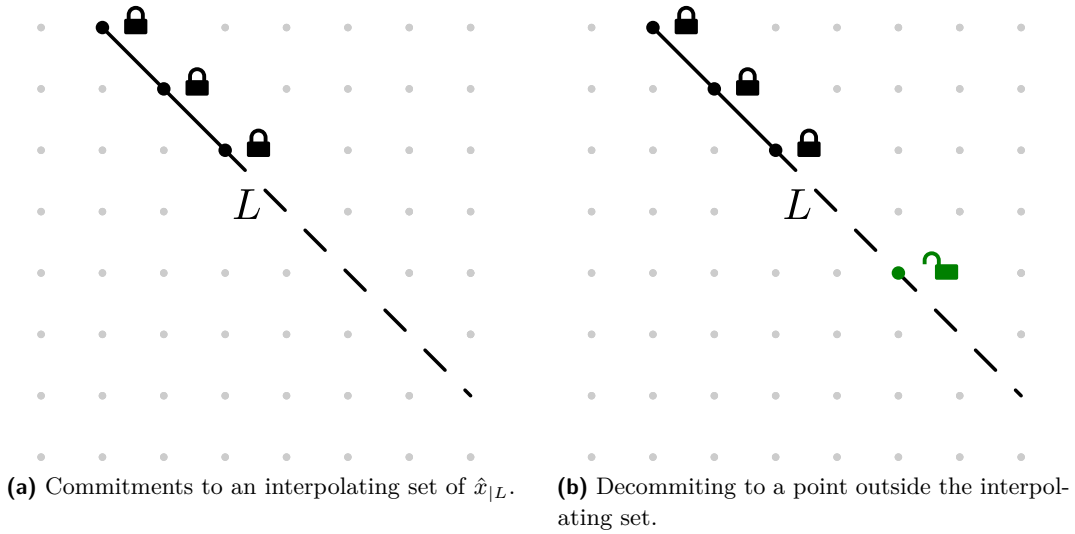
With such a commitment protocol in hand, a natural attempt to prevent the `pep` protocol from leaking information is to have the prover P send a commitment to $\hat{x}_{|L}$, the restriction of the input’s LDE to the line chosen by V (rather than sending the polynomial in the clear). That is, the prover would commit to the evaluation table of $\hat{x}_{|L}$, sending $(\text{commit}(\hat{x}_{|L}(\rho')) : \rho' \in \mathbb{F})$, after which V can reveal its random evaluation point ρ and P decommits *only* to the evaluations of 0 and ρ (see Figure 1b). This does indeed reveal less information (2 rather than $|\mathbb{F}|$ evaluations of \hat{x}), but is still far from what we set out for.

There are two severe shortcomings with this idea; we shall tackle one now and defer the other to Section 2.3. First we need to ask: what is to prevent a cheating prover from committing to a function g that is inconsistent with $\hat{x}_{|L}$? Indeed, since V is (by design) unable to learn the field elements that were committed to, it cannot detect whether the function is a low-degree polynomial; then a cheating prover may commit to any $\alpha \neq x_j = \hat{x}_{|L}(0)$ as the claimed evaluation at 0, while committing to the correct evaluations elsewhere. The resulting function is not a low-degree polynomial anymore, but V is oblivious to this fact.

Therefore, we require a scheme that allows not only to commit to a function, but to also ensure it is a low-degree polynomial. We solve this problem by constructing an *algebraic* commitment protocol, whereby P commits to a set of field elements and can decommit to *any linear combination* of them. Then P may commit to $d + 1$ points – which uniquely determine a degree- d polynomial g – and V requests a decommitment to the linear combination that coincides with $g(\rho)$ (see Figure 2). We next present the basic commitment protocol, and then extend it to be algebraic.

The basic protocol

Recall that our goal is to construct a commitment protocol between asymmetric parties, allowing a computationally unbounded P to send and later reveal a message $\alpha \in \mathbb{F}$ to a low-space verifier V . We focus on the first step, where P sends a hidden message, and deal with how to reveal it later. A natural attempt is to play the prover’s strength against the



■ **Figure 2** Preventing leakage by committing to $\hat{x}_{|L}$ as an interpolating set for the polynomial. To decommit to an evaluation outside the set, the scheme must be algebraic.

verifier’s weakness: we know, from the hardness of INDEX, that the space limitation of V prevents it from recalling an item from a long stream whose position is only revealed later; we can thus have P send a long stream y with the message hidden at a coordinate k that is revealed at the end.

While the idea seems intuitively sound, there are nontrivial issues to address. For example, the string-coordinate pair (y, k) should not have any structure from which V could extract information, which we can ensure by sampling both uniformly at random; but to prove security for this strategy, INDEX must be hard to solve *on average*. Luckily, reductions from one-way communication complexity enable us to prove this fact: one-way protocols where Alice receives $x \sim \{0, 1\}^n$ and sends an s -bit message to Bob, who receives $j \sim [n]$ and attempts to output x_j , succeed with probability at most $\frac{1}{2} + O(\sqrt{s/n})$ [49]. We show that the bound extends to larger alphabets, carrying over to space- s streaming algorithms (see Proposition 21 and Lemma 25).

In short, we have P encode its message $\alpha \in \mathbb{F}$ as the solution to a random INDEX instance, exploiting the problem’s average-case hardness to ensure that V is unable to extract α ; more precisely, P sends a uniformly random string-coordinate pair (y, k) and then the “correction” $\gamma = \alpha - y_k$.⁴ Of course, the discussion thus far only shows how P can commit; but we also need a decommitment protocol whereby V can check that P is being honest when it reveals β (which may or may not coincide with the message α). Fortunately, we already have a tool V can use to solve INDEX with an untrusted prover’s assistance! The decommitment thus consists of an execution of `pep` by P and V with respect to the instance (y, k) : this allows V to learn y_k and check that $\gamma + y_k = \beta$, i.e., that the correction γ sent earlier matches the (alleged) message.

Recall that we are building technical tools towards a zkSIP for INDEX, so we ultimately *exploit the hardness of a problem to solve an instance of the same problem*. Should we not expect, then, that the same leakage issues should arise with respect to the “virtual” instance

⁴ We remark that while replacing y_{ik} with α (rather than sending a random element and a correction later) looks simpler, then (y, k) ceases to be a random INDEX instance, and it is not clear how to show a reduction from INDEX.

(y, k) as they did with the “real” instance (x, j) ? While this may appear to be circular reasoning, we stress that revealing evaluations of \hat{y} leaks no information whatsoever about the input; indeed, (y, k) is a uniform random variable that is independent of (x, j) . Put differently, V only obtains information about uniformly random strings that are completely uncorrelated with the input. See Section 5.2 for details.

Making the scheme algebraic

We now extend the foregoing idea into an *algebraic* protocol, which allows P to commit to a *tuple* of field elements $\alpha = (\alpha_1, \dots, \alpha_\ell)$ and decommit to a linear combination $\alpha \cdot \beta$. (Committing to a polynomial and decommitting to an evaluation follows as a special case; see Section 5.1.) Note that such an extension seems to follow if linear combinations “commute” with commitments; that is, by showing that linear combinations of a fingerprint (as defined in Section 2.1) match a fingerprint of the linear combinations, we should be able to use essentially the same strategy of the basic scheme: committing with a random INDEX instance and decommitting with `pep`. Details follow.

Consider a trivial extension of the scheme that allows P to transmit a pair of messages $\alpha, \alpha' \in \mathbb{F}$: sending two independent commitments $(y, k, \alpha - y_k)$ and $(y', k', \alpha' - y'_k)$. The key observation is that, if V saves two fingerprints *at the same evaluation point* ρ , then linear combinations and low-degree extensions do commute: for any $\beta, \beta' \in \mathbb{F}$, defining $z := \beta y + \beta' y'$, we have $\hat{z}(\rho) = \beta \hat{y}(\rho) + \beta' \hat{y}'(\rho)$; in short, evaluating a low-degree extension is a linear operation.

A problem still remains, however: since $k \neq k'$ with overwhelming probability, an execution of the `pep` protocol enables V to learn $z_k = \beta y_k + \beta' y'_k$; but the correction for y' refers to another coordinate $k' \neq k$ (with overwhelming probability). We address this issue by *hiding both messages at the same index*, i.e., setting $k' = k$ and only revealing the coordinate after both y and y' are sent; see Section 5.3 for details.

2.3 From honest to malicious verifiers: temporal commitments

Recall that a source of leakage in the INDEX protocol of Section 2.1 is the prover P sending the restriction of \hat{x} (the LDE of the input) to a line L in the clear. In the previous section, we constructed a prover-to-verifier scheme that enables P to commit to a low-degree polynomial and decommit to a single evaluation of it. We may then use it to modify the original protocol, having P instead *commit* to $\hat{x}|_L$ and decommit to the points inspected by V .

While this modification amounts to significant progress – indeed, it achieves an *honest-verifier* SIP for INDEX – there is a second major challenge to address. The issue is that *if a verifier \tilde{V} cheats*, it can use the protocol to extract information that it could not have learned on its own, as we will see next. The goal of this section is to describe a strategy that prevents leakage of information *without* requiring that \tilde{V} behave honestly; in other words, we would like to make the protocol *malicious-verifier zero-knowledge*.

Concretely, consider the (cheating) verifier \tilde{V} that ignores the input string x , reads j and requests the line through j and $j + 1$ from the prover. P then commits to the restriction of \hat{x} to this line and decommits to the evaluation of the LDE at both j and $j + 1$. This reveals x_j and x_{j+1} to \tilde{V} , which shows clearly that the modified protocol still leaks: x_j is *the only information the verifier should learn* that it could not have computed on its own, but the protocol also reveals x_{j+1} (which is just as hard to compute as the j^{th} coordinate).

An idealised scenario: V -to- P commitments

Let us assume, for the moment, that there also exists a commitment protocol in the reverse direction, allowing V to commit and later reveal a message to P . We will show how, in this idealised setting, we can prevent information leakage altogether. Note that the difficulty posed by a malicious verifier \tilde{V} is the usage of an allegedly random evaluation point ρ that is, in reality, a function of the input.

If \tilde{V} proves that ρ is indeed random, however, we may conclude that \tilde{V} could have computed $\hat{x}(\rho)$ alone – and thus that no leakage occurs. The idealised scheme allows \tilde{V} to do (almost) that, by having it commit to ρ before reading the input stream and decommit to it at a later step (after the prover’s commitment). While this does not ensure ρ is random, the fact that \tilde{V} cannot decommit to anything other than ρ constrains its evaluation point to be chosen before the input stream, so that it cannot be a function of the input.

Of course, it is not at all clear that such a commitment protocol, allowing a weak computational party to commit to a computationally unbounded one, even exists; after all, the commitment step generally exploits their very difference to hide the message, as we did in the previous section. Is this just wishful thinking?

The solution: a temporal commitment

We will now see that, perhaps surprisingly, we can once again exploit the space limitation of \tilde{V} to accomplish this goal. What we obtain in fact falls short of a full-fledged commitment protocol: roughly speaking, the *temporal commitment* will enable a space- s verifier \tilde{V} to reveal not one, but s messages. But this collection is still determined before the input, so that it remains fit for purpose (incurring a small overhead in the simulator algorithm that we discuss in the following section).

As discussed above, we cannot expect \tilde{V} to be able to send a hidden message to P : however \tilde{V} may try to hide it, P can simply store the entirety of the communication and extract the message itself. Since sending is out of the picture, could \tilde{V} instead commit by receiving a message? Note that, while somewhat counter-intuitive, this would allow \tilde{V} to play what is essentially its only strength, its private randomness, against P . Recall, moreover, that there is a temporal aspect to the positions of a long stream z that \tilde{V} can remember: if it remembers z_i , this can be seen as evidence that i was determined no later than when z was seen.

Let us now make the idea more precise, and construct our verifier-to-prover temporal commitment protocol. The main idea is to impose some cost onto the ability of \tilde{V} to “unlock” the decommitment from P , without overly constraining the honest verifier V . Note that after P sends the commitment to a low-degree polynomial, having V reveal the point $\rho = L(\rho)$ at which it computed \hat{x} is not a problem (as opposed to revealing ρ before P sends the polynomial, which allows the prover to cheat easily). Therefore, we will have \tilde{V} reveal its alleged evaluation point ρ along with a certificate $c(\rho)$ that shows \tilde{V} selected the point before seeing the input stream. P will only proceed with the protocol if the certificate is valid; if not, it aborts to prevent \tilde{V} from learning information beyond its reach.

Given that the verifier’s scarce resource is space, we design this certificate to require a number of bits that is not too large and yet not negligible; then the honest V should have no trouble, as it only needs to remember one piece of information, whereas the malicious \tilde{V} described before would need to store a certificate for the evaluation point $j + 1$, which it does not know before reading x .

We thus prepend our INDEX protocol with a step where P sends \tilde{V} a long string z containing all possible evaluation points (i.e., the entire domain) of the low-degree extension \hat{x} .⁵ Now, if \tilde{V} wants the prover, in the future, to decommit to a polynomial evaluation at the point ρ , it must offer evidence that ρ is uncorrelated with the input stream: \tilde{V} does so by revealing ρ along with the coordinate i that contains ρ in z ; i.e., the certificate for ρ is $c(\rho) = i$, the coordinate satisfying $z_i = \rho$.

The temporal commitment indeed achieves what we set out for: regardless of what \tilde{V} does, as long as its space is bounded we are able to extract the points it may ask P for *in advance of its streaming of x* (see Section 5.4). Note that the commitment is non-interactive (consisting of a single message from P to V) and need not be rerun if the verifier streams multiple inputs; we shall use it as the setup stage of our protocol. Its analysis is subtle and involved: it begins with a study of a variant of INDEX in the one-way communication model that we call RECONSTRUCT, where, upon receipt of a message from Alice, Bob outputs a guess for every coordinate of the input string rather than for only one. Using tools from information theory, we obtain an upper bound on the expected number of correct coordinates, which we call the protocol’s *score*.

Next, we use the expected score bound of RECONSTRUCT to prove a related upper bound for a problem we call PAIR: a variant of INDEX where Bob, rather than receiving the coordinate to be recovered as part of the input, is free to choose it. The implication is that any protocol for PAIR has a small number C of indices such that the output of the protocol is outside C and yet correct (i.e., a pair (i, z_i) with $i \notin C$) with arbitrarily small probability. This will underpin the *simulator argument* that ultimately shows our protocol is zero-knowledge, which we sketch in the next section.

2.4 A sketch of the zero-knowledge INDEX protocol

We now have all of the components necessary to sketch a zero-knowledge streaming interactive proof for INDEX. Recall that we constructed a prover-to-verifier *algebraic* commitment protocol in Section 2.2 and a verifier-to-prover *temporal* commitment in Section 2.3. We will now compose them in the appropriate order, using the temporal commitment to constrain V to choose its inner randomness before reading the input stream; and the algebraic commitment to ensure P only reveals what the verifier needs. The protocol follows.

Parameters

Without loss of generality, we consider the alphabet over which the input string is defined to be a field of size $|\mathbb{F}| = q$; that is, $x \in \mathbb{F}^n$. We also fix two additional parameters, d and m , which characterise the low-degree extension $\hat{x} : \mathbb{F}^m \rightarrow \mathbb{F}$ as an m -variate polynomial of individual degree d . We assume all parameters are known to P and V in advance.

Setup: verifier-to-prover temporal commitment

P sends V a permutation of \mathbb{F}^m as a string z (of length $v = q^m$). Before receiving the string, V samples $\rho \sim \mathbb{F}^m$ and then streams z . When it sees ρ at the ℓ^{th} coordinate of z , the verifier stores ℓ .

⁵ In fact, any given point has a small probability of being absent from the string. We ignore this issue in the technical overview.

Step 1: input streaming

V streams the input string x and records the fingerprint $\hat{x}(\rho)$ as well as the target index j .

Step 2: prover-to-verifier algebraic commitment

V samples $\rho \sim \mathbb{F}$ and sends P the line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ through j and ρ (satisfying $L(0) = j$ and $L(\rho) = \rho$).

P sends $x_j = \hat{x}_{|L}(0)$ (in the clear) and an algebraic commitment (y, γ, k) to the remainder of an interpolating set of the degree- dm polynomial $\hat{x}_{|L} : \mathbb{F} \rightarrow \mathbb{F}$, i.e., to the field elements $\hat{x}_{|L}(i)$ for all $i \in [dm]$. The commitment consists of a random matrix $y \sim \mathbb{F}^{dm \times p}$ with dm rows and a large enough number p of columns; a random (column) coordinate $k \sim [p]$; and the correction tuple γ satisfying $\gamma_i = \hat{x}_{|L}(i) - y_{ik}$.

V samples (another) evaluation point σ and computes the fingerprint $y(\sigma, \beta) = \sum_i \beta_i \hat{y}_i(\sigma)$, where the tuple β satisfies $\sum_i \beta_i \hat{x}_{|L}(i) = \hat{x}(\rho)$;⁶ it also computes $\gamma = \sum_i \beta_i \gamma_i$ and stores k .

Step 3: temporal decommitment

V reveals its fingerprint's evaluation point ρ along with the index ℓ where it appeared in z . The prover checks that $z_\ell = \rho$, and only continues to the final step if the check passes.

Step 4: algebraic decommitment

P and V engage in the decommitment of the k^{th} coordinate of the string $y' = \beta \cdot y$ (the linear combination of the rows y_i with coefficients β_i).⁷ V outputs the (alleged) x_j if the decommitment is consistent with $\hat{x}(\rho)$, and rejects otherwise.

In an honest execution of the above protocol, the final decommitment reveals

$$\begin{aligned} y'_k &= \sum_i \beta_i y_{ik} \\ &= \sum_i \beta_i (\hat{x}_{|L}(i) - \gamma_i) \\ &= \hat{x}(\rho) - \gamma, \end{aligned}$$

so that V , having stored $\hat{x}(\rho)$ and γ , can indeed perform this consistency check (which shows the protocol is complete). The protocol's soundness follows from that of **pep**, noting that none of the mechanisms we add harm soundness (indeed, the last check relies, as does **pep**, on a random evaluation of the low-degree extension), while zero-knowledge, which we discuss next, follows from the correctness of our commitment protocols.

Proving the zero knowledge property

We conclude with a discussion of the simulator argument for the protocol laid out in this section. Recall that proving zero-knowledge for the foregoing protocol entails the construction of a *simulator* S , a streaming algorithm with knowledge of x_j and roughly the same memory as \tilde{V} , which is able to interact with \tilde{V} without it being able to tell whether it is communicating with S or P .

⁶ Note that β_i is determined solely by i and ρ : it is the evaluation $\chi_i(\rho)$ of the i^{th} Lagrange polynomial.

⁷ This requires P to know the linear coefficients β , and, while we could have the verifier send them, this is not necessary: P learns ρ in step 3, which allows it to determine $\rho = L^{-1}(\rho)$ and thus $\beta = \beta(\rho)$ as well.

Roughly speaking, S does the following: after the temporal commitment step, it inspects the memory state of \tilde{V} and records (almost) all the points to which \tilde{V} can decommit; as shown in the last section, this is a relatively small set C . It then streams the input and records $\hat{x}(\rho)$ for all $\rho \in C$.⁸ Upon receipt of a line L from \tilde{V} , the simulator computes and commits to an arbitrary low-degree polynomial g that interpolates across the points in $L \cap C$. When \tilde{V} requests the algebraic decommitment to obtain an evaluation of g , the simulator checks that the evaluation point ρ is contained in C (in which case $g(\rho)$ matches a fingerprint $\hat{x}(\rho)$ known to S), proceeds with the decommitment if that is the case, and otherwise aborts.

We note that implementing the strategy above raises yet another challenge, namely, extracting the set C of evaluation points from the description and memory state of \tilde{V} . This is accomplished via a form of *white-box access* to \tilde{V} , see Section 4.

The simulator S is thus able to generate the transcript of an interaction where the message $\hat{x}|_L$ of the algebraic commitment is replaced with another low-degree polynomial g whose evaluations match $\hat{x}|_L$ at all points where \tilde{V} is able to temporally decommit. Then, distinguishing between a real and a simulated transcript amounts to distinguishing an INDEX instance whose solution is $\hat{x}|_L$ from one whose solution is g .

We prove that any streaming algorithm that does so with nontrivial bias implies a one-way communication protocol for INDEX with a small message, contradicting the known hardness of the problem. We remark that the reduction is rather nontrivial, as we must insert an INDEX instance into the algebraic commitment (y, γ, k) while ensuring the decommitment can be simulated without any knowledge about the instance. See Theorem 36 for details.

► **Remark 8 (Superpolynomial to near-linear communication).** We stress that, while we may prove zero-knowledge with the strategy above, the natural reduction from INDEX is over a large alphabet $\Gamma = \mathbb{F}^{dm}$. But then, for indistinguishability to follow, the length p of the temporal commitment must be q^{dm} , which implies *superpolynomial* communication complexity.

We avoid this blowup via Lemma 25, which shows that an INDEX (one-way) protocol for large alphabets implies another protocol for the binary alphabet with only a mild loss to its success probability; this restricts our ambient field to be an extension of \mathbb{F}_2 , but reduces the superpolynomial complexity to *barely superlinear*.

2.5 A general-purpose zero-knowledge SIP: sumcheck

Lastly, we briefly mention how the commitment protocols developed in Sections 2.2 and 2.3 can be used not only to solve INDEX (and, more generally, the polynomial evaluation problem), but also to construct another widely applicable tool: a streaming zero-knowledge *sumcheck* protocol.

As before, we start with an SIP that is clearly not zero-knowledge: the standard sumcheck protocol leaks hard-to-compute sums over subcubes. By carefully using the algebraic and temporal commitment protocols, we can also endow the sumcheck protocol with zero-knowledge in the data stream model. However, we note that doing so is considerably more involved than in the case of INDEX, owing to, among other reasons, several rounds of interaction with nontrivial dependencies of messages on past communication.

More precisely, we consider a slight variation of the standard sumcheck protocol: while in the latter every round is followed by a (random) consistency check, we instead defer all such checks to the end. It is clear that this variant is equivalent to the standard protocol; however,

⁸ We note that storing C is the most space-intensive task of S , which implies a small overhead to its space complexity as compared to \tilde{V} ; see Theorem 36.

without the modification, the zero-knowledge property seems to require a strengthening of the chained commit-decommit strategy we follow. Moreover, rather than a single algebraic commitment followed by a (single) decommitment, the sumcheck protocol requires many decommitments; indeed, for an m -variate polynomial f , the prover commits to m partial sums of f , and each partial sum is involved in two decommitments (for a total of $m + 1$ decommitments).

Therefore, by extending the techniques that underpin our approach for the INDEX problem to a *multi-round* setting, we are able to construct a zero-knowledge sumcheck SIP. Such a protocol can then be used to compute frequency moments and inner products, problems known to require linear space without a prover's assistance [2]. See Section 7 for details.

3 Preliminaries

General notation

For an integer $k \geq 1$, we denote by $[k]$ the set $\{1, 2, \dots, k\}$. Vectors are denoted with notation analogous to that of sets, i.e., $(\alpha_i : i \in [k])$ denotes the vector $(\alpha_1, \dots, \alpha_k)$. We use n to denote the length of a string that is the input to an algorithm, and $\text{poly}(n)$ (respectively, $\text{polylog}(n)$) to denote an arbitrary polynomial (respectively, polylogarithmic) function in n .

We use lowercase Latin letters to denote positive integers (e.g., $d, i, j, k, \ell, m, p, v$) or strings (e.g., x, y, z); r and t often (but not always) denote random strings. Lowercase Greek letters denote elements of a finite alphabet or field (e.g., α, β, γ), and we reserve ρ, σ for random elements. Uppercase letters denote either algorithms (e.g., A, B, P, S, V) or sets (e.g., C, K), with T used as the indeterminate of a polynomial.

When f and g are functions, we sometimes use $\alpha \in f$ as a shorthand for $\alpha \in \text{Im } f$ and $f|_g$ for $f \circ g$; if f is a low-degree polynomial that is communicated in an interactive protocol, we assume it is sent in a canonical form (e.g., a line is communicated by a pair of points $f(0), f(1)$). We use $\mathbb{1}[\cdot = x_0]$ to denote the delta function at x_0 (i.e., $\mathbb{1}[x_0 = x_0] = 1$ and $\mathbb{1}[x = x_0] = 0$ for $x \neq x_0$) and \log to denote \log_2 .

As integrality issues do not substantially change any of our results, equality between an integer and an expression (that may not necessarily evaluate to one) is assumed to be rounded to the nearest integer.

Vectors and matrices

The notation we use for matrices is the same as for strings (lowercase Latin letters), and it will be clear from context which is the case. When x is a matrix, we use x_i to refer to the i^{th} row of x .

We use vectors or tuples, interchangeably, to refer to elements of a vector space over a finite field \mathbb{F} . Such tuples are denoted with boldface (e.g., α, β, γ) and random tuples are (similarly to strings) denoted ρ, σ . We use $\alpha \cdot \beta$ to denote the inner product between the two vectors, and, when the dimension of α matches the number of rows of a matrix x , we use $\alpha \cdot x$ to denote the vector corresponding to the linear combination of the rows of x with coefficients α , i.e., $\sum_i \alpha_i x_i$. (Equivalently, we assume vectors to be in row form.)

Probability

We use $X \sim \mu$ to denote a random variable with distribution μ , and, for the uniform distribution over a set S , we write $X \sim S$. We sometimes make the sources of randomness in a probabilistic expression explicit, and when we do they are assumed to be independent; e.g.,

only when X and Y are independent do we write $\mathbb{P}_{X \sim \mu, Y \sim \lambda}[E]$. The internal randomness of an algorithm is generally omitted; e.g., $\mathbb{P}[A(X) = 0]$ (if the distribution of X is known from context) or $\mathbb{P}_{X \sim \mu}[A(X) = 0]$ are shorthand for $\mathbb{P}_{X \sim \mu, r \sim \{0,1\}^m}[A(X; r) = 0]$, where r is A ’s internal randomness.

We will also make use of the following versions of the Chernoff and Hoeffding bounds.

► **Lemma 9** (Additive Chernoff-Hoeffding bound). *Let X_1, \dots, X_k be independent Bernoulli random variables distributed as X . Then, for every $\delta \in [0, 1]$,*

$$\Pr \left[\frac{1}{k} \sum_{i=1}^k X_i \leq \mathbb{E}[X] - \delta \right] \leq e^{-2\delta^2 k} \text{ and}$$

$$\Pr \left[\frac{1}{k} \sum_{i=1}^k X_i \geq \mathbb{E}[X] + \delta \right] \leq e^{-2\delta^2 k}.$$

► **Lemma 10** (Hoeffding’s inequality). *Let X_1, \dots, X_k be independent random variables distributed as $X \in [a, b]$. Then, for every $\delta \in [0, 1]$,*

$$\Pr \left[\frac{1}{k} \sum_{i=1}^k X_i \leq (1 - \delta)\mathbb{E}[X] \right] \leq e^{-\left(\frac{\delta\mathbb{E}[X]}{b-a}\right)^2 k} \text{ and}$$

$$\Pr \left[\frac{1}{k} \sum_{i=1}^k X_i \geq (1 + \delta)\mathbb{E}[X] \right] \leq e^{-\left(\frac{\delta\mathbb{E}[X]}{b-a}\right)^2 k}.$$

Algorithms and protocols

We use the same term to refer to computational problems and to protocols that solve them, but distinguish the two cases with different fonts (so that the **pep** and **sumcheck** protocols solve the PEP and SUMCHECK problems, respectively).

We generally use A , D , S and V to denote streaming algorithms, while P denotes an algorithm with unbounded computational resources (including space). $A(x)$ is the output of an algorithm that receives x as input; when A is a streaming algorithm, x is read sequentially in one pass, from the first symbol (x_1) to the last. When $A(x, y, z)$ reads multiple inputs, $A(y)$ denotes the partial execution of A after it has read x . When the entries of a length- n string x are taken over a finite alphabet Γ , we may also use x for the equivalent bit string of length $n \log |\Gamma|$.

We shall often make use of the *minimax principle*, and assume, without loss of generality, that a computationally unbounded algorithm A whose goal is to maximise some value $\mathbb{E}_{x \sim \mu}[f(A(x))]$ (e.g., the probability that $A(x)$ equals x) can be assumed to be deterministic, and thus given by a function $x \mapsto a(x)$; equivalently, A can be taken as the deterministic algorithm that maximises $\mathbb{E}[f \circ a(x)]$ for the distribution of inputs μ .

In a protocol, two algorithms P and V interact by exchanging messages in a predefined order; after all messages have been exchanged, V chooses an output that we denote $\langle P, V \rangle$ and call the output of the protocol. When V rejects or P aborts midway through the interaction, we assume the algorithm proceeds until the end of the protocol with dummy messages (e.g., strings of zeroes).

The *snapshot* of an algorithm is synonymous to its memory state; when A reads a sequence of more than one input, e.g., $A(x, y)$, the “snapshot of A after x ” is the snapshot immediately before the first symbol of y is streamed (i.e., after A has read and processed the last symbol of x). When A is interacting in a protocol and sends a message between reading x and y , the snapshot after x is that immediately before sending the message.

Low-degree extensions

For any field \mathbb{F} and integer k such that $|\mathbb{F}| \geq k$, we consider $[k] \subseteq \mathbb{F}$ via a canonical injection (e.g., taking the image of $\ell \in [k]$ as the field element whose binary representation is the same as that of ℓ). Accordingly, we write $\ell \in \mathbb{F}$ as shorthand for the field element corresponding to the image of $\ell \in [k]$ via this canonical injection.

For a string $y \in \mathbb{F}^k$, the *low-degree extension* (LDE) with *degree* d and *dimension* m where $|\mathbb{F}| \geq d + 1$ and $k \leq (d + 1)^m$, denoted \hat{y} , is the unique m -variate polynomial of individual degree d that coincides with y in $[k]$; more precisely, viewing $[k] \subseteq [d + 1]^m \subseteq \mathbb{F}^m$, the LDE $\hat{y} : \mathbb{F}^m \rightarrow \mathbb{F}$ is the unique polynomial satisfying $\hat{y}(i) = y_i$ for all $i \in [k]$. Our notation for the polynomial \hat{y} omits the degree and dimension, as they will be clear from context.

When y is a matrix, we use $\hat{y}(\alpha, \beta)$ to denote the linear combination of the LDEs of the rows with linear coefficients β , i.e., $\hat{y}(\alpha, \beta) = \sum_i \beta_i \hat{y}_i(\alpha)$.

3.1 Information theory

We will make use of several notions of information theory and approximations of information-theoretic quantities. The q -ary *entropy function* $H_q : [0, 1] \rightarrow [0, 1]$ is

$$\begin{aligned} H_q(t) &= t \log_q(q-1) - t \log_q t - (1-t) \log_q(1-t) \\ &= \frac{1}{\log q} (t \log(q-1) - t \log t - (1-t) \log(1-t)) \\ &= \frac{1}{\log q} (t \log(q-1) + H_2(t)), \end{aligned} \quad (1)$$

where $H_q(0) = 0$; we also use the shorthand H for H_2 , which simplifies to

$$H(t) = H(1-t) = -t \log t - (1-t) \log(1-t). \quad (2)$$

We will make use of the following approximation for the (natural) logarithm function: for $0 \leq t \leq 1/2$,

$$-t(1+t) \leq \ln(1-t) \leq -t. \quad (3)$$

The (relative) *Hamming distance* between two strings $a, b \in \Gamma^k$ over a finite alphabet is the fraction of coordinates where they differ, i.e., $d(a, b) = \frac{1}{k} |\{i \in [k] : a_i \neq b_i\}| \in [0, 1]$. With $\gamma = |\Gamma|$, the volume of a *Hamming ball* $\mathcal{B}(b, \delta) := \{a \in \Gamma^k : d(a, b) \leq \delta\}$ of radius $\delta = 1 - \varepsilon$, when k is large enough and $\varepsilon = k^{-1} \text{polylog}(k)$, satisfies⁹

$$\gamma^{H_\gamma(\delta)k} \geq |\mathcal{B}(b, \delta)| = \Omega\left(\frac{\gamma^{H_\gamma(\delta)k}}{\sqrt{\varepsilon k}}\right) = \frac{\gamma^{H_\gamma(\delta)k}}{\text{polylog}(k)}. \quad (4)$$

The entropy of a discrete random variable X taking values in Γ is

$$H(X) = - \sum_{\alpha \in \Gamma} \mathbb{P}[X = \alpha] \log(\mathbb{P}[X = \alpha]).$$

⁹ The lower bound is a simplification of

$$|\mathcal{B}(b, \delta)| \geq \gamma^{H_\gamma(\delta)k} \cdot \exp\left(\frac{1}{12k+1} - \frac{1}{12\delta k} - \frac{1}{12\varepsilon k}\right) / \sqrt{2\pi\delta(1-\delta)k};$$

since $1/\varepsilon k = o(1)$, the numerator is $1 - o(1)$, and the denominator is of order $\Theta(\sqrt{\varepsilon k}) = \text{polylog}(k)$. (See, e.g., [41].)

Every such random variable satisfies

$$H(X) \in [0, \log |\Gamma|]. \quad (5)$$

The conditional entropy $H(X|Y)$ is the entropy of the conditional random variable, which satisfies

$$H(X|Y) \leq H(X). \quad (6)$$

If X, Y are independent, then

$$H(X, Y) = H(X) + H(Y). \quad (7)$$

The last property of entropy we will make use is the *chain rule*: for random variables X_1, \dots, X_n ,

$$H(X_1, \dots, X_n) = \sum_{i=1}^n H(X_i | X_1, \dots, X_{i-1}). \quad (8)$$

For ease of notation, when (X, Y) are jointly distributed over Γ^2 with marginals μ and λ , respectively, we denote the distribution of Y conditioned on $X = x$ as λ_x . The *KL divergence* between the distributions is

$$\text{KL}(\mu \parallel \lambda) = \sum_{\alpha \in \Gamma} \mu(\alpha) \log \frac{\mu(\alpha)}{\lambda(\alpha)}, \quad (9)$$

which upper bounds the Euclidean distance between probability vectors via *Pinsker’s inequality* (see, e.g., [10]):

$$\|\mu - \lambda\|^2 \leq \frac{\text{KL}(\mu \parallel \lambda)}{2 \ln 2}. \quad (10)$$

Finally, the *mutual information* is defined as (and equivalent to)

$$\begin{aligned} I(\mu : \lambda) &:= I(X : Y) \\ &= I(Y : X) \\ &= H(Y) - H(Y|X) \\ &= \mathbb{E}_{X \sim \mu}[\text{KL}(\lambda_X \parallel \lambda)]. \end{aligned} \quad (11)$$

4 Zero-knowledge streaming interactive proofs

This section motivates and provides a definition of zero-knowledge proofs in the data stream model. We start by discussing the differences between the streaming and the traditional settings as well as establish necessary notation. We then we provide a formal definition in Section 4.1.

The notion of zero-knowledge proofs in a computational model should capture the intuition that, when engaged in an interactive protocol, a verifier algorithm V should learn nothing but the truth of some hard-to-compute statement about its input x (e.g., that x is in a language L). For consistency with the general notion we define zero-knowledge for *decision problems* in the streaming model, but remark that the definition extends to search problems in the standard way (i.e., the verifier V learns nothing but a valid solution to the search problem).

In the traditional setting, V can easily store the entirety of x and make polynomial-time computations without the assistance of a prover. This implies that the sensitive information a zero-knowledge proof in this setting must not leak is the result of a computation on x beyond the verifier’s reach, i.e., one that requires superpolynomial time to obtain from the information available to V . In the streaming setting, however, the notion of “hard-to-compute” changes dramatically: the model puts *space* as the primary resource, so that computations within the reach of V are those possible with a small amount of space and sequential one-pass access to the input (but arbitrarily large time complexity). Knowledge then essentially corresponds to all information that V cannot compute in low space complexity using its streaming access. As a result, zero-knowledge streaming interactive proofs (zkSIPs) must satisfy a much more stringent requirement: that they not leak any information *about the input x itself* (which in the traditional setting is fully known to the verifier).

In order to capture such a stringent notion of sensitive information, we define zkSIPs as protocols such that no *streaming* algorithm can distinguish a real transcript of the protocol from one that is generated by a (streaming) simulator. To this end, we first recall the formalisation of *streaming interactive proofs* (SIPs) [24] without any zero-knowledge requirement.

► **Definition 11.** A streaming interactive proof (SIP) for a language L is an interactive proof defined by a pair (P, V) of algorithms: a computationally unbounded prover P and streaming verifier V with space $s = o(n)$. The verifier engages in an interactive protocol with P and streams, at a predetermined step, the bit string $x \in \{0, 1\}^n$, which P also observes.¹⁰ At the end of the protocol, V outputs a binary decision $\langle P, V \rangle(x)$ satisfying

- (completeness) if $x \in L$, then $\mathbb{P}[\langle P, V \rangle(x) = 1] \geq 2/3$; and
- (soundness) if $x \notin L$, then $\mathbb{P}[\langle P, V \rangle(x) = 1] \leq 1/3$.

We call s the *space complexity* (of the verifier). Note that, while the constant $1/3$ is arbitrary, soundness amplification does not hold for streaming algorithms due to the need to reread the input; nevertheless, many SIPs (including all those considered in this paper) allow for improving soundness by a desired factor with a logarithmic increase to their space complexity (see Section 5.1). We stress that Definition 11 constrains the verifier *only* in terms of space, which allows arbitrarily large time complexities for both prover and verifier. (This is similar to other settings such as communication complexity and property testing, where the primary resources are communication and queries, respectively.)

Loosely speaking, we capture the notion of zero-knowledge in the data stream model by saying that an SIP is zero-knowledge if there exists a streaming *simulator algorithm* S , with roughly the same space as the verifier V , able to simulate a prover-verifier interaction that is indistinguishable from a real one; that is, S generates a *view* of the verifier (defined next) that no *distinguisher* algorithm with power comparable to V (i.e., a streaming algorithm with roughly the same space) can tell apart from a real interaction. We stress that while the distinguisher D is reminiscent of computational zero-knowledge, the security of our protocols is information-theoretic and *does not rely on computational assumptions*.

¹⁰The definition could allow for alternating between streaming parts of x and communicating with the prover, as well as adaptively choosing the round(s) on which to read the input. Our protocols do not require this flexibility, however, so we assume the entirety of x is read at a fixed step along the communication protocol.

► **Definition 12.** Let (P, V) be an SIP with a space- s verifier, where P sends k_1 messages to V before the verifier streams its input, and an additional k_2 messages afterwards. Denote the prover’s messages by $y_1 \in \{0, 1\}^{p_1}, \dots, y_{k_1+k_2} \in \{0, 1\}^{p_{k_1+k_2}}$; the input by x ; and the verifier’s and prover’s internal randomness by r and t , respectively.

The view of the verifier \tilde{V} , denoted $\text{View}_{P, \tilde{V}}(x, r)$, is the random variable defined as

$$\text{View}_{P, \tilde{V}}(x, r; t) = (r, y_1, \dots, y_{k_1}, x, y_{k_1+1}, \dots, y_{k_1+k_2}).^{11}$$

While Definition 12 is similar to its polynomial-time analogue, we highlight an important distinction: to faithfully correspond to what \tilde{V} sees, the order in which the view is streamed must be preserved. Indeed, a step-by-step execution of \tilde{V} in an interaction with P corresponds exactly to its streaming $\text{View}_{P, \tilde{V}}(x, r)$ one symbol at a time. Order preservation is also consistent with the input stream x being observed by all parties simultaneously (which are, in a simulation, \tilde{V} , the simulator S and a distinguisher D).

4.1 Definition

We now ready to give a formal definition of zero-knowledge streaming interactive proofs.

► **Definition 13 (zkSIP).** Let L be a language and (P, V, S) be a triplet where (P, V) is an SIP with a space- s verifier V and S is a streaming poly(s)-space simulator with white-box access to the verifier, streaming access to the input x and additional query access to a random bit string t .

(P, V, S) forms a zero-knowledge streaming interactive proof (zkSIP) for L that is secure against space- s' adversaries if, for any space- s algorithm \tilde{V} and $x \in L$, the random variables $\text{View}_{P, \tilde{V}}(x, r)$ and $S(\tilde{V}, x, r)$ are indistinguishable by any streaming space- s' algorithm. That is, for every space- s' streaming algorithm D ,

$$\left| \mathbb{P} \left[D(\text{View}_{P, \tilde{V}}(x, r)) \text{ accepts} \right] - \mathbb{P} \left[D(S(\tilde{V}, x, r)) \text{ accepts} \right] \right| = o(1).$$

We note that all our applications have $s = \text{polylog}(n)$, and the protocols are secure against adversaries with any space $s' = \text{poly}(s)$ (see Remark 38).

► **Remark 14.** Recall that the analogue of Definition 13 in the polynomial-time setting requires a much stronger notion of indistinguishability: *negligible* (i.e., sub-inverse-polynomial), rather than $o(1)$, bias. This is necessary for the notion to be robust with respect to poly-time algorithms, as otherwise repeating polynomially many executions of D would boost its success probability arbitrarily close to 1.

This raises a number of interesting questions on the achievable notions of security for zkSIPs: can we obtain tighter bounds, such as $1/\text{poly}(n)$ or negligible? (Perhaps even in the statistical case?) An answer to each such question ensures security against one type of adversary (i.e., distinguisher): we will study the natural threat model where all parties are streaming algorithms and argue why $o(1)$ is a sufficient bound in this case. Before doing so, however, we briefly discuss an important alternative.

As explained above, streaming verifiers secure against polynomial-time adversaries require negligible distinguishability. This has been previously studied, most notably for zero-knowledge interactive proofs that reduce to evaluating low-degree polynomials defined by the input and allow for it to be processed in a streaming fashion, such as [34]. (We stress, however, that such protocols rely on computational assumptions.) An interesting question that we leave to future work is whether zkSIPs can *simultaneously* achieve security against different adversaries – e.g., with negligible bias for poly-time distinguishers (under cryptographic assumptions) in addition to subconstant bias for streaming distinguishers.

Recall that a key distinction between the poly-time and streaming settings is the *one-pass* restriction of the latter, which prevents even a single repetition of (a streaming) D – indeed, INDEX trivialises with 2 passes (as do many fundamental streaming problems). In other words, as the common technique of *amplification* is unavailable in the streaming model, $o(1)$ bias is a sufficiently robust requirement that guarantees the probability of information leakage tends to 0. (We note that the weaker requirement of arbitrarily small constant bias would also suffice, i.e., the existence of $(P_\varepsilon, V_\varepsilon, S_\varepsilon)$ achieving ε bias for every $\varepsilon > 0$. We adopt the simpler and stronger subconstant version, which our protocols satisfy.)

The streaming simulator

For technical reasons, the simulator is given white-box access to the verifier and explicit access to a random string. We stress that this auxiliary information is completely independent of the input. This can be viewed as allowing the verifier to obtain some computation about auxiliary information (about its own strategy, or a uniformly chosen random string), but learn absolutely *zero information about the input stream* x .

While white-box access gives the simulator S knowledge of any function of the verifier’s strategy, we do not require such generality; indeed, we will only be interested in questions about the most likely messages that \tilde{V} may send at a single point of the protocol. As such, the weaker definition that follows is sufficient.

► **Definition 15.** *Let A be a space- s streaming algorithm that reads an n -bit string y and outputs an m -bit string z . We define white-box access to A as oracle access to a function \mathcal{W} with two inputs, a snapshot $b \in \{0, 1\}^s$ and a candidate output $z \in \{0, 1\}^m$; the oracle returns the maximum probability over all inputs y with which A , starting with memory state b , outputs z ; that is,*

$$\mathcal{W}(b, z) = \max_{y \in \{0, 1\}^n} \{\mathbb{P}[A(y) \text{ outputs } z \text{ when its initial snapshot is } b]\}.$$

► **Remark 16.** While the honest verifier V does not use a large random string, malicious verifiers \tilde{V} with this additional resource can readily be simulated by S as above. We assume hereafter that \tilde{V} has the same resources as the honest verifier, but note that the simulations extend straightforwardly to verifiers with both white-box access (to their strategies) and query access to a random string.

5 Algebraic and temporal commitments

A commitment protocol is a two-party protocol (or, more accurately, a pair of protocols) that allows the transmission of a message from one party to another to be split into two parts: a *commitment*, where the message is transmitted in a form that cannot be interpreted by the recipient; followed, at some point in the future, by a *decommitment*, where the sender transmits additional information with which the recipient can read the message. (A useful analogy is that the commitment amounts to sending a locked box containing the message, and the decommitment to sending the key.)

In the standard setting [9] we have two parties: a sender and a receiver, which we will refer to as prover and verifier, respectively. The prover wishes to communicate a symbol α , and does so by first choosing a random *key* k and sending another string $c = \text{commit}(\alpha, k)$. Then, at some point in the future, prover and verifier engage in a protocol at the end of which the receiver obtains $\alpha = \text{decommit}(c)$. (We will refer to the streaming analogue as a commitment *protocol*, rather than *scheme*, to avoid ambiguity with the polynomial-time analogue.)

Commitment protocols are extremely useful components for the construction of interactive protocols, and should satisfy two properties: *hiding*, i.e., the commitment alone should prevent the verifier from obtaining a non-negligible amount of information about the message α ; and *binding*, i.e., the prover should not be able to decommit to a message that differs from the one it committed to. We will construct a commitment protocol whose hiding property follows from the average-case hardness of SEARCH-INDEX for streaming algorithms, while binding follows from the soundness of the pep protocol (which we introduce formally in Section 5.1).

We first formally define streaming commitment protocols. We note that while the definition that follows can be generalised,¹² it suffices to capture our constructions.

► **Definition 17.** A streaming commitment protocol for alphabet Γ (with security parameter p) and space bound s consists of a function $\text{commit} : \Gamma \times K \rightarrow C$, where $K \subseteq \{0, 1\}^p$ is the set of keys and C is the set of commitments, and a space- s SIP (P, V) which satisfy the following conditions.

- **Hiding:** Fix any pair of distinct messages $\alpha, \beta \in \Gamma$ and sample $k \sim K$. Set $c = \text{commit}(\alpha) = \text{commit}(\alpha, k)$ and $c' = \text{commit}(\beta) = \text{commit}(\beta, k)$. Every (streaming) space- s distinguisher D tells the two commitments apart with at most subconstant bias (with respect to the parameter p); that is,

$$|\mathbb{P}[D(c) \text{ accepts}] - \mathbb{P}[D(c') \text{ accepts}]| = o(1).$$

- **Binding:** Fix $k \in K$ and $\alpha \in \Gamma$. Then

$$\mathbb{P}[\langle P, V \rangle(\text{commit}(\alpha, k), \alpha) = 1] = 1,$$

and for any $\beta \neq \alpha$,

$$\mathbb{P}[\langle P, V \rangle(\text{commit}(\alpha, k), \beta) = 1] = o(1).$$

Note that, with some abuse of notation, the binding condition corresponds to (P, V) being an SIP for the language $L = \{(\text{commit}(\alpha, k), \alpha) : \alpha \in \Gamma, k \in K\}$.

The next sections introduce the commitment protocols we will use to build our protocols. Section 5.1 begins by defining the concepts and tools we build upon: low-degree extensions and the polynomial evaluation protocol (pep). In Section 5.2, we use them to construct a basic scheme that allows for the communication of a single symbol (which we use as a stepping stone), based on the hardness of INDEX (or, more accurately, SEARCH-INDEX); in it, the keys are simply long strings paired with a coordinate, i.e., $K = \Gamma^p \times [p]$, and commitments are keys appended with a single extra symbol (i.e., $C \subset \Gamma^{p+1} \times [p]$).

Section 5.3 then extends the construction of Section 5.2 into an *algebraic* commitment protocol, which allows for the commitment of low-degree polynomials. In both the basic and algebraic schemes, hiding is achieved by overwhelming V with “too much information”, and can only be broken if a malicious verifier is lucky enough to retain a critical fragment of the information stream; indeed, as we will see, breaking it amounts to solving INDEX. Binding, on the other hand, relies on the pep protocol, which we introduce in the next section.

While commitment protocols are not a prerequisite for a zero-knowledge protocol, they also serve as inspiration for our second main component: Section 5.4 shows how the verifier can perform a *temporal commitment* to show its alleged internal randomness is uncorrelated with its input, and thus that it is not behaving maliciously.

¹²A natural generalisation is to parameterise the bias in the hiding property as well as the completeness and soundness in binding by $\varepsilon_b, \varepsilon_c, \varepsilon_s \in (0, 1)$; our definition has $\varepsilon_b, \varepsilon_s = o(1)$ and $\varepsilon_c = 0$.

5.1 Low-degree extensions and polynomial evaluation

Fingerprinting is a technique that enables streaming algorithms to approximately verify an arbitrary coordinate of a long string in small space. It exploits *low-degree extensions* (LDEs), extremely useful objects in the design of interactive proofs more broadly.

Given a data set x , viewed as a string of n elements in a finite field $\mathbb{F} = \mathbb{F}_q$, an LDE is a low-degree polynomial that interpolates every data point. More precisely, we may view x as a function $x : [n] \rightarrow \mathbb{F}$; given a *dimension* m and defining the *degree* d as the smallest (positive) integer such that $n \leq (d+1)^m$, we can also view $x : [d+1]^m \rightarrow \mathbb{F}$ by some canonical injection $[n] \hookrightarrow [d+1]^m$ (padding with zeroes if $n < (d+1)^m$). Then, as long as $q > d$, we can also view (via another canonical injection $[d+1] \hookrightarrow \mathbb{F}$) the data set as the restriction of a function from \mathbb{F}^m to \mathbb{F} .

Standard properties of polynomials imply that if this function is an m -variate polynomial of individual degree d , then the extension is unique; we thus denote by $\hat{x} : \mathbb{F}^m \rightarrow \mathbb{F}$ the unique degree- d polynomial whose restriction to $[n]$ is equal to x . Explicitly, with (i_1, \dots, i_m) as the image of i by $[n] \hookrightarrow \mathbb{F}^m$,

$$\hat{x} = \sum_{i=1}^n x_i \chi_i = \sum_{i_1, \dots, i_m \in [d+1]} x_{i_1, \dots, i_m} \chi_{i_1, \dots, i_m}$$

where the χ_i are the Lagrange basis polynomials, given by

$$\chi_i(\alpha_1, \dots, \alpha_m) := \prod_{j=1}^m \prod_{\substack{k=1 \\ k \neq i_j}}^{d+1} \frac{\alpha_j - k}{i_j - k}$$

(viewing $k \in [d+1]$ as an element of \mathbb{F}); equivalently, the Lagrange polynomials are the unique m -variate degree- d polynomials satisfying $\chi_i(j) = \mathbb{1}[i=j]$ when $i, j \in [d+1]$. We note that LDEs and Lagrange polynomials can equivalently be defined with an injection from $\{0\} \cup [d]$, rather than $[d+1]$, to \mathbb{F} ; then they satisfy the previous condition for all $0 \leq i, j \leq d$. We will use the characterisation that is most convenient, which will be clear from context (e.g., an LDE that involves the evaluation of a polynomial at 0 is of the latter type).

We will also use $\chi(\alpha)$ to denote the vector $(\chi_1(\alpha), \dots, \chi_n(\alpha))$ of evaluations of Lagrange polynomials; note that this allows us to write $\hat{x}(\alpha)$ as the dot product $\chi(\alpha) \cdot x$ of n -dimensional vectors.

Now, given a string $x \in \mathbb{F}^n$, a *fingerprint* is simply an evaluation of the LDE of x at a random point, that is, $\hat{x}(\rho)$ with $\rho \sim \mathbb{F}^m$. The key property of fingerprints is that they are extremely unlikely to match for two different strings when the underlying field is large enough, as a consequence of the Schwartz-Zippel lemma [53, 48].

► **Lemma 18** (Schwartz-Zippel). *If $x, y \in \mathbb{F}_q^n$ are distinct, then $\mathbb{P}_{\rho \sim \mathbb{F}^m} [\hat{x}(\rho) = \hat{y}(\rho)] \leq dm/q$.*

Importantly for streaming algorithms, fingerprints can be computed with $O(dm)$ time per entry of the input and $O(m)$ field elements (thus $O(m \log q)$ bits) of space [24].

The polynomial evaluation protocol is an interactive proof that enables a streaming verifier with a single random evaluation $f(\rho)$ of a degree- d polynomial $f : \mathbb{F}^m \rightarrow \mathbb{F}$ to evaluate f at any other point, assisted by a prover with knowledge of f in its entirety. Note that the prover could help the verifier compute f at a point (non-interactively) by simply sending an interpolating set of the polynomial; but any such set has size $(d+1)^m$. The **pep** (polynomial evaluation) protocol, detailed in Figure 3, allows us to reduce the communication from $O(d^m \log q)$ to $O(dm \log q)$ by adding interaction.

Input: Explicit access to $\alpha \in \mathbb{F}$ and a set $f \subseteq \{f^x : x \in \mathbb{F}^n\}$ of m -variate degree- d polynomials over \mathbb{F} . Streaming access to $(x, \beta) \in \mathbb{F}^n \times \mathbb{F}^m$.

V: Sample $\rho \sim \mathbb{F}^m$. Stream x and compute $f^x(\rho)$. Store β .

Compute the line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ such that $L(0) = \beta$ and $L(\rho) = \rho$ with $\rho \sim \mathbb{F}$, then send L to the prover.

P: Compute and send $f_{|L}^x$.¹³

V: Compute $g(\rho)$, where $g : \mathbb{F} \rightarrow \mathbb{F}$ is the degree- dm low-degree extension of the sequence of evaluations sent by P such that $g(0) = \alpha$.¹⁴ Accept if $g(\rho) = f^x(\rho)$ and reject otherwise.

■ **Figure 3** Protocol $\text{pep}(f, \alpha)$.

In order to better compare the original pep protocol with the zero-knowledge version that we will construct, we consider a general problem that the protocol is able to solve (as in [15]). We use f as shorthand for a mapping $x \mapsto f^x$ (or, equivalently, a set $f \subseteq \{f^x : x \in \mathbb{F}^n\}$) where one evaluation $f^x(\rho)$ can be computed by a space-bounded algorithm that streams x . The problem $\text{PEP}(f, \alpha)$ is to decide whether $f^x(\beta) = \alpha$ when the input stream is x followed by an evaluation point $\beta \in \mathbb{F}^m$.

Assuming an evaluation of f^x can be computed by streaming x with $O(m \log q)$ space, Figure 3 is a streaming interactive proof for $\text{PEP}(f, \alpha)$ with communication complexity $O(dm \log q)$ and verifier space complexity $O(m \log q)$. We note that $\text{pep}(f, \alpha)$ can easily be modified into an algorithm for a search problem without a candidate value α for $f^x(\beta)$, by having V output $g(0)$ instead of accepting.

It is clear that V accepts in Figure 3 when P is honest; the protocol’s soundness relies on the fact that if the prover were to send an incorrect $g \neq f_{|L}^x$, it is highly unlikely that it will agree with the verifier’s evaluation at the (unknown) location ρ .

In conjunction with the streaming nature of LDEs, (the search version of) Figure 3 yields a simple and efficient streaming interactive proof for SEARCH-INDEX . This SIP, introduced by [15], has $O(\log n \log \log n)$ space and communication complexities for a stream $(x, j) \in \mathbb{F}^n \times [n]$ where $q = |\mathbb{F}| = \text{polylog}(n)$ (and $\beta \in \mathbb{F}^m$ is the identification of j); it is simply an instantiation of pep where $d = 2$, $m = \log n$ and the function $f^x = \hat{x}$ is the m -variate (multilinear) LDE of x ,¹⁵ an evaluation $\hat{x}(\rho)$ of which can be computed incrementally as values of x are revealed in the stream. Then $\hat{x}(\rho) = \hat{x}_{|L}(\rho)$ allows the verifier to check that the prover is being honest (i.e., that the polynomial it sent is $\hat{x}_{|L}$), as well as to learn $x_j = \hat{x}(j) = \hat{x}_{|L}(0)$.

Observe that pep is *not* zero knowledge: the verifier learns all of $f_{|L}^x$, which it is not be able to construct by virtue of only learning β (and thus L) *after* streaming x . Note, however, that the *honest* verifier only inspects two evaluations of $f_{|L}^x$, namely, at 0 and ρ . In the following sections we construct a commitment protocol that lets the prover only reveal information about these two points, without sacrificing soundness.

¹³ Recall that the line L and $f_{|L}^x$ are sent in a canonical form: L as the evaluation $L(1)$ and $f_{|L}^x$ as the vector $(f^x \circ L(i) : i \in [dm])$. (There is no need to send $L(0) = \beta$ or $f_{|L}^x(0) = f^x(\beta) = \alpha$, as they are known to V .)

¹⁴ Note that the Lagrange polynomials in this case satisfy $\chi_i(j) = \mathbb{1}[i = j]$ for all $0 \leq i, j \leq dm$.

¹⁵ The space complexity can be reduced to $O(\log n)$ with the choice of parameters for q , d and m in Corollary 39.

Input: explicit access to $p, d, m, q \in \mathbb{N}$ with $p \leq d^m$, $q > d$ and $\mathbb{F} = \mathbb{F}_q$. Streaming access to $y \sim \mathbb{F}^p$ followed by a correction $\gamma \in \mathbb{F}$ and a coordinate $k \sim [p]$.

V: Sample $\rho \sim \mathbb{F}^m$ and compute $\hat{y}(\rho) = \sum_{i=1}^p \chi_i(\rho) y_i$ while streaming y .
Store ρ, k, γ and $\hat{y}(\rho)$.

■ **Figure 4** Protocol $\text{commit}(\alpha)$.

5.2 A prover-to-verifier commitment protocol

Our commitment protocol, designed to allow an unbounded-space sender to commit to a streaming receiver, directly uses the (average-case) hardness of the INDEX problem. By sending a message hidden at a random coordinate, we exploit the fact that any streaming algorithm requires a linear amount of space to be able to recall a random item from a string after it has been seen. We begin by formally defining (the search and decision versions of) INDEX in the one-way communication complexity model.

► **Definition 19.** SEARCH-INDEX, over alphabet Γ and with message length s , is the one-way communication problem defined as follows: Alice receives a string $x \in \Gamma^n$ and sends Bob an s -bit message $a = A(x)$. Bob receives, besides $a \in \{0, 1\}^s$, an index $j \in [n]$, and outputs a symbol $b = B(a, j) \in \Gamma$. The execution succeeds if $b = x_j$.

► **Definition 20.** DECISION-INDEX(α) (with alphabet Γ and message length s) is the one-way communication problem defined as follows: Alice receives a string $x \in \Gamma^n$ and sends Bob an s -bit message $a = A(x)$. Bob receives, besides Alice's message, an index $j \in [n]$, and outputs a bit $b = B(a, j) \in \{0, 1\}$. The execution succeeds if $b = 1$ when $x_j = \alpha$, and $b = 0$ otherwise.

It is well known that INDEX is extremely hard, even *on average* and in the one-way communication model *with shared randomness*.

► **Proposition 21.** Any one-way communication protocol (A, B) for SEARCH-INDEX that sends a message of length s satisfies

$$\mathbb{P}_{\substack{x \sim \Gamma^n \\ j \sim [n]}} [B(A(x), j) = x_j] = \frac{1}{|\Gamma|} + O\left(\sqrt{\frac{s}{p}}\right).$$

In other words, the chance of correctly recalling a random symbol is at best slightly better than uniform guessing if the string p is much longer than the message length s of the protocol. We note that this bound was known for $\Gamma = \{0, 1\}$ [49], but it extends to larger alphabets (we provide a proof of this fact in Appendix A.1 for completeness).

The commitment phase of our scheme exploits this hardness result directly: we take $\Gamma \hookrightarrow \mathbb{F}$ where \mathbb{F} is a large enough finite field (which will allow us to use `pep` to decommit) and have P send the triple $(y, \alpha - y_k, k)$ for random y and k as a commitment to α . (In particular, the commitment key is a random string-coordinate pair (y, k)). Loosely speaking, the protocol has the sender communicate a random stream y with the message hidden at a random coordinate k , which is revealed after y .

The honest verifier keeps a (random) fingerprint of y , which it can use to validate the message at y_k (see Figure 4), while the `decommit` stage simply instantiates `pep` appropriately (see Figure 5). We note that the inputs listed in the description of the protocols are those available to the verifier.

Now, we show that Figures 4 and 5 form a streaming commitment protocol, i.e., they satisfy the hiding and binding properties of Definition 17 if p is large enough; these follow from the hardness of SEARCH-INDEX and the soundness of `pep`, respectively.

Input: $\alpha \in \mathbb{F}$, as well as the (parameters and) values stored in the **commit** stage: $k, \gamma, \rho, \hat{y}(\rho)$.

V: Compute and send the line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ such that $L(0) = k$ and $L(\rho) = \rho$ with $\rho \sim \mathbb{F}$.

P: Send $\hat{y}_{|L}$.

V: Compute $g(\rho)$ and $g(0)$, where $g : \mathbb{F} \rightarrow \mathbb{F}$ is the degree- dm extension of the sequence of evaluations sent by P .

Accept if $g(\rho) = \hat{y}(\rho)$ and $g(0) + \gamma = \alpha$, rejecting otherwise.

■ **Figure 5** Protocol $\text{decommit}(\alpha, y, k)$.

► **Theorem 22.** *Figures 4 and 5 form a streaming commitment protocol with space complexity $s = O(m \log q)$ when $p = q^3$ and $dm = \text{polylog}(q)$. The protocol is secure against $\text{poly}(s)$ -space adversaries and communicates $O(q^3 \log q)$ bits.*

Proof. First, note that the communication complexity is dominated by the prover sending $p = q^3$ field elements in the **commit** step, for a total of $O(q^3 \log q)$ bits.

The binding property is an immediate consequence of the completeness and soundness of **pep**: if P is honest, i.e., sends the correction $\gamma = \alpha - y_k$ in the **commit** stage and the polynomial $\hat{y}_{|L}$ in the **decommit** stage, then V accepts, as $\hat{y}_{|L}(\rho) = \hat{y}(\rho)$ and $\hat{y}_{|L}(0) + \gamma = \alpha$. (Recall that the line L satisfies $L(0) = k$ and $L(\rho) = \rho$.)

Now, suppose the prover replies with a polynomial g such that $g(0) \neq y_k = \hat{y}(k) = \hat{y}_{|L}(0)$; then the Schwartz-Zippel lemma (Lemma 18) implies $\hat{y}(\rho) = \hat{y}_{|L}(\rho) \neq g(\rho)$ except with probability $dm/q = o(1)$, in which case V rejects.¹⁶ Note that the verifier only needs to store $\rho \in \mathbb{F}^m$, $k \in [p]$ and a constant number of additional field elements, for a space complexity of $O(m \log q + \log p) = O(m \log q)$.

To show the hiding property, assume towards contradiction that there exists a streaming algorithm D with space $\text{poly}(s) = \text{polylog}(q)$ that distinguishes commitments between some $\alpha \in \mathbb{F}$ and $\alpha' \in \mathbb{F} \setminus \{\alpha\}$ with constant bias:¹⁷ that is,

$$\mathbb{P}_{\substack{y \sim \mathbb{F}^p \\ k \sim [p]}} [D(y, k, \alpha - y_k) \text{ accepts}] - \mathbb{P}_{\substack{y \sim \mathbb{F}^p \\ k \sim [p]}} [D(y, k, \alpha' - y_k) \text{ accepts}] \geq \varepsilon$$

for some $\varepsilon = \Omega(1)$. Now consider the following algorithm A for **SEARCH-INDEX** over the alphabet \mathbb{F} with input (x, j) : simulate D on the stream (x, γ, j) where $\gamma \sim \mathbb{F}$; output $\alpha - \gamma$ if D accepts, and otherwise output $\alpha' - \gamma$. Note that A outputs correctly exactly when $\gamma = \alpha - y_k$ and D accepts, or $\gamma = \alpha' - y_k$ and D rejects; moreover, A can simulate D with constant space overhead, so that its space complexity is also $\text{polylog}(q)$. We will now show that A solves **SEARCH-INDEX** with a bias that is too large, contradicting Proposition 21.

$$\begin{aligned} \mathbb{P}_{\substack{x \sim \mathbb{F}^p \\ j \sim [p]}} [A(x, j) = x_j] &= \frac{1}{q} \cdot \mathbb{P}_{\substack{x \sim \mathbb{F}^p \\ j \sim [p]}} [D(x, j, \alpha - x_j) \text{ accepts}] + \frac{1}{q} \cdot \mathbb{P}_{\substack{x \sim \mathbb{F}^p \\ j \sim [p]}} [D(x, j, \alpha' - x_j) \text{ rejects}] \\ &= \frac{1}{q} \left(1 + \mathbb{P}_{\substack{x \sim \mathbb{F}^p \\ j \sim [p]}} [D(x, j, \alpha - x_j) \text{ accepts}] - \mathbb{P}_{\substack{x \sim \mathbb{F}^p \\ j \sim [p]}} [D(x, j, \alpha' - x_j) \text{ accepts}] \right) \\ &\geq \frac{1 + \varepsilon}{q} \\ &= \frac{1}{q} + \Omega\left(\frac{1}{q}\right). \end{aligned}$$

Since $1/q = \sqrt{q/p} = \omega\left(\sqrt{\text{poly}(s)/p}\right)$, owing to $s = \text{polylog}(q)$, the result follows. ◀

¹⁶We remark that ρ need not be sampled from the entire field; the same result holds if $\rho \sim R \subset \mathbb{F}$ when R is large enough. This will be useful in proving that our protocols for **PEP** and **SUMCHECK** are zero-knowledge.

¹⁷Note that allowing $\text{poly}(s)$ space for D will imply a space-robust indistinguishability property; bounding it by, say, $\tilde{O}(s)$ or $O(s^2)$ would prove a weaker but still nontrivial statement.

► Remark 23. Just as in pep, the verifier learns much more than the message $\hat{y}_{|L}(0) = \alpha \in \mathbb{F}$: it learns all of $\hat{y}_{|L}$. Crucially, however, the additional information consists of *random field elements uncorrelated with α* . This enables the commitment protocol laid out in this section to be proven zero-knowledge when the simulator has read-only access to a large random string t , as in Definition 13. (More accurately, such a simulator can perfectly generate the random variable that corresponds to the view resulting from the commit followed by the decommit steps.)

Indeed, a simulator with space $O(m \log q)$ and query access to $y \sim \mathbb{F}^p$ may sample $k \sim [p]$ and send $(y, \alpha - y_k, k)$ in the **commit** step; then, in **decommit**, after receiving the line L , it computes and sends $\hat{y}_{|L} = (\hat{y}_{|L}(i) : i \in \{0\} \cup [dm])$ by reading the string y an additional $dm + 1$ times, computing and sending one LDE evaluation at a time.

However, this basic commitment protocol is not yet sufficient. As discussed in Section 2.2, it allows P to commit (and decommit) to a single field element; but the prover should be able to commit to a polynomial and decommit to a single evaluation thereof. In the next section we show how to accomplish this, by modifying our scheme to make it *algebraic*.

5.3 Making the commitment algebraic

In this section, we will show how to modify the commitment protocol laid out in Section 5.2 so that the prover can commit to ℓ messages and decommit to a *single linear combination* of the verifier's choosing. As we shall see, this can in fact be accomplished by adapting only the commitment step.

The idea behind this new protocol is simple, but has an important caveat. If the prover P wishes to commit to the messages $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_\ell)$, the obvious solution is to send $(y_i, \alpha_i - y_{ik}, k_i)$ for all i , a sequence of commitments to each α_i . However, the indices k_i where each message is hidden are sampled independently, so that even though taking low-degree extensions is a linear operation (i.e., the LDE of $\sum_i \beta_i y_i$ is $\sum \beta_i \hat{y}_i$), a linear combination of the y_i does not yield a commitment to a linear combination of the α_i : evaluating it at k_i yields a sum where only the i^{th} summand is guaranteed to be correct.

We can fix this problem by hiding all the messages at the same coordinate k . Then, setting $\gamma = (\alpha_i - y_{ik} : i \in [\ell])$ and $\gamma = \beta \cdot \gamma = \sum \beta_i \gamma_i$, we have

$$\gamma + \left(\sum \beta_i y_i \right)_k = \sum \beta_i (y_{ik} + \gamma_i) = \alpha \cdot \beta;$$

so a linear combination of commitments yields a commitment to a linear combination of the messages. Therefore, the prover may send $(y_1, \dots, y_\ell, \gamma, k)$ and the new protocol will satisfy the binding property (a slightly stronger version of which, with respect to a random β , will be necessary; we elaborate upon this later in the section).

More precisely, viewing $y \in \mathbb{F}^{\ell \times p}$ as a matrix whose i^{th} row is y_i , the prover may send y , say, column by column.¹⁸ The resulting string, appended with γ and k , is a random INDEX instance whose alphabet is \mathbb{F}^ℓ ; and this enables us to show the hiding property for algebraic-commit as we did for commit.

¹⁸We remark that while sending y column by column naturally corresponds to an INDEX instance with a larger alphabet (where symbols are ℓ -tuples of field elements), since the hardness of INDEX holds for the stronger model of one-way communication protocols, the hiding property of the scheme is preserved regardless of the order in which y is sent. This is important in our sumcheck protocol, where a column cannot be sent all at once.

Input: explicit access to $p, m, d, q \in \mathbb{N}$ with $p \leq d^m$, $q > d$ and $\mathbb{F} = \mathbb{F}_q$; as well as linear coefficients $\beta \in \mathbb{F}^\ell$. Streaming access to $y \in \mathbb{F}^{\ell \times p}$ followed by $\gamma \in \mathbb{F}^\ell$ and $k \in [p]$.

V : Sample $\rho \sim \mathbb{F}^m$ and compute $\hat{y}(\rho, \beta) = \sum_{i=1}^{\ell} \beta_i \hat{y}_i(\rho)$, a random linear fingerprint of y with coefficients β , while streaming y .

Store $\rho, k, \hat{y}(\rho, \beta)$ and the correction $\gamma = \sum_{i=1}^{\ell} \beta_i \gamma_i$.

■ **Figure 6** Protocol algebraic-commit(α).

The result is Figure 6, which enables a prover to commit to multiple messages and decommit (via Figure 5, using $\hat{y}(\rho, \beta)$ as the fingerprint and $\beta \cdot \gamma$ as the correction) to an arbitrary linear combination of them.

► **Theorem 24.** *Figure 6 (algebraic-commit) and Figure 5 (decommit) form a streaming commitment protocol with space complexity $s = O((\ell + m) \log q)$ if $p = q^{3\ell}$ and $dm = \text{polylog}(q)$. The scheme is secure against $\text{poly}(s)$ -space adversaries and communicates $O(\ell q^{3\ell} \log q)$ bits.*

Furthermore, if each linear coefficient can be computed in $O(m \log q)$ space, then $s = O(m \log q)$.

Since the proof is a straightforward extension of Theorem 22, we defer it to Appendix A.2.

We stress that the binding property of the linear commitment protocol has an important caveat: it is with respect to *the linear combination* $\alpha \cdot \beta$, rather than the entire tuple α . Therefore, if the prover has knowledge of the linear coefficients, it can easily commit to a set of messages $\alpha' \neq \alpha$ that nonetheless decommits to the same linear combination $\alpha \cdot \beta$, and P has many choices indeed: the equation $\sum \beta_i \alpha'_i = \sum \beta_i \alpha_i$ is satisfied by all β in the hyperplane (of size $q^{\ell-1}$) orthogonal to $\alpha' - \alpha$.

Since our applications require a stronger guarantee – that V should be able to detect when P commits to α and a decommits according to $\alpha' \neq \alpha$ – this binding property is insufficient unless V chooses the coefficients β *at random*; then the linear combination of α' matches that of α only with probability $1/q$. While in our zero-knowledge protocol for PEP the coefficients are not *uniform*, they are a random evaluation of low-degree polynomials, and the same reasoning holds with a small loss (see Theorem 35).

However, an important issue still remains: the exponential dependency of Theorem 24 in the number ℓ of field elements that comprise the tuple P commits and decommits to. Concretely, in our applications we have $\ell = \omega(1)$ but can only afford to communicate $\text{poly}(q)$ bits. To circumvent this issue, we shall use the following efficient reduction from INDEX over bits to the problem of distinguishing a commitment to a fixed element of \mathbb{F}^ℓ from a commitment to a random one.

► **Lemma 25.** *Let (A, B) be a one-way protocol with s -bit messages that distinguishes between a length- p algebraic commitment to a fixed $\alpha \in \mathbb{F}^\ell$ and a random commitment with advantage ε ; that is, such that*

$$\left| \mathbb{P}_{\substack{y \sim \mathbb{F}^{\ell \times p} \\ k \sim [p]}} [B(A(y), (\alpha_i \oplus y_{ik} : i \in [\ell]), k) \text{ accepts}] - \mathbb{P}_{\substack{y \sim \mathbb{F}^{\ell \times p} \\ k \sim [p] \\ \tau \sim \mathbb{F}^\ell}} [B(A(y), \tau, k) \text{ accepts}] \right| = \varepsilon.$$

Then there exists an average-case one-way communication protocol for (binary) INDEX over p -bit strings that communicates $O(\ell^2 s \log^2 q / \varepsilon^2)$ bits and succeeds with probability $1 - \frac{1}{e} = \frac{1}{2} + \Omega(1)$.

Proof. Define, for ease of notation, $y^{(k)} := (y_{ik} : i \in [\ell])$ (i.e., the k^{th} column of y) and

$$a_{\tau} := \mathbb{P} \left[B \left(A(y), \tau \oplus y^{(k)}, k \right) \text{ accepts} \right] = \mathbb{E} \left[B \left(A(y), \tau \oplus y^{(k)}, k \right) \right],$$

where we interpret Bob's output as 1 (respectively 0) when he accepts (respectively rejects). Define, also, $\varepsilon_{\tau} := a_{\alpha} - a_{\tau}$.

We first argue that, without loss of generality, we can assume $\mathbb{F} = \mathbb{F}_2 = \{0, 1\}$. Note that, with $q = |\mathbb{F}|$,¹⁹

$$\varepsilon = a_{\alpha} - \frac{1}{q^{\ell}} \sum_{\tau \in \mathbb{F}^{\ell}} a_{\tau} = \frac{1}{q^{\ell}} \sum_{\tau \in \mathbb{F}^{\ell}} \varepsilon_{\tau}.$$

Taking $\ell' := \lfloor \ell \log q \rfloor$ and $S \subseteq \mathbb{F}^{\ell}$ as the set of size $2^{\ell'}$ containing α and the tuples τ with the largest ε_{τ} , and viewing $\{0, 1\}^{\ell'} \subseteq \mathbb{F}^{\ell}$ via a bijection between $\{0, 1\}^{\ell'}$ and S , we have

$$\varepsilon' := \frac{1}{2^{\ell'}} \sum_{\tau \in \{0, 1\}^{\ell'}} \varepsilon_{\tau} \geq \frac{\varepsilon}{3},$$

owing to $|S| \geq q^{\ell}/2$ and $\varepsilon_{\tau} \geq \varepsilon_{\tau'}$ when $\tau \in S \setminus \{\alpha\}$ and $\tau' \in \mathbb{F}^{\ell} \setminus S$. Therefore, assuming $\mathbb{F} = \mathbb{F}_2$ incurs at most a constant factor in ε and a $\log q$ factor in ℓ ; we shall use ε and ℓ (rather than ε' and ℓ') hereafter for simplicity of notation.

Finally, define, for each $0 \leq i < \ell$,

$$\varepsilon_i := \frac{1}{2^{\ell-i}} \sum_{\substack{\tau \in \{0, 1\}^{\ell} \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} \varepsilon_{\tau}.$$

We divide the analysis into two cases: suppose, first, that $\varepsilon_i \geq \varepsilon_{i-1} \cdot \left(1 - \frac{1}{2\ell}\right)$ for all $i \in [\ell-1]$. Then, by Bernoulli's inequality ($t \leq -1$ implies $(1+t)^{\ell} \geq 1+t\ell$), we have

$$\varepsilon_{\ell-1} = \frac{1}{2} (a_{\alpha} - a_{\alpha^{\oplus \ell}}) \geq \left(1 - \frac{1}{2\ell}\right)^{\ell} \cdot \varepsilon \geq \frac{\varepsilon}{2},$$

where $\alpha^{\oplus i} = (\alpha_1, \dots, \alpha_{i-1}, 1 - \alpha_i, \alpha_{i+1}, \dots, \alpha_{\ell})$. Consider the following one-way protocol (with shared randomness) for an INDEX instance $(x, j) \in \{0, 1\}^p \times [p]$: Alice and Bob jointly sample $2/\varepsilon^2$ independent matrices $y' \sim \{0, 1\}^{\ell \times p}$ and permutations $\sigma \sim S_p$; Alice sets $y_i = y'_i \oplus \mathbb{1}[i = \ell] \cdot \sigma(x)$ (where $\sigma(x)_k := x_{\sigma(k)}$), simulates $A(y)$ and sends the resulting messages in a $2s/\varepsilon^2$ -bit string to Bob.

With knowledge of j , Bob finishes the simulations $B(A(y), \gamma, k)$, using coordinate $k = \sigma^{-1}(j)$ and correction $\gamma = \alpha \oplus y^{(k)}$; he computes their empirical mean μ , outputs α_{ℓ} if $\mu \geq a_{\alpha} + \varepsilon/2$, and outputs $1 - \alpha_{\ell}$ otherwise.

Correctness follows from the observation that, if $x_j = \sigma(x)_k = \alpha_{\ell}$, then $\gamma = \alpha \oplus y^{(k)}$, so $\mathbb{E}[\mu] = a_{\alpha}$; since the (y, k) pairs are uniform and independent,

$$\mathbb{P} \left[\mu \leq a_{\alpha} - \frac{\varepsilon}{2} \right] \leq \frac{1}{e}$$

by the Chernoff-Hoeffding bound (Lemma 9, with $2/\varepsilon^2$ samples and $\delta = \varepsilon/2$). Likewise, when $x_j = 1$ we have $\alpha = \alpha^{\oplus \ell} \oplus y^{(k)}$; then $\mathbb{E}[\mu] = a_{\alpha^{\oplus \ell}} \leq a_{\alpha} - \varepsilon$ and an application of the Chernoff-Hoeffding bound (with the same parameters) yields the same guarantee.

¹⁹This assumes the acceptance probability of a commitment to α is larger than that of a random commitment, which is without loss of generality (otherwise Bob can simply flip his output bit).

We now consider the second case: suppose $\varepsilon_i < \varepsilon_{i-1} \cdot (1 - \frac{1}{2\ell})$ for some $i \in [\ell - 1]$; we take, without loss of generality, the minimal such i . Then

$$\begin{aligned} \frac{1}{2^{\ell-i}} \sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} \varepsilon_{\tau^{\oplus i}} &= \frac{1}{2^{\ell-i}} \sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' < i, \tau_{i'} = \alpha_{i'} \\ \tau_i = 1 - \alpha_i}} \varepsilon_\tau \\ &= 2\varepsilon_{i-1} - \varepsilon_i \\ &> \varepsilon_{i-1} \left(1 + \frac{1}{2\ell}\right), \end{aligned}$$

and thus

$$\begin{aligned} \frac{1}{2^{\ell-i}} \left(\sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} (\varepsilon_{\tau^{\oplus i}} - \varepsilon_\tau) \right) &= \frac{1}{2^{\ell-i}} \left(\sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} (a_{\tau^{\oplus i}} - a_\tau) \right) \\ &> \frac{\varepsilon_{i-1}}{\ell} \\ &\geq \frac{\varepsilon}{2\ell}. \end{aligned}$$

We will use a similar strategy to the previous case, although the expression we must estimate involves many more terms (indeed, $2^{\ell-i+1}$ of them). Consider the following one-way protocol for an INDEX instance $(x, j) \in \{0, 1\}^p \times [p]$: Alice and Bob jointly sample $64\ell^2/\varepsilon^2$ independent matrices $y' \sim \{0, 1\}^{\ell \times p}$ and permutations $\sigma \sim S_p$; Alice sets $y_{i'} = y'_{i'} \oplus \mathbb{1}[i' = i] \cdot \sigma(x)$, computes and sends all messages $A(y)$ in a $64\ell^2 s/\varepsilon^2$ -bit string to Bob.²⁰ (Recall that assuming $\mathbb{F} = \{0, 1\}$ incurs constant and logarithmic factors in ε and ℓ , respectively, so that Alice's message is $O(\ell^2 s \log^2 q/\varepsilon^2)$ bits long.)

For each $A(y)$ sent by Alice, Bob simulates $B(A(y), \tau \oplus y^{(k)}, k)$ with $k = \sigma^{-1}(j)$ for all τ satisfying $\tau_{i'} = \alpha_{i'}$ when $i' \leq i$. He computes the empirical mean μ of

$$\frac{1}{2^{\ell-i}} \left(\sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} \left(B(A(y), \tau^{\oplus i} \oplus y^{(k)}, k) - B(A(y), \tau \oplus y^{(k)}, k) \right) \right),$$

outputs 0 if the result is non-negative, and outputs 1 otherwise.

To prove correctness, first note that

$$\tau \oplus y^{(k)} = \begin{cases} \tau \oplus y^{(k)}, & \text{when } x_j = 0 \\ \tau^{\oplus i} \oplus y^{(k)}, & \text{when } x_j = 1, \end{cases}$$

so that, when $x_j = 0$,

$$\mathbb{E}[\mu] = \frac{1}{2^{\ell-i}} \left(\sum_{\substack{\tau \in \{0,1\}^\ell \\ \forall i' \leq i, \tau_{i'} = \alpha_{i'}}} (a_{\tau^{\oplus i}} - a_\tau) \right) > \frac{\varepsilon}{2\ell},$$

²⁰ Note that the only difference in Alice's strategy, as compared to the previous case, is the row where she inserts $\sigma(x)$ and the number of simulations of A .

and when $x_j = 1$ we have $\mathbb{E}[\mu] < -\varepsilon/2\ell$ (since the order of each pair of terms in the sum is flipped).

We conclude with an application of Hoeffding's inequality (Lemma 10, with $a = -1$, $b = 1$, $\delta = 1/2$ and $64\ell^2/\varepsilon^2$ samples): in the $x_j = 0$ case,

$$\mathbb{P}\left[\mu \leq \frac{\varepsilon}{4\ell}\right] \leq \frac{1}{e};$$

and, likewise, in the $x_j = 1$ case we have $\mathbb{P}\left[\mu \geq -\frac{\varepsilon}{4\ell}\right] \leq \frac{1}{e}$. ◀

5.4 A verifier-to-prover temporal commitment

The goal of this section is to construct the second main component towards our streaming zero-knowledge protocols. While it is not formally a commitment protocol (as per Definition 17), it is useful to conceptualise it as V committing to its internal randomness *before* the input is streamed (hence *temporal*).

Roughly speaking, we would like to ensure that a malicious verifier cannot choose the point ρ at which it (allegedly) computes its fingerprint *after it sees the input* (x, β) , as that would allow it to learn more than $f^x(\beta)$. (For example, in the INDEX case it could claim that $\rho = j + 1$ and learn $\hat{x}(j + 1) = x_{j+1}$.) We will prove, in 3 steps, a lemma formalising the intuition that a space- s algorithm cannot remember the positions of significantly more than s elements, which will later enable the construction of a simulator. As in the case of algebraic commitments, we will in fact prove a stronger statement: that this holds not only in the case of streaming algorithms, but in the stronger model of one-way communication protocols.

We first define two variants of SEARCH-INDEX in the one-way communication complexity model, which we call RECONSTRUCT and PAIR (see Definitions 26 and 27). In RECONSTRUCT, Bob's task is to output the symbols at *every* coordinate of the input z (rather than receiving a single coordinate j and outputting only z_j , as in INDEX); in other words, Bob should reconstruct the input as best he can. In PAIR, as in SEARCH-INDEX, Bob's task is again to output the symbol at a single coordinate; but rather than receiving the index as part of the input, Bob is free to choose a coordinate-symbol pair (i, α) and succeeds if $\alpha = z_i$. (Note that in both RECONSTRUCT and PAIR, Bob does not receive any additional input besides Alice's message.)

Our first two steps are as follows. We first study RECONSTRUCT and show, in Lemma 28, that if Alice's message has s bits, Bob cannot reconstruct significantly more than s coordinates of the input. Then, in Lemma 29, we show how this bound for RECONSTRUCT implies a related bound for PAIR; more precisely, we prove that there exists a size- s set C of coordinates such that the probability Bob outputs a correct coordinate-symbol pair (i, z_i) where $i \notin C$ is arbitrarily small.

While Lemma 29 immediately implies an analogous statement for streaming algorithms, it is not yet enough for our purposes. The reason is that our verifier will read additional information, i.e., a fixed – but unknown – PEP instance (x, β) between reading a PAIR input and writing its output. While it is intuitively clear that this should not help the verifier in any way (as the PEP and PAIR instances are uncorrelated), we still require a slight extension of Lemma 29.

To this end we define, for each fixed string $x \in \Gamma^n$, a variant of PAIR that we call PAIR(x) (Definition 31). The only difference between this one-way communication problem and PAIR is that Bob receives the string x in addition to Alice's message a . In Theorem 33, we show that the existence of a set capturing most of the correct outputs of PAIR implies such a set C also exists for PAIR(x); crucially, C is determined by a and *does not depend on x* . This last result then immediately implies an analogous one for streaming algorithms.

Let us begin with the definitions:

► **Definition 26.** RECONSTRUCT is the following one-way communication problem: Alice receives a string $z \sim \Gamma^v$ and sends Bob an s -bit message a ; after receiving a , Bob outputs a string $b \in \Gamma^v$. The *score* of an execution is the number of matching coordinates between z and b , i.e., $|\{i \in [v] : b_i = z_i\}|$.

► **Definition 27.** Let PAIR denote the following one-way communication problem: Alice receives a string $z \sim \Gamma^v$ and sends Bob an s -bit message a ; after receiving a , Bob outputs a pair $(\alpha, i) \in \Gamma \times [v]$. The execution succeeds if $\alpha = z_i$.

Note that both are definitionally average-case problems, as z is sampled uniformly. We now proceed to the first step towards the goal of this section: a proof that, in our parameter settings of interest for $|\Gamma|$ and s (as functions of v), the expected score of any protocol for RECONSTRUCT is tightly constrained by the message length s .

► **Lemma 28.** Any one-way protocol for RECONSTRUCT with alphabet size $|\Gamma| = O(v/\log \log v)$, $|\Gamma| \geq 32v/\log \log v$ and message length s , where $\log v \leq s = \text{polylog}(v)$, achieves an expected score of at most $s + o(s)$.

Proof. By the minimax theorem, we may assume Alice’s and Bob’s strategies are both deterministic, so that there exists a set of messages $A \subseteq \{0, 1\}^s$ that partitions the set Γ^v of input strings by $\{P_a : a \in A\}$, where Bob outputs $b = b(a) \in \Gamma^v$ whenever $z \in P_a$.

Observe that Bob’s optimal strategy is to set b_i as the most frequent symbol at the i^{th} coordinate among the strings of P_a ; we can thus index the partition by $b \in B := \{b(a) : a \in A\}$, setting $P_b = P_{b(a)} = P_a$. (Note that while $\{P_b : b \in B\}$ may be a smaller partition than $\{P_a : a \in A\}$, the expected scores of the protocols induced by both partitions are the same.)

Define the random variable $M_b := \{i \in [v] : z_i = b_i\}$. For simplicity of notation, denote also $\gamma := |\Gamma|$. Note that the expected score of this one-way protocol is

$$\begin{aligned} \mathbb{E}_{z \sim \Gamma^v} \left[\sum_{b \in B} \mathbb{1}[z \in P_b] \cdot |M_b| \right] &= \sum_{b \in B} \mathbb{P}[z \in P_b] \cdot \mathbb{E}_{z \sim P_b} [|M_b|] \\ &= \sum_{\substack{b \in B \\ |P_b| \geq \frac{s}{v} \cdot \frac{\gamma^v}{2^s}}} \mathbb{P}[z \in P_b] \cdot \mathbb{E}_{z \sim P_b} [|M_b|] + \sum_{\substack{b \in B \\ |P_b| < \frac{s}{v} \cdot \frac{\gamma^v}{2^s}}} \mathbb{P}[z \in P_b] \cdot \mathbb{E}_{z \sim P_b} [|M_b|]. \end{aligned}$$

We bound the first term by the largest expectation, and the second by observing that the union of sets P_b with $|P_b| \leq \frac{s}{v} \cdot \frac{\gamma^v}{2^s}$ contain at most an s/v fraction of all length- v strings:

$$\begin{aligned} \mathbb{E}_{z \sim \Gamma^v} \left[\sum_{b \in B} \mathbb{1}[z \in P_b] \cdot |M_b| \right] &\leq \max_{\substack{b \in B \\ |P_b| \geq \frac{s}{v} \cdot \frac{\gamma^v}{2^s}}} \mathbb{E}_{z \sim P_b} [|M_b|] + \sum_{\substack{b \in B \\ |P_b| < \frac{s}{v} \cdot \frac{\gamma^v}{2^s}}} \mathbb{P}[z \in P_b] \cdot v \\ &\leq \max_{\substack{b \in B \\ |P_b| \geq \frac{s}{v} \cdot \frac{\gamma^v}{2^s}}} \mathbb{E}_{z \sim P_b} [|M_b|] + s. \end{aligned}$$

Let $\delta \in (0, 1)$ be such that the volume of Hamming balls of radius δ is $\mathcal{V} := \frac{s\gamma^v}{v \cdot 2^s} \leq \frac{s\gamma^v}{v^2}$. (Recall that $s \geq \log v$.) For any $b \in B$, the set P_b that maximises

$$\mathbb{E}_{z \sim P_b} [|M_b|] = |P_b|^{-1} \sum_{z \in P_b} |\{i \in [v] : z_i = b_i\}|$$

is $P_b = \mathcal{B}(b, \delta')$, the ball centered at b (whose radius δ' is determined by the equality $|\mathcal{B}(b, \delta')| = |P_b|$). Since $|P_b| \geq \mathcal{V}$ implies $\delta' \geq \delta$, we have

$$\frac{1}{|P_b|} \cdot \sum_{z \in \mathcal{B}(b, \delta')} |\{i \in [v] : z_i = b_i\}| \leq \frac{1}{\mathcal{V}} \cdot \sum_{z \in \mathcal{B}(b, \delta)} |\{i \in [v] : z_i = b_i\}|,$$

so it suffices to bound the right-hand side. (The inequality follows from the observation that the left-hand side is a weighted average between the right-hand side and the expectation over $z \sim \mathcal{B}(b, \delta') \setminus \mathcal{B}(b, \delta)$, which is smaller.)

Define $\varepsilon := 1 - \delta$. We aim to upper bound $E_{z \sim P_b} [|M_b|]$, and set as an intermediate goal to prove upper and lower bounds for ε . To this end, we will use the following standard approximations (see, e.g., [41]) for $H = H_2$ when σ (or $1 - \sigma$) is small:

$$H(\sigma) = H(1 - \sigma) \in \left[\sigma \log \frac{1}{\sigma}, \sigma \left(\log \frac{1}{\sigma} + \frac{2}{\ln 2} \right) \right] \quad (12)$$

We begin with the lower bound on ε , which uses the lower bound of Equation 12 and follows by showing that the volume of a ball with radius $1 - \frac{\log \gamma}{v \log \log \gamma}$ is larger than \mathcal{V} ; then $\delta < 1 - \frac{\log \gamma}{v \log \log \gamma}$, or, equivalently, $\varepsilon = 1 - \delta > \frac{\log \gamma}{v \log \log \gamma}$.

We have

$$\begin{aligned} & H_\gamma \left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) \\ &= \frac{\left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) \log(\gamma - 1) + H \left(1 - \frac{\log \gamma}{v \log \log \gamma} \right)}{\log \gamma} && \text{(by Equation 1)} \\ &= \frac{\left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) \log(\gamma - 1) + H \left(\frac{\log \gamma}{v \log \log \gamma} \right)}{\log \gamma} && \text{(by Equation 2)} \\ &\geq \frac{\left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) (\log \gamma + \log(1 - \frac{1}{\gamma}))}{\log \gamma} + \frac{\log \left(\frac{v \log \log \gamma}{\log \gamma} \right)}{v \log \log \gamma} && \text{(by Equation 12)} \\ &= 1 + \left(\frac{1}{\log \gamma} - \frac{1}{v \log \log \gamma} \right) \log \left(1 - \frac{1}{\gamma} \right) + \frac{\log \frac{v}{\gamma} + \log \log \log \gamma}{v \log \log \gamma} - \frac{1}{v} \\ &\geq 1 - \frac{1}{\gamma \ln 2} \left(1 + \frac{1}{\gamma} \right) \left(\frac{1}{\log \gamma} - \frac{1}{v \log \log \gamma} \right) + \frac{\log \frac{v}{\gamma} + \log \log \log \gamma}{v \log \log \gamma} - \frac{1}{v} && \text{(by Equation 3)} \\ &\geq 1 - \frac{1}{\gamma \ln 2} \left(1 + \frac{1}{\gamma} \right) \left(\frac{1}{\log \gamma} - \frac{1}{v \log \log \gamma} \right) - \frac{1}{v} \\ &\geq 1 - \frac{3}{2v}, \end{aligned}$$

where the second-to-last inequality uses $v \geq \gamma$; and the last inequality uses $\gamma = \Theta \left(\frac{v}{\log \log v} \right)$ to bound the first negative term to order $\Theta \left(\frac{\log \log v}{v \log v} \right)$, so the $1/v$ term dominates. Therefore,

$$\gamma^{H_\gamma \left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) v} \geq \gamma^v / \gamma^{3/2},$$

and thus, by Equation 4, the volume of a ball (centered at any point b) of radius $1 - \frac{\log \gamma}{v \log \log \gamma} = 1 - \frac{\text{polylog}(v)}{v}$ satisfies

$$\begin{aligned} \left| \mathcal{B} \left(b, 1 - \frac{\log \gamma}{v \log \log \gamma} \right) \right| &\geq \frac{\gamma^{H_\gamma \left(1 - \frac{\log \gamma}{v \log \log \gamma} \right) v}}{\sqrt{\log v}} \\ &\geq \frac{\gamma^v}{2^{\frac{3}{2} \log \gamma + \frac{1}{2} \log \log v}} \\ &\geq \frac{\gamma^v}{2^{\frac{7}{4} \log \gamma}}. \end{aligned}$$

Then

$$\begin{aligned} |\mathcal{B}(b, \delta)| = \mathcal{V} &= \frac{s\gamma^v}{v \cdot 2^s} \\ &\leq \frac{\gamma^v \text{polylog}(v)}{v^2} \\ &\leq \frac{\gamma^v}{2^{\frac{15}{8} \log v}} \\ &\leq \left| \mathcal{B} \left(b, 1 - \frac{\log \gamma}{v \log \log \gamma} \right) \right|, \end{aligned}$$

and we conclude that $\varepsilon = 1 - \delta > \frac{\log \gamma}{v \log \log \gamma}$.

We now proceed to the upper bound on ε , which will use the upper bound of Equation 12. Since $\gamma^{H_\gamma(\delta)v} \geq \mathcal{V} = \frac{s\gamma^v}{v \cdot 2^s}$ (Equation 4), taking the logarithm of both sides and using Equation 1 yields

$$\frac{(1 - \varepsilon) \log(\gamma - 1) + H(1 - \varepsilon)}{\log \gamma} = H_\gamma(1 - \varepsilon) = H_\gamma(\delta) \geq 1 - \frac{s + \log \frac{v}{s}}{v \log \gamma}. \quad (13)$$

Note that the right-hand side is $1 - o(1)$ because $s = o(v)$; then, δ is within $o(1)$ distance of the maximiser $1 - 1/\gamma = 1 - o(1)$ of H_γ , so that $\delta = 1 - o(1)$ and $\varepsilon = o(1)$.

This allows us to bound $H(\varepsilon) = H(1 - \varepsilon)$ from above via Equation 12, which, combined with Equation 13 (multiplied by $\log \gamma$), implies

$$(1 - \varepsilon) \log(\gamma - 1) + \varepsilon \log \frac{1}{\varepsilon} + \frac{2\varepsilon}{\ln 2} \geq \log \gamma - \frac{s + \log v - \log s}{v}.$$

Rearranging yields

$$\varepsilon \left(\log \varepsilon + \log \gamma + \log \left(1 - \frac{1}{\gamma} \right) - \frac{2}{\ln 2} \right) \leq \frac{s + \log v - \log s}{v} + \log \left(1 - \frac{1}{\gamma} \right).$$

The bounds $-\log(1 - 1/\gamma) = O(1/\gamma) = O(\log \log v/v)$ (Equation 3) and $s \geq \log v$ show that the right-hand side is $O(s/v)$; and Equation 3 along with $\log \gamma = \log v - \log \log \log v + \Theta(1) = (1 - o(1)) \log v$ implies the left-hand side is $\Omega(\varepsilon(\log \varepsilon + \log v))$. Therefore, the inequality above simplifies to

$$\varepsilon(\log \varepsilon + \log v) = O\left(\frac{s}{v}\right).$$

Now, if we had $\varepsilon = \Omega(s/v)$, then

$$\begin{aligned} \varepsilon(\log \varepsilon + \log v) &= \varepsilon(\log s - \log v + \log v + \Omega(1)) \\ &= \Omega(\varepsilon \log s) = \omega(s/v), \end{aligned}$$

a contradiction. We thus conclude that $\varepsilon = o(s/v)$ (and, in particular, that ε is both lower and upper bounded by $\text{polylog}(v)/v$).

Returning to the goal of bounding the expected score, we now show that most of the volume of a Hamming ball of radius δ is close to its boundary. More precisely, consider the volume \mathcal{V}' of a ball of radius $\delta' = 1 - 2\varepsilon$. As $\varepsilon = v^{-1} \text{polylog}(v)$, Equation 4 applies, giving $\mathcal{V}' \leq \gamma^{H_\gamma(1-2\varepsilon)}$ and

$$\mathcal{V} = \Omega\left(\frac{\gamma^{H_\gamma(1-\varepsilon)}}{\sqrt{\varepsilon v}}\right) = \Omega\left(\frac{\gamma^{H_\gamma(1-\varepsilon)}}{\sqrt{s}}\right),$$

so that

$$\frac{\mathcal{V}'}{\mathcal{V}} = O\left(\sqrt{s} \cdot \gamma^{-(H_\gamma(1-\varepsilon) - H_\gamma(1-2\varepsilon))v}\right).$$

We can bound the coefficient in the exponent as follows:

$$\begin{aligned} H_\gamma(1-\varepsilon) - H_\gamma(1-2\varepsilon) &= \frac{\varepsilon \log(\gamma-1) + H(\varepsilon) - H(2\varepsilon)}{\log \gamma} \\ &\geq \frac{\varepsilon}{\log \gamma} \left(\log(\gamma-1) + \log \frac{1}{\varepsilon} - 2 \log \frac{1}{2\varepsilon} - \frac{4}{\ln 2} \right) \quad (\text{by Equation 12}) \\ &= \frac{\varepsilon}{\log \gamma} \left(\log(\varepsilon\gamma) + \log \left(1 - \frac{1}{\gamma}\right) + 2 - \frac{4}{\ln 2} \right) \\ &\geq \frac{\varepsilon \log \log \gamma}{\log \gamma}, \end{aligned}$$

where the last inequality follows from $\varepsilon\gamma > \frac{\gamma \log \gamma}{v \log \log \gamma} = \Theta\left(\frac{\log \gamma}{\log^2 \log \gamma}\right)$ when the constant in $\Theta(\cdot)$ is large enough ($\gamma \geq 32v / \log \log v$ suffices, as $\log(1 - 1/\gamma) + 2 - 4/\ln 2 > -5$). Therefore,

$$\begin{aligned} \sqrt{s} \cdot \gamma^{-(H_\gamma(1-\varepsilon) - H_\gamma(1-2\varepsilon))v} &\leq \sqrt{s} \cdot \gamma^{-\frac{\varepsilon v \log \log \gamma}{\log \gamma}} \\ &= \sqrt{s} \cdot 2^{-\varepsilon v \log \log \gamma} \\ &< \sqrt{s} \cdot 2^{-\log \gamma} \\ &= \frac{\sqrt{s}}{\gamma} \\ &= \Theta\left(\frac{\sqrt{s} \log \log v}{v}\right) \\ &= o(s/v), \end{aligned}$$

where the last line is due to $\sqrt{s} \geq \sqrt{\log v} = \omega(\log \log v)$ and the strict inequality to $\varepsilon > \frac{\log \gamma}{v \log \log \gamma}$. Therefore, $\mathcal{V}'/\mathcal{V} = o(s/v)$, showing that the volume of a ball of radius $1 - \varepsilon$ is indeed concentrated in points of distance at least $1 - 2\varepsilon$.

Finally, we conclude that

$$\begin{aligned} \mathbb{E}_{z \sim \Gamma^v} \left[\sum_{b \in B} \mathbb{1}[z \in P_b] \cdot |M_b| \right] &\leq \max_{\substack{b \in B \\ |P_b| \geq \frac{s\gamma^v}{v \cdot 2^s}}} \mathbb{E}_{z \sim P_b} [|M_b|] + s \\ &\leq s + \frac{1}{\mathcal{V}} \cdot \sum_{z \in B(b, \delta)} |\{i \in [v] : z_i = b_i\}| \\ &\leq s + \frac{\mathcal{V}'}{\mathcal{V}} \cdot v + \left(1 - \frac{\mathcal{V}'}{\mathcal{V}}\right) \cdot 2\varepsilon v \\ &= s + o(s), \end{aligned}$$

as desired. ◀

At this stage, we have an upper bound on the expected score of any one-way communication protocol for RECONSTRUCT. The next step is to show that it implies a similar bound for the communication problem PAIR; indeed, it seems intuitively clear that RECONSTRUCT is no harder than PAIR, as it allows Bob to output an independent guess for each coordinate. We formalise this intuition in the following lemma.

► **Lemma 29.** *Any one-way protocol for PAIR with alphabet size $\frac{32v}{\log \log v} \leq |\Gamma| = O\left(\frac{v}{\log \log v}\right)$ and message length s , where $\log v \leq s = \text{polylog}(v)$, satisfies the following: there exists an event E (depending only on z) with $\mathbb{P}[E] = 1 - o(1)$ and a set C of size s (depending only on Alice’s message) such that*

$$\mathbb{P}[\text{Bob outputs } (z_i, i) \text{ with } i \notin C | E] = o(1).$$

Proof. We will first show how to construct a protocol for RECONSTRUCT given one for PAIR, and then use Lemma 28 to conclude; as in that lemma, we define $\{P_a\}$ as the partition induced by Alice’s messages $a = a(z) \in A$ (we can assume Alice to be deterministic, as before, by the minimax theorem; then a is a random variable determined by z).

Recall that in a protocol for PAIR, Bob’s output is a random variable $b(a) \in \Gamma \times [v]$;²¹ our goal is to construct, from this random variable, an entire string $y \in \Gamma^v$ and apply the expected score bound to it. For ease of notation, when the message a is fixed we write $b = (b_1, b_2) = b(a)$; note that b is independent of the conditional distribution $z \sim P_a$ of the input, since upon fixing a it is solely a function of Bob’s internal randomness. We will denote its distribution by $\mu = \mu(a)$, and the conditional distribution of b_2 when $b_1 = i$ by μ_i .

The (PAIR) protocol’s success probability, conditional on receiving a , is given by

$$\begin{aligned} \sum_{i=1}^v \mathbb{P}_{z \sim P_a} [b = (z_i, i)] &= \sum_{i=1}^v \mathbb{P}_{b \sim \mu} [b_2 = i] \cdot \mathbb{P}_{z \sim P_a} [b_1 = z_i | b_2 = i] \\ &= \sum_{i=1}^v \mathbb{P}_{b \sim \mu} [b_2 = i] \cdot \mathbb{P}_{z \sim P_a} [b_1 = z_i]. \end{aligned}$$

Define $y = y(a) \in \Gamma^v$ as the string whose i^{th} coordinate is the most frequent symbol at the i^{th} coordinate in P_a (as before, y is the best attempt at reconstructing the input z given to Alice). Now, consider the RECONSTRUCT protocol that outputs the string whose i^{th} coordinate is the random variable $b_1 \sim \mu_i$. Since, for each $i \in [v]$, the symbol $\alpha \in \Gamma$ maximising $\mathbb{P}_{z \sim P_a} [\alpha = z_i]$ is y_i , the expected score of the resulting protocol (conditioned on a) is

$$\begin{aligned} \sum_{i=1}^v \mathbb{P}_{z \sim P_a} [b_1 = z_i | b_2 = i] &= \sum_{i=1}^v \mathbb{P}_{z \sim P_a} [b_1 = z_i] \\ &= \sum_{i=1}^v \sum_{\alpha \in \Gamma} \mathbb{P}_{b_1 \sim \mu_i} [b_1 = \alpha] \cdot \mathbb{P}_{z \sim P_a} [\alpha = z_i] \\ &\leq \sum_{i=1}^v \mathbb{P}_{z \sim P_a} [y_i = z_i] \\ &= \mathbb{E}_{z \sim P_a} [|M_a|], \end{aligned}$$

where, as before, $M_a = \{i \in [v] : y_i = z_i\}$.

Recall that in Lemma 28 we showed that, as long as $|P_a| \geq \frac{s|\Gamma|^v}{v2^s}$, the above expectation is $o(s)$. To conclude, we will use the following claim, whose proof is deferred to Appendix A.3:

► **Claim 30.** Let $p, q \in [0, 1]^v$ be probability vectors and $t \leq v$ be an integer. There exists a set $C \subseteq [v]$ of size t such that $\sum_{i \in [v] \setminus C} p_i q_i \leq 1/t$.

²¹ Note that, in contrast with Alice, we cannot assume Bob is deterministic. We wish to bound the number of points in the support of b that aggregate all but a subconstant amount of probability weight in correct solutions to the problem. This is not a function of the *value* of b , but of its *distribution*, so the minimax principle does not apply.

Note that while $r \in [0, 1]^v$ defined by $r_i = \mathbb{P}[b_1 = z_i \mid b_2 = i]$ is not a probability vector, we may normalise it to obtain one: applying Claim 30 to $p = (\mathbb{P}[b_2 = i] : i \in [v])$, $q = r/\|r\|_1$ and $t = s$, we obtain a set $C_a \subset [v]$ of size s such that

$$\begin{aligned} \mathbb{P}_{\substack{z \sim P_a \\ b \sim \mu(a)}} [b = (z_i, i) \text{ with } i \notin C_a] &= \sum_{i \notin C_a}^v p_i r_i \\ &= \|r\|_1 \sum_{i \notin C_a}^v p_i q_i \\ &\leq \frac{\|r\|_1}{s} \\ &= \frac{\sum_{i=1}^v \mathbb{P}[b_1 = z_i \mid b_2 = i]}{s} \\ &= o(1) \end{aligned}$$

whenever $|P_a| \geq \frac{s|\Gamma|^v}{v2^s}$. Finally, take C_a as given by the claim. Recall that the sets P_a of size less than $\frac{s|\Gamma|^v}{v2^s}$ cover at most a $s/v = o(1)$ fraction of length- v strings, so that the probability $z \sim \Gamma^v$ falls into the union of such sets is $o(1)$. In the complement of this event, we have

$$\begin{aligned} &\mathbb{P} \left[b(a) = (z_i, i) \text{ with } i \notin C_a \mid |P_a| \geq \frac{s|\Gamma|^v}{v2^s} \right] \\ &= \frac{1}{\mathbb{P}_{z \sim \Gamma^v} \left[|P_a| \geq \frac{s|\Gamma|^v}{v2^s} \right]} \sum_{\substack{a \in A \\ |P_a| \geq \frac{s|\Gamma|^v}{v2^s}}} \mathbb{P}_{z \sim \Gamma^v} [z \in P_a] \cdot \mathbb{P}_{\substack{z \sim P_a \\ b \sim \mu(a)}} [b = (z_i, i) \text{ with } i \notin C_a] \\ &= \frac{1}{1 - o(1)} \cdot o(1) = o(1), \end{aligned}$$

which concludes the proof. \blacktriangleleft

With the second step of our proof finished, we already have a nontrivial result by the known implication from hardness for one-way communication complexity: any streaming algorithm that streams a uniformly random string $z \in \Gamma^v$ and immediately outputs a pair (α, i) has a small set $C \subset [v]$ capturing most of the probability that it outputs correctly. However, the verifier in our zero-knowledge streaming protocol will stream an INDEX instance between streaming z and outputting a pair. To capture this behaviour, we define a (slight) variant of PAIR and prove that the result of Lemma 29 carries over to it.

► **Definition 31.** For each string $x \in \Gamma^n$, let $\text{PAIR}(x)$ denote the following one-way communication problem: Alice receives a string $z \sim \Gamma^v$ and sends Bob an s -bit message a ; Bob reads x and a and outputs a pair $(\alpha, i) \in \Gamma \times [v]$. The protocol succeeds if $\alpha = z_i$.

We have now reached the end goal of this section:

► **Lemma 32.** Fix a (single) one-way communication protocol for $\text{PAIR}(x)$ for all $x \in \Gamma^n$ with alphabet size $32v/\log \log v \leq |\Gamma| = O(v/\log \log v)$ and message length $\log v \leq s = \text{polylog}(v)$. Then, for any $x \in \Gamma^n$, there exists an event E (that depends only on z) with $\mathbb{P}[E] = 1 - o(1)$ and a set C of size s (that depends only on Alice's message) satisfying

$$\mathbb{P}[b(a, x) = (z_i, i) \text{ with } i \notin C_a \mid E] = o(1).$$

Proof. We will make a small adaptation in one of the steps of Lemma 29 to show there is a size- s set C independent of x that captures most of the probability of Bob's correct outputs.

Following the notation of Lemma 29, $\{P_a\}$ is the partition induced by Alice’s messages and Bob’s output is a random variable $b(a(z), x) = b(a, x) \in \Gamma \times [v]$. We also denote the distribution of $b = b(a, x)$ by $\mu(a, x)$ and the conditional distribution of b_1 when $b_2 = i$ by $\mu_i(a, x)$.

For every x and a , the protocol’s success probability conditioned on $z \in P_a$ is

$$\begin{aligned} \sum_{i=1}^v \mathbb{P}_{\substack{z \sim P_a \\ b \sim \mu(a, x)}} [b = (z_i, i)] &= \sum_{i=1}^v \mathbb{P}_{b \sim \mu(a, x)} [b_2 = i] \cdot \mathbb{P}_{\substack{z \sim P_a \\ b \sim \mu(a, x)}} [b_1 = z_i \mid b_2 = i] \\ &= \sum_{i=1}^v \mathbb{P}_{b \sim \mu(a, x)} [b_2 = i] \cdot \mathbb{P}_{\substack{z \sim P_a \\ b_1 \sim \mu_i(a, x)}} [b_1 = z_i]. \end{aligned}$$

With $y = y(a) \in \Gamma^v$ as the string whose i^{th} coordinate is the most frequent symbol at the i^{th} coordinate in P_a , we know that $\mathbb{P}_{z \sim P_a} [\alpha = z_i]$ is maximal when $\alpha = y_i$. This holds also if α is a random variable (independent from z), so that, in particular, with $r \in [0, 1]^v$ defined by

$$r_i := \max_{x \in \Gamma^n} \left\{ \mathbb{P}_{\substack{z \sim P_a \\ b_1 \sim \mu_i(a, x)}} [b_1 = z_i] \right\} \leq \mathbb{P}_{z \sim P_a} [y_i = z_i]$$

we have $\|r\|_1 = o(s)$ when $|P_a|$ is sufficiently large. Defining $p \in [0, 1]^v$ by $p_i = \mathbb{P}_{b \sim \mu(a, x)} [b_2 = i]$, $p' \in [0, 1]^v$ by $q_i = r_i / \|r\|_1$ and using Claim 30, we obtain a set $C_a \subset [v]$ of size s such that for every $x \in \Gamma^n$,

$$\begin{aligned} \mathbb{P}_{\substack{z \sim P_a \\ b \sim \mu(a, x)}} [b = (z_i, i) \text{ and } i \notin C_a] &= \sum_{i=1}^v \mathbb{P}_{b \sim \mu(a, x)} [b_2 = i] \cdot \mathbb{P}_{\substack{z \sim P_a \\ b_1 \sim \mu_i(a, x)}} [b_1 = z_i] \\ &\leq \|r\|_1 \sum_{i \notin C_a} p_i q_i = o(1), \end{aligned}$$

and we conclude with same calculation of Lemma 29. ◀

As an immediate corollary (by taking C to be a set of symbol-coordinate pairs, rather than only coordinates; and setting, say, $C = \emptyset$ in the complement of the event E), we have:

► **Theorem 33.** *Let Γ be an alphabet of size $32v / \log \log v \leq |\Gamma| = \Theta(v / \log \log v)$ and fix $x \in \Gamma^n$. Let \tilde{V} be a streaming space- s algorithm with $\log v \leq s = \text{polylog}(v)$ that streams $z \sim \Gamma^v$ followed by x , and outputs a pair $(\alpha, i) \in \Gamma \times [v]$.*

There exists a set $C \subset \Gamma \times [v]$ of size s , determined by the snapshot of \tilde{V} at the end of the stream z , such that

$$\mathbb{P} \left[\tilde{V}(z, x) \text{ outputs } (z_i, i) \notin C \right] = o(1).$$

The theorem above attains what we set out for in this section: since \tilde{V} cannot remember many pairs (z_i, i) , we may prepend to any protocol a step where P sends z to the verifier. Then, whenever \tilde{V} sends an allegedly random $\alpha \in \Gamma$ to the prover, we ask that it *also* send the coordinate i such that $\alpha = z_i$ as evidence that α was indeed sampled in the past, i.e., before it finished streaming z . In other words, this step provides a *temporal commitment* by means of which \tilde{V} can show that its internal randomness is uncorrelated with the input.

6 A zero-knowledge SIP for polynomial evaluation

Our goal in this section will be to combine the components constructed in Sections 5.3 and 5.4 – *algebraic* and *temporal* commitment protocols – into a zero-knowledge protocol for PEP. It is useful to keep in mind that PEP is a generalisation of INDEX, and thus a protocol for the former yields one for the latter; in other words, for concreteness one may replace PEP by INDEX throughout this section. A formal definition of PEP follows.

► **Definition 34.** Let $\alpha \in \mathbb{F}$ and $f = \{f^x : x \in \Gamma^n\}$ be a mapping such that $f^x : \mathbb{F}^m \rightarrow \mathbb{F}$ is a degree- d polynomial. $\text{PEP}(f, \alpha)$ is the language $\{(x, \beta) \in \Gamma^n \times \mathbb{F}^m : f^x(\beta) = \alpha\}$.

We remark that the parameters of the problem generally increase as a function of n ; in particular, the field size is always assumed to satisfy $q = |\mathbb{F}| = \omega(1)$.

6.1 The protocol

For any mapping f and field element α , Figure 7 lays out $\text{zk-pep}(f, \alpha)$, our zero-knowledge SIP for $\text{PEP}(f, \alpha)$. Theorems 35 and 36 prove, respectively, the correctness (i.e., completeness and soundness) and the zero-knowledge properties of zk-pep .

The protocol uses commitment (sub)protocols to allow each party to only reveal key information after the other party gives evidence that it is being honest; this is achieved by interspersing the commit-decommit steps of one party with those of the other. More precisely, in the setup (Step 0) the verifier performs its (temporal) commitment; after the input is streamed (Step 1), the prover makes its (algebraic) commitment in Step 2. Then follow decommitments in the same order: verifier and prover decommit at Steps 3 and 4, respectively.

For ease of notation, we use \mathbb{F}^\times to denote $\mathbb{F} \setminus \{0\}$, the multiplicative group of the field \mathbb{F} . Recall, moreover, that for a matrix y , we use $\hat{y}(\rho, \theta) \in \mathbb{F}$ to denote an evaluation of the low-degree extension of the string $\theta \cdot y$ over \mathbb{F} (see Section 3), and that $\chi(\rho)$ denotes a vector of Lagrange polynomials (see Section 5.1); in the following protocol, the vector contains all but the first point of the interpolating set $\{0\} \cup [dm]$ for a univariate degree- dm polynomial over \mathbb{F} , i.e., $\chi(\rho) = (\chi_i(\rho) : i \in [dm]) \in \mathbb{F}^{dm}$.

6.2 Analysis of the protocol

We now show that zk-pep is a valid (i.e., complete and sound) streaming interactive proof, as well as compute its space and communication complexities.

► **Theorem 35.** Let f be such that an evaluation of the \mathbb{F}_q -polynomial f^x can be computed by streaming x in $O(m \log q)$ space. Then, for any $\alpha \in \mathbb{F}_q$, Figure 7 is an SIP for $\text{PEP}(f, \alpha)$ with $s = O(m \log q)$ space complexity. Its communication complexity is $O(q^m m \log^2 q)$ in the setup and $O(d^4 m^5 q^3 \log q)$ in the interactive phase.

Proof. We will prove completeness then soundness, and compute the complexities last.

Completeness. The verifier only aborts in Step 0 (the setup) if ρ is not among the $v > q^m \log \log q$ random tuples sent by the prover, an event with probability $(1 - 1/q^m)^v \leq e^{-v/q^m} = o(1)$. Otherwise, since the prover behaves honestly, in Step 2 (the algebraic commitment) we have

$$y_{ik} = f_{|L}^x(i) - \gamma_i$$

Input: Explicit access to \mathbb{F} , element $\alpha \in \mathbb{F}$, degree d , dimension m and a mapping $x \mapsto f^x$; streaming access to $x \in \Gamma^n$ followed by $\beta \in \mathbb{F}^m$.

Parameters:

Field size $q = |\mathbb{F}|$ satisfying $dm = o(q)$;

Commitment lengths $v = q^m(\log m + \log \log q)/32$ and $p = m(dm q)^3$;

Step 0: Temporal commitment

P: Send a string $z \sim (\mathbb{F}^m)^v$.

V: Sample $\rho \sim \mathbb{F}^m$ and stream z . For each i , check if $z_i = \rho$ and store $\ell := i$ if so. Reject if $\rho \neq z_i$ for all $i \in [v]$.

Step 1: Input streaming

V: Stream x and compute $f^x(\rho) \in \mathbb{F}$. Store $\beta \in \mathbb{F}^m$.

If $\rho = \beta$, check that $f^x(\rho) = \alpha$, accepting if so and rejecting otherwise.

Step 2: Algebraic commitment

V: Sample $\rho \sim \mathbb{F}^m \setminus [dm]$ and send the line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ with $L(0) = \beta$ and $L(\rho) = \rho$.

P: Send an algebraic commitment (y, γ, k) to $f_{|L}^x$, i.e., $(y, k) \sim \mathbb{F}^{dm \times p} \times [p]$ and $\gamma \in \mathbb{F}^{dm}$ with $\gamma_i = f_{|L}^x(i) - y_{ik}$ for all $i \in [dm]$.

V: Sample $\sigma \sim \mathbb{F}^m$ and, while streaming y , compute $\hat{y}(\sigma, \chi(\rho))$.

Compute the correction $\gamma = \chi(\rho) \cdot \gamma$ and save (the identification of) $k \in \mathbb{F}^m$.

Step 3: Temporal decommitment

V: Send ρ and ℓ .

P: Check that $z_\ell = \rho \in L$ and $\rho := L^{-1}(\rho) \notin \{0\} \cup [dm]$, aborting otherwise.

Step 4: Algebraic decommitment

V: Run $\text{decommit}(f^x(\rho) - \chi_0(\rho)\alpha, \chi(\rho) \cdot y, k)$, with correction γ and fingerprint $\hat{y}(\sigma, \chi(\rho))$.

Accept if decommit accepts and reject otherwise.

■ **Figure 7** Protocol $\text{zk-pep}(f, \alpha)$.

for all $i \in [dm]$.

Let $w = \chi(\rho) \cdot y = \sum_{i=1}^{dm} \chi_i(\rho) y_i \in \mathbb{F}^p$ and $\hat{w} : \mathbb{F}^m \rightarrow \mathbb{F}$ be its m -variate LDE. Recall that, in $\text{decommit}(f^x(\rho) - \chi_0(\rho)\alpha, w, k)$ (Figure 5), with correction γ and fingerprint $\hat{y}(\sigma, \chi(\rho))$, the verifier sends a line $L' : \mathbb{F} \rightarrow \mathbb{F}^m$ with $L'(0) = k$, $L'(\sigma) = \sigma$, receives $\hat{w}_{|L'}$ and makes two checks: that $\hat{w}_{|L'}(\sigma)$ matches the fingerprint and that $\hat{w}(0) + \gamma = f^x(\rho) - \chi_0(\rho)\alpha$. Since

$$\hat{w}_{|L'}(\sigma) = \hat{w}(\sigma) = \sum_{i=1}^{dm} \chi_i(\rho) \hat{y}_i(\sigma) = \hat{y}(\sigma, \chi(\rho))$$

and

$$\begin{aligned} \hat{w}(0) + \gamma &= w_k + \gamma = \sum_{i=1}^{dm} \chi_i(\rho) (y_{ik} + \gamma_i) \\ &= \sum_{i=1}^{dm} \chi_i(\rho) f_{|L}^x(i) \\ &= f^x(\rho) - \chi_0(\rho) f^x(\beta) \\ &= f^x(\rho) - \chi_0(\rho)\alpha, \end{aligned}$$

the verifier accepts when P is honest except with probability $o(1)$.

Soundness. First, note that if $\rho \notin \{z_i : i \in [v]\}$, the verifier rejects already in Step 0. We can thus assume the tuple ρ equals some coordinate in z , and, since the string and tuple are independent random variables, the distribution of ρ is still uniform conditioned on this event. (We may also assume that $\rho \neq \beta$, since otherwise V also rejects regardless of the prover's behaviour.)

The only other point where V may reject is Step 4 (the algebraic decommitment). Once again, recall that V sends the prover a line L' with $L'(0) = k$, $L'(\sigma) = \sigma$ where $\sigma \sim \mathbb{F}$ and P replies with a degree- dm polynomial $g : \mathbb{F} \rightarrow \mathbb{F}$ that is allegedly $\hat{w}_{|L'}$. The verifier then checks that $g(\sigma) = \hat{y}(\sigma, \chi(\rho)) = \hat{w}_{|L'}(\sigma)$ and $g(0) + \gamma = f^x(\rho) - \chi_0(\rho)\alpha$, rejecting if either equality fails to hold.

We now analyse three cases: first, suppose that $g = \hat{w}_{|L'}$. Then the first check passes but

$$\begin{aligned} g(0) + \gamma &= w_k + \gamma \\ &= f^x(\rho) - \chi_0(\rho)f^x(\beta) \\ &\neq f^x(\rho) - \chi_0(\rho)\alpha, \end{aligned}$$

so the verifier rejects (with probability 1).

Suppose, now, that $g(0) \neq \hat{w}_{|L'}(0)$. Then Lemma 18 (Schwartz-Zippel) implies $g(\sigma) \neq \hat{w}_{|L'}(\sigma)$, so the verifier rejects, except with probability $dm/q = o(1)$.

Finally, suppose that $g \neq \hat{w}_{|L'}$ but $g(0) = \hat{w}(0) = \sum_{i=1}^{dm} \chi_i(\rho)y_{ik}$. Then either the first check fails, i.e., $g(\sigma) \neq \hat{y}(\sigma, \chi(\rho))$, and V rejects; or $g(\sigma) = \hat{y}(\sigma, \chi(\rho))$, and the second check passes if

$$g(0) + \gamma = \sum_{i=1}^{dm} \chi_i(\rho)(y_{ik} + \gamma_i)$$

is equal to

$$f^x(\rho) - \chi_0(\rho)\alpha = \chi_0(\rho)(f^x(\beta) - \alpha) + \sum_{i=1}^{dm} \chi_i(\rho)f_{|L'}^x(i).$$

Rearranging, the second check corresponds to the following equation:

$$\chi_0(\rho)(f^x(\beta) - \alpha) + \sum_{i=1}^{dm} \chi_i(\rho) \left(f_{|L'}^x(i) - \gamma_i - y_{ik} \right) = 0.$$

Now, consider the left-hand side of the equation as a polynomial in ρ : plugging in 0 for the variable ρ evaluates to $f^x(\beta) - \alpha \neq 0$, so that it is a nonzero polynomial; and, crucially, ρ was sampled uniformly (from $\mathbb{F}^\times \setminus [dm]$) and independently of the communication (in particular, of y and γ) by V . By Lemma 18 lemma once again, the equation is satisfied with probability at most $dm/(q - dm - 1) = o(1)$ and soundness follows.

Communication complexity. Most of the communication occurs in Steps 0 and 2 (the commitments), which communicate

$$\begin{aligned} O(q^m(\log m + \log \log q)m \log q) &= O(q^m m \log^2 q) \quad \text{and} \\ O(pdm \log q) &= O(d^4 m^5 q^3 \log q) \end{aligned}$$

bits, respectively. (The communication in other steps is significantly smaller: Step 1 has none, while Steps 3 and 4 communicate $m \log q + \log v = O(m \log q)$ and $O(dm \log q)$ bits, respectively.)

Space complexity. Apart from a constant number of elements of \mathbb{F} (requiring $O(\log q)$ bits), the verifier stores $\ell \in [v]$, $k \in [p]$ and $\rho, \sigma \in \mathbb{F}^m$. Since $v \geq p$, the space complexity is dominated by ℓ and ρ, σ . Since storing ℓ requires $\log v = O(m \log q)$ bits (as does computing $f^x(\rho)$) while ρ and σ require $m \log q$ bits each, the space complexity follows. \blacktriangleleft

6.3 Zero-knowledge

Having shown that zk-pep is a valid streaming interactive proof, we now show it is also zero-knowledge.

► **Theorem 36.** *Figure 7 is zero-knowledge, secure against distinguishers with space $dm^2 \text{polylog}(q)$. The simulator runs in $O((d + m \log q)m \log q) = O(dm^2 \log^2 q)$ space.*

Proof. Recall that an SIP with a space- s verifier is zero-knowledge against $dm^2 \text{polylog}(q)$ -space distinguishers if there exists a streaming simulator S that satisfies the following. For any space- s (honest or malicious) verifier \tilde{V} and input (x, β) where $f^x(\beta) = \alpha$, given whitebox access to \tilde{V} the simulator S produces a view that is indistinguishable to a $dm^2 \text{polylog}(q)$ -space (streaming) algorithm from the view generated by an interaction of \tilde{V} with the honest prover. Note that \tilde{V} can be simulated in space $O(s)$, so the space complexity of the statement suffices to simulate the verifier of Figure 7 since $s = O(m \log q)$.

The simulator interprets its read-only random bit string as (z, y) with $z \sim \mathbb{F}^v$ and $y \sim \mathbb{F}^{dm \times p}$ (so that $vm \log q + pdm \log q \leq q^{m+8}$ bits suffice and an algorithm with $(m+8) \log q$ space can address into this string). This pair will be used to simulate prover messages, whereas the simulation of \tilde{V} will use a source of randomness that cannot be reread (but has unbounded length). In the description that follows, as well as the more succinct one in Figure 8, recall that \tilde{V} is assumed to only output a decision at the end of the protocol (so that, if it decides to reject in the middle, it continues the protocol with dummy messages); and likewise if S (or P) aborts.

In the setup, Step 0 (the temporal commitment), S simulates $\tilde{V}(z)$. Then, using the snapshot of the verifier’s memory and its whitebox access to \tilde{V} , the simulator finds the set C of s elements of \mathbb{F}^m that \tilde{V} may successfully decommit to with the largest probabilities. More precisely, S calls the whitebox oracle \mathcal{W} (see Definition 15) on the algorithm that corresponds to the verifier immediately before streaming x , with initial memory state equal to the current snapshot $b \in \{0, 1\}^s$, and whose output is a pair (ρ, ℓ) at Step 3 (ignoring L , the intermediate output at Step 2).

S initialises a(n empty) sorted list of message-probability pairs in $\mathbb{F}^m \times [v] \times [0, 1]$, and, for all $\ell \in [v]$, uses its oracle access to both z and \mathcal{W} to find $\mu_\ell := \mathcal{W}(b, (z_\ell, \ell))$. If the size of the list is smaller than s , or μ_ℓ is larger than the smallest probability in it, S adds (z_ℓ, ℓ, μ_ℓ) to it (and removes the tuple with the smallest $\mu_{\ell'}$ if the size of the resulting would have exceeded s).

This yields the set $C \subset \mathbb{F}^m \times [v]$ with the s most likely correct decommitments of \tilde{V} . Since the string z is over the alphabet \mathbb{F}^m , whose size satisfies

$$\frac{v}{\log \log v} = \frac{q^m (\log m + \log \log q)}{32 \log (m \log q + \log (\log m + \log \log q) - 5)} \leq \frac{q^m}{32},$$

$q^m = \Theta(v / \log \log v)$ as well as $s \geq \log p = \Theta(\log q)$ and $s = \text{polylog}(p)$, Theorem 33 applies for this parameter setting. This ensures that, except with probability $o(1)$, the verifier \tilde{V} will output either $(z_\ell, \ell) \in C$ or an incorrect (ρ, ℓ) with $z_\ell \neq \rho$ in its decommitment at Step 3.

Then S proceeds to Step 1, where it simulates $\tilde{V}(x)$ and, with $F := \{z_i : (z_i, i) \in C\}$, computes $f^x(\rho)$ for every $\rho \in F$. At the start of Step 2 (the algebraic commitment), \tilde{V} sends a line L . The simulator inspects the intersection of L (viewed as a set) with the set of

fingerprints F and computes a random degree- dm polynomial g subject to the constraints $g(\beta) = f_{|L}^x(\beta) = f^x(L(\beta))$ for all $\beta \in L^{-1}(F)$.²² Note that the description of g is comprised of $O(dm)$ field elements.

S samples $k \sim [p]$ then simulates \tilde{V} streaming y followed by $\gamma_i = g(i) - y_{ik}$ for all $i \in [dm]$ and k ; note that S is able to compute all γ_i from the description of g combined with its oracle access to y .

There is no prover-to-verifier communication in Step 3 (the temporal decommitment), so S simulates \tilde{V} until the verifier sends a tuple $\rho \in \mathbb{F}^m$ and an index $\ell \in [v]$. The simulator then checks that $z_\ell = \rho \in L$ and $\rho := L^{-1}(\rho) \in \mathbb{F}^\times \setminus [dm]$; if not, then S aborts (as P would).

Finally, in Step 4 (the algebraic decommitment), S simulates \tilde{V} until it sends a line $L' : \mathbb{F} \rightarrow \mathbb{F}^m$. The only remaining part of the verifier's view left to generate are the evaluations of the polynomial $\sum_{i \in [dm]} \chi_i(\rho) \hat{y}_i \circ L'$ for all points in $[dm]$. These are computed by S in a streaming fashion using its oracle access to y .

The space complexity of S is dominated by the description of the polynomial g , which requires $O(dm \log q)$ bits, and by the set C of $s = O(m \log q)$ elements of $\mathbb{F}^m \times [v]$. Since each element requires $m \log q + \log v = O(m \log q)$ bits, the total space complexity is

$$O(dm \log q + sm \log q) = O((d + m \log q)m \log q) = O(dm^2 \log^2 q),$$

as claimed. (Apart from C , the simulator stores $f^x(\rho) \in \mathbb{F}$ for every $\rho \in C$, which requires $s \log q$ bits; and the lines L, L' as well as k , which require $O(\log q)$ bits each.)

Now, all that remains is to prove indistinguishability by space- s' streaming algorithms between the output of S and a real transcript, for some s' comparable to the space complexities of the verifier and simulator. The following claim proves this with $s' = dm^2 \text{polylog}(q)$ (which is larger than both).

▷ **Claim 37.** Fix $\alpha \in \mathbb{F}$ and f as in the definition of PEP, an input $(x, \beta) \in \mathbb{F}^n \times \mathbb{F}^m$, a bit string r of arbitrary length and a $O(m \log q)$ -space verifier algorithm \tilde{V} . Let D be a streaming algorithm with space $dm^2 \text{polylog}(q)$ such that

$$\mathbb{P} \left[D \left(\text{View}_{P, \tilde{V}}(x, r) \right) \text{ accepts} \right] - \mathbb{P} \left[D \left(S \left(\tilde{V}, x, r \right) \right) \text{ accepts} \right] = \varepsilon,$$

with $\text{View}_{P, \tilde{V}}(x, r)$ a view of Figure 7 and $S(\tilde{V}, x, r)$ output by Figure 8. Then $\varepsilon = o(1)$.

Assume, towards contradiction, that there exist α, f , an input $(x, \beta) \in \mathbb{F}^n \times \mathbb{F}^m$, a streaming verifier \tilde{V} with $O(m \log q)$ space and a (streaming) distinguisher D with $dm^2 \text{polylog}(q)$ space such that D distinguishes real transcripts of $\text{zk-pep}(f, \alpha)$ from simulations with bias $\varepsilon = \Omega(1)$ when the input is (x, β) .

Recall that we assume that \tilde{V} rejects only after receiving all messages from P ; therefore, the algebraic commitment (y, γ, k) is always present in both real and simulated views. Our goal is to show D implies a one-way protocol for INDEX over the binary alphabet with a small message and a large bias, using Lemma 25. We do so by constructing, from D , a one-way communication protocol that distinguishes algebraic commitments to a fixed message $\alpha \in \mathbb{F}^\ell$ from algebraic commitments to a random $\alpha' \in \mathbb{F}^\ell$, where $\ell \leq dm$.

As both the real and simulated transcripts are identically distributed up to (and including) the verifier's message in Step 2, the expected distinguishing advantage and probability of a simulation failure (i.e., of an abortion in Step 3 due to $(\rho, \ell) \notin C$) are ε and $o(1)$, respectively

²² Knowledge of $f^x(\rho)$ for all $\rho \in F$ enables the simulator to sample from this distribution: F fixes $|L \cap F|$ evaluations, and the simulator sets the $dm - |L \cap F|$ remaining ones uniformly.

Input: Whitebox access to \tilde{V} ; oracle access to a length- q^{m+8} random bit string interpreted as $(z, y) \in (\mathbb{F}^m)^v \times \mathbb{F}^{dm \times p}$; streaming access to $(x, \beta) \in \Gamma^m \times \mathbb{F}^m$.

Output: View $(z, x, \beta, y, \gamma, k, (h(i) : i \in [dm]))$ with $k \in [p]$, $\gamma \in \mathbb{F}^{dm}$ and $h : \mathbb{F} \rightarrow \mathbb{F}$.

Step 0: Temporal commitment

S: Send z .

\tilde{V} : Simulate until the end of this step and let $b \in \{0, 1\}^s$ be the resulting snapshot of \tilde{V} . Use the whitebox oracle \mathcal{W} to determine the set $C \subset \{(z_i, i) : i \in [v]\}$ of size s with the largest $\mathcal{W}(b, (z_i, i))$.

Step 1: Input streaming

\tilde{V} : Stream x , computing and storing $f^x(\rho)$ for every $\rho \in \{z_i : (z_i, i) \in C\}$ while simulating the verifier.

S: Store β .

Step 2: Algebraic commitment

\tilde{V} : Simulate until \tilde{V} sends a line L , aborting if $L(0) \neq \beta$.

S: Sample a random polynomial $g : \mathbb{F} \rightarrow \mathbb{F}$ of degree at most dm subject to $g(0) = \alpha$ and $g(\beta) = f^x(L(\beta))$ for all β such that $(i, L(\beta)) \in C$ for some $i \in [v]$. Send y followed by $\gamma = (g(i) - y_{ik} : i \in [dm])$ and $k \sim [p]$.

\tilde{V} : Simulate until the end of the step.

Step 3: Temporal decommitment

\tilde{V} : Simulate until \tilde{V} sends $\rho \in \mathbb{F}^m$ and $\ell \in [v]$.

S: Check that $z_\ell = \rho \in L$ and $\rho \in \mathbb{F}^m \setminus [dm]$, aborting if either check fails or $(\rho, \ell) \notin C$.

Step 4: Algebraic decommitment

\tilde{V} : Simulate until \tilde{V} sends a line $L' : \mathbb{F} \rightarrow \mathbb{F}^m$, aborting if $L'(0) \neq k$.

S: Set $\rho := L^{-1}(\rho)$ and send $(\sum_{i=1}^{dm} \chi_i(\rho) \cdot \hat{y}_i \circ L'(j) : j \in [dm])$.

■ **Figure 8** Simulator for Figure 7.

(over z and the bits of the verifier randomness r used until then). Therefore, there exists a fixed prefix of the transcript that retains distinguishing advantage $\varepsilon/2$ and whose probability of a simulation failure is $o(1)$; indeed, at least an $\varepsilon/2$ fraction of prefixes retains advantage $\varepsilon/2$ and at most an $o(1)$ fraction yields simulation failures with $\Omega(1)$ probability, so an $\varepsilon/2 - o(1)$ fraction of prefixes work. We thus assume, in the one-way protocol we define next, not only x and β to be fixed, but also the line L and z – and, consequently, the set $C \subset \{(z_i, i) : i \in [v]\}$ (as well as the corresponding $f^x(z_i)$) that captures most of the weight of correct tuples \tilde{V} may decommit to, as given by Theorem 33.

Viewing L as the set of pairs $\{(L(\sigma), \sigma) : \sigma \in \mathbb{F}\}$, define $\ell := dm - |L \cap C|$ and assume,²³ without loss of generality, that $L \cap C = [dm] \setminus [\ell]$. Consider the following one-way communication protocol with shared randomness (for strings w of length p) that distinguishes a commitment (w, k, η) to $(f^x(i) : i \in [\ell])$ from a commitment to a random message: Alice uses S to simulate an interaction between P and \tilde{V} with input (x, β) and verifier randomness r , executing D on the (partial and fixed) transcript thus obtained, until \tilde{V} sends a line $L : \mathbb{F} \rightarrow \mathbb{F}^m$ in Step 2.

²³Note that when $|L \cap C| \geq dm$ the simulator knows the entirety of f^x_L , in which case the distinguishing bias is 0. Nonzero bias thus implies $dm > |L \cap C|$.

$$\boxed{\chi(\rho')} \cdot \begin{array}{|c|} \hline y_1 = w_1 \\ \hline y_2 = w_2 \\ \hline y_3 \\ \hline y_4 \\ \hline \end{array} = \boxed{t}$$

■ **Figure 9** Reduction from INDEX to distinguishability of views when $\ell = 3$ and $dm = 4$. The instance w is inserted into the first 2 rows of y , while y_3 is filled in with joint randomness and y_4 is the solution of the linear system shown in the diagram.

Alice samples $\rho' \sim \mathbb{F}^\times \setminus [dm]$ and continues the simulation of D by feeding it $y \in \mathbb{F}^{dm \times p}$ defined as follows: $y_i := w_i$ for $i \in [\ell]$, $y_i \sim \mathbb{F}^p$ for $\ell < i < dm$ and

$$y_{dm} := \chi_{dm}(\rho')^{-1} \cdot \left(t - \sum_{i=1}^{dm-1} \chi_i(\rho') y_i \right),$$

where $y_{\ell+1}, \dots, y_{dm-1}$ and t are random strings (in \mathbb{F}^p) shared with Bob. Note that $\rho' \notin \{0\} \cup [dm]$ implies $\chi_{dm}(\rho') \neq 0$, so that y_{dm} is well-defined. (See Figure 9 for a diagram of the reduction.)

After simulating \tilde{V} , D and S in Step 2 with y , she sends Bob all three snapshots as well as L and ρ' in a $dm^2 \text{ polylog}(q)$ -bit message.²⁴ (The space complexities of \tilde{V} and S are both dominated by the distinguisher's.)

Bob, in turn, finishes the simulation of Step 2 with his (random) index $k \in [p]$ and the correction tuple γ defined as follows:²⁵

$$\gamma_i = \begin{cases} \eta_i, & \text{if } i \leq \ell \\ \hat{x}(i) - y_{ik}, & \text{if } \ell < i < dm \end{cases}$$

and

$$\gamma_{dm} := \chi_{dm}(\rho')^{-1} \left(f_{|L}^x(\rho') - \chi_0(\rho')\alpha - t_k - \sum_{i=1}^{dm-1} \chi_i(\rho') \gamma_i \right).$$

Bob proceeds to simulate Steps 3 and 4, using S to generate the remainder of the view. Note that in the former step $(\rho, \ell) \notin C$ is the only case in which S aborts when P would not, which identifies a simulated transcript with certainty; but this is a small-probability event. When S fails (i.e., $(\rho, \ell) \notin C$) or the field element $\rho = L^{-1}(\rho)$ is not equal to ρ' , Bob halts the simulations and accepts or rejects uniformly; otherwise, he finishes the transcript by sending the low-degree polynomial that comprises the last round. This is possible because, while Bob does *not* know all \hat{y}_i , he does know the required linear combination:

$$\begin{aligned} \sum_{i=1}^{dm} \chi_i(\rho) \cdot y_i &= \sum_{i=1}^{dm-1} \chi_i(\rho) \cdot y_i + \chi_{dm}(\rho) \cdot \chi_{dm}(\rho)^{-1} \left(t - \sum_{i=1}^{dm-1} \chi_i(\rho) y_i \right) \\ &= t, \end{aligned}$$

and since t is a (random) string known to both Alice and Bob, in particular he can compute $\hat{t}_{L'}$ for any line $L' : \mathbb{F}^m \rightarrow \mathbb{F}$.

²⁴We assume Bob receives the tuple η and reads C along with the corresponding evaluations from the simulator's snapshot; alternatively, Alice could send this information in a message that is asymptotically no larger.

²⁵Recall that all y_i for all $\ell < i < dm$ are contained in Alice and Bob's shared randomness.

Finally, Bob inspects the output of D and chooses his output accordingly, accepting if and only if D accepts. Note that this one-way protocol succeeds

- with probability $1/2$ (and thus bias 0) either when S fails or when S succeeds and $\rho' \neq \rho$;
- with bias $\varepsilon/2$ when S succeeds and $\rho' = \rho$.

The latter follows from the fact that, if S succeeds and $\rho' = \rho$, it produces a full transcript where γ is a correction for the (unique) degree- dm polynomial g such that $g(0) = \alpha$, $g(i) = \eta_i + y_{ik}$ for $i \in [\ell]$ and $g(i) = f^x(i)$ for $i \in [dm] \setminus [\ell] = L \cap C$.²⁶ Therefore, if $\eta = (f^x(i) - y_{ik} : i \in [\ell])$, then γ is a correction to f^x ; while if η is random, then γ is a random degree- dm polynomial that matches f^x in (0 and) $L \cap C$. Since D distinguishes between the two cases with bias $\varepsilon/2$, then so does the one-way protocol. Therefore,

$$\begin{aligned} & \mathbb{P}_{\substack{w \sim \mathbb{F}^{\ell \times p} \\ k \sim [p]}} [B(A(w), (f^x(i) - w_{ik} : i \in [\ell]), k) \text{ accepts}] \\ & \quad - \mathbb{P}_{\substack{w \sim \mathbb{F}^{\ell \times p} \\ k \sim [p] \\ \eta \sim \mathbb{F}^{\ell}}} [B(A(w), \eta, k) \text{ accepts}] \\ & = o(1) \cdot \left(\frac{1}{2} - \frac{1}{2}\right) + (1 - o(1)) \cdot \left(1 - \frac{1}{q}\right) \cdot \left(\frac{1}{2} - \frac{1}{2}\right) + (1 - o(1)) \cdot \frac{1}{q} \cdot \frac{\varepsilon}{2} \\ & \geq \frac{\varepsilon}{3q}. \end{aligned}$$

Finally, applying Lemma 25, we conclude that there exists a one-way binary INDEX protocol for strings of length $p = m(dm q)^3$ with messages of length $\frac{dm^2 q^2 \ell^2 \log^2 q}{\varepsilon^2} \text{polylog}(q) \leq d^3 m^4 q^{2.01}$ and constant bias, a contradiction with $\sqrt{\frac{d^3 m^4 q^{2.01}}{p}} = o(1)$. ◀

► **Remark 38.** Inspecting the proof of Claim 37, we see that increasing the prover’s commitment length allows us to achieve significantly stronger indistinguishability: with $p = \log^{\omega(1)} n$, we have $\sqrt{s' q^2 \ell^2 / p} = o(1)$ for any $s' = \text{polylog}(n)$. This setting of p increases only the communication complexity of the interactive phase (Steps 2–4) – which can still be bounded by $n^{o(1)}$ – and makes the protocol secure against $\text{polylog}(n)$ -space distinguishers.

6.4 Applications: INDEX, POINT-QUERY, RANGE-COUNT and SELECTION

From the general $\text{zk-pep}(f, \alpha)$ protocol, we immediately obtain a zero-knowledge streaming interactive proof for the $\text{DECISION-INDEX}(\alpha)$ problem (Definition 20) as a corollary:

► **Corollary 39.** Fix $\delta \in (0, 1]$. For any $\alpha \in \mathbb{F}_q$ where $q = \Theta\left(\log^{1+\frac{2}{\delta}} n\right)$, $\text{DECISION-INDEX}(\alpha)$ admits a zkSIP with space complexity $O(\log n)$ and communication complexities $O(n^{1+\delta})$ and $\text{polylog}(n)$ in the setup and interactive stages, respectively. The protocol is secure against $\tilde{O}\left(\log^{2+\frac{2}{\delta}} n\right)$ -space distinguishers.

Proof. Set $d = \log^{\frac{2}{\delta}} n$ and $m = \delta \log n / 2 \log \log n$, so that $d^m = n$ and $dm/q = o(1)$. Note, moreover, that $\text{DECISION-INDEX}(\alpha)$ is the polynomial evaluation problem where $f_x = \hat{x}$, the low-degree extension of x (which can be computed in $O(m \log q)$ space) and β is the identification of a coordinate $j \in [n]$. Thus, applying Figure 7 to the mapping $x \mapsto \hat{x}$ with the aforementioned parameters, we obtain a protocol with verifier space complexity

²⁶When $L \cap C \neq [dm] \setminus [\ell]$, the set still fixes $|L \cap C|$ values of g and leaves $dm - |L \cap C|$ to be chosen randomly.

$$\begin{aligned} O(m \log q) &= O\left(\frac{\log n}{\log \log n} \cdot \log \log n\right) \\ &= O(\log n) \end{aligned}$$

and communication complexities

$$\begin{aligned} O(q^m m \log^2 q) &= \left(\log^{1+\frac{2}{\delta}} n\right)^{\frac{\delta \log n}{2 \log \log n}} \text{polylog}(n) \\ &= n^{1+\frac{\delta}{2}} \text{polylog}(n) \\ &= O(n^{1+\delta}) \end{aligned}$$

in the setup and $O(d^4 m^5 q^3 \log q) = \text{polylog}(n)$ in the interactive stage; moreover, it is secure against distinguishers with $dm^2 \text{polylog}(q) = \tilde{O}\left(\log^{2+\frac{2}{\delta}} n\right)$ space. ◀

We now select a few applications of the **zk-pep** protocol to solve other streaming problems; the remainder of this section follows reductions to **pep** due to [15].

In the **POINT-QUERY** problem, the input is a stream of updates $(u, i) \in \mathbb{Z} \times [\ell]$ to an ℓ -dimensional vector y initialised to zero, followed by an index j , and the task is to output y_j . A formal definition follows.²⁷

► **Definition 40.** Let $\ell, M \in \mathbb{N}$ and $t \in [-M, M] \cap \mathbb{Z}$. The language **POINT-QUERY**(t) is defined as

$$\left\{ (u_1, k_1, \dots, u_n, k_n, j) : \begin{array}{l} \forall i, u_i \in [-M, M] \cap \mathbb{Z} \text{ and } k_i, j \in [\ell], \\ \forall k, \left| \sum_{i \in [n], k_i=k} u_i \right| \leq M \text{ and} \\ \sum_{i \in [n], k_i=j} u_i = t \end{array} \right\}.$$

► **Corollary 41.** Fix $\delta \in (0, 1]$. Let $\ell, M \in \mathbb{N}$ with $\ell \in [n]$, $M = \text{poly}(n)$ and $t \in [-M, M] \cap \mathbb{Z}$. There exists a **zkSIP** for **POINT-QUERY**(t) with space complexity $O(\log^2 n)$ and communication complexities $O(n^{1+\delta})$ and $\text{polylog}(n)$ in the setup and interactive stages, respectively.

Proof. We first note that, by an application of the Chinese Remainder Theorem (see, e.g., [38]), we may assume $M = O(\log n)$ at the cost of a logarithmic blowup to the space complexity: the verifier runs Figure 7 in parallel with $O(\log n)$ fields $\mathbb{F}_q \supset \mathbb{F}_p$ for distinct primes $p = O(\log n)$, so that any integer in $[-M, M]$ can be uniquely represented by logarithmically many field elements.

We set the same parameters as in Corollary 39: degree $d = \log^{\frac{2}{\delta}} n$ and $m = \delta \log n / 2 \log \log n$, but also ensure $q = \Theta\left(\log^{1+\frac{2}{\delta}} n\right)$ is the power of a prime larger than $2M + 1$ (so that elements of $[-M, M] \cap \mathbb{Z}$ map to distinct field elements).

Viewing integers in $[-M, M]$ as elements of \mathbb{F} , we define $y \in \mathbb{F}^\ell$ by

$$y_k := \sum_{\substack{i \in [n] \\ k_i=k}} u_i,$$

²⁷We remark that **POINT-QUERY** is formally a promise problem: the condition that coordinatewise sums are bounded by M is assumed to hold for no-instances of the language too. However, a polynomial bound is often trivially true (as in the applications that follow).

and the mapping $x = ((u_i, k_i) : i \in [n]) \mapsto f^x$ by $f^x := \hat{y}$. Note that the verifier can compute

$$\hat{y}(\rho) = \sum_{k \in [\ell]} \left(\sum_{\substack{i=1 \\ k_i=k}}^n u_i \right) \chi_k(\rho)$$

by recording the running sum of $u_i \chi_{k_i}(\rho)$, a task for which $O(m \log q) = O(\log n)$ space suffices.

Applying Figure 7 with the mapping and parameters above, we obtain a zero-knowledge SIP with space complexity $O(\log^2 n)$ (due to the aforementioned logarithmic overhead), communication complexity $O(n^{1+\delta})$ in the setup and $\text{polylog}(n)$ in the interactive stage. ◀

With the protocol of Corollary 41, we obtain a zero-knowledge SIP for the RANGE-COUNT problem, where the stream consists of a sequence x of elements in a set $[\ell]$ followed by a subset $R \subseteq [\ell]$, and the task is to return the number of times an element of R appeared in the stream. Formally,

► **Definition 42.** Let $\mathcal{R} \subseteq 2^{[\ell]}$. The language $\text{RANGE-COUNT}(t)$ is defined as

$$\{(x, R) \in [\ell]^n \times \mathcal{R} : |\{i \in [n] : x_i \in R\}| = t\}.$$

► **Corollary 43.** Fix $\delta \in (0, 1]$. For every $\mathcal{R} \subseteq 2^{[\ell]}$ of size $\text{poly}(n)$, the language $\text{RANGE-COUNT}(t)$ admits a zkSIP with space complexity $O(\log^2 n)$ and communication complexities $O(n^{1+\delta})$ and $\text{polylog}(n)$ in the setup and interactive stages, respectively.

Proof sketch. We run the protocol for POINT-QUERY (Corollary 41) on the stream obtained by concatenating $(R' \in \mathcal{R} : x_i \in R')$ for every $i \in [n]$ (which the verifier can simulate while streaming x), followed by R (viewed as an element of $[\mathcal{R}]$). More precisely, we redefine the mapping $x \mapsto f^x$ as what would be obtained by processing the derived stream, which avoids the length overhead (to $n|\mathcal{R}| = \text{poly}(n)$, rather than n) incurred otherwise.

Since $M = n$ is an upper bound for the number of points in any subset of $[\ell]$, we obtain a protocol with the complexities as claimed. ◀

We conclude with an application of the RANGE-COUNT protocol to solve SELECTION (and MEDIAN in particular). For $x \in [\ell]^n$ and $i \in [\ell]$, we call $\varphi(x)$ the *frequency vector* of x , defined as $\varphi_i(x) = |\{j \in [n] : x_j = i\}|$ (see, also, Definition 49). A word in the language SELECTION consists of x along with a *rank* $r \in [n]$ the integer $k \in [\ell]$ with this rank and offsets $\phi \in [n]$, $\phi' \in \{0\} \cup [n]$. (We remark that the additional parameters take into account what the verifier learns in the search version of the SIP: not only the element k with rank r , but the values of the cumulative frequencies up to $k - 1$ and up to k .)

► **Definition 44.** For $\ell \in [n]$, the language SELECTION is defined as

$$\left\{ (x, k, r, \phi, \phi') \in [\ell]^n \times [\ell] \times [n] \times [n] \times \{0\} \cup [n] : \sum_{i=1}^{k-1} \varphi_i(x) = r - \phi \text{ and } \sum_{i=1}^k \varphi_i(x) = r + \phi' \right\}.$$

► **Corollary 45.** Fix $\delta \in (0, 1]$. There exists a zkSIP for SELECTION with space complexity $O(\log^2 n)$ and communication complexities $O(n^{1+\delta})$ and $\text{polylog}(n)$ in the setup and interactive stages, respectively.

Proof sketch. We execute the protocol for RANGE-COUNT twice (by temporally committing and streaming x only once; this can be done by saving two independent fingerprints for f^x , and only running zk-pep twice from Step 2 onwards). The class of ranges is $\mathcal{R} = \{[n] \setminus [i] : 0 \leq i \leq n\}$, of size $O(n)$, and the verifier checks that the number of hits in the ranges $[n] \setminus [k - 1]$ and $[n] \setminus [k]$ are $r - \phi$ and $r + \phi'$, respectively. ◀

Input: Explicit access to $\mathbb{F} = \mathbb{F}_q$, evaluation domain $H \subset \mathbb{F}$, degree d , dimension m and $\alpha \in \mathbb{F}$ as well as $f(\boldsymbol{\rho})$ with $\boldsymbol{\rho} \sim \mathbb{F}^m$, where $f : \mathbb{F}^m \rightarrow \mathbb{F}$ is a degree- d polynomial.

Repeat, from $i = 1$ to m :

P : Send the polynomial $f_i(T) = \sum_{\beta_{i+1}, \dots, \beta_m \in H} f(\boldsymbol{\rho}_1, \dots, \boldsymbol{\rho}_{i-1}, T, \beta_{i+1}, \dots, \beta_m)$.

V : Send $\boldsymbol{\rho}_i$.

V : Check that $\sum_{\beta_1 \in H} f_1(\beta_1) = \alpha$, $f(\boldsymbol{\rho}) = f_m(\boldsymbol{\rho}_m)$ and the intermediate polynomials satisfy $\sum_{\beta_i \in H} f_i(\beta_i) = f_{i-1}(\boldsymbol{\rho}_{i-1})$ for all $2 \leq i < m$, accepting if so and rejecting otherwise.

■ **Figure 10** Protocol $\text{sumcheck}(f, \alpha)$.

7 A zero-knowledge sumcheck SIP

In the previous section we showed how Figure 3, the polynomial evaluation protocol of [24], can be made zero-knowledge with the careful addition of algebraic and temporal commitment protocols. Although PEP is a foundational problem for streaming algorithms – generalising INDEX, for example – it is not immediately clear whether the same techniques enable us to construct a zero-knowledge version of the second widely used tool in SIPs: the *sumcheck* protocol. In this section, we prove that they do: Figure 11 lays out zk-sumcheck , a zkSIP for the SUMCHECK problem (Definition 46) with the same components, namely, the algebraic and temporal commitments that enabled zk-pep .

Sumcheck protocols are extremely useful building blocks for the construction of interactive proofs; indeed, some of the most celebrated results of the last two decades rely on them, most notably the GKR [34] and subsequent delegation-of-computation protocols (e.g., [52, 50, 51]). Roughly speaking, they allow a verifier to check that the sum, over a subcube, of the evaluations of a polynomial yields a prescribed field element; they save *exponentially* in the communication (and time) complexity as compared to sending the entire description of the polynomial. In particular, they enable the (exact) computation of frequency moments of a stream via an interactive protocol in sublinear space [13], which is impossible without interaction [2].

More precisely, let $f : \mathbb{F}^m \rightarrow \mathbb{F}$ be a polynomial of (individual) degree d and $H \subset \mathbb{F}$ be an evaluation domain. One obvious way to check that $\sum_{\boldsymbol{\beta} \in H^m} f(\boldsymbol{\beta})$ is equal to some $\alpha \in \mathbb{F}$ is via the description of f (say, as a list of sufficiently many evaluations), from which the sum can be computed directly. This requires not only the entire description of f , which has size $(d+1)^m$; but also entails evaluating f over $|H|^m$ many points, implying an even larger runtime.

The standard sumcheck protocol (Figure 10) enables a verifier V to offload this costly computation to a powerful prover P and check the claim by communicating $O(dm)$ field elements in $O(|H|md)$ time steps, with a *single random evaluation of f* .²⁸

It is well known that the protocol above (always) accepts if $\sum_{\boldsymbol{\beta} \in H^m} f(\boldsymbol{\beta}) = \alpha$, and rejects with probability at least $1 - dm/q$ otherwise (see, e.g., [3]). As sums of polynomials can be performed in a streaming fashion, the verifier only needs $O(m \log q)$ bits of space.

²⁸ Figure 10 is laid out in a somewhat non-standard (but equivalent) form, with checks deferred to the end, that more closely resembles the streaming version we construct.

7.1 The protocol

We now show that the techniques of Section 5 enable us to construct a streaming zero-knowledge variant of $\text{sumcheck}(f, \alpha)$, which solves the problem defined next.

► **Definition 46.** Let $\alpha \in \mathbb{F}$, $H \subseteq \mathbb{F}$ and $f = \{f^x : x \in \Gamma^n\}$ be a mapping such that $f^x : \mathbb{F}^m \rightarrow \mathbb{F}$ is a degree- d polynomial. $\text{SUMCHECK}(f, \alpha)$ is the language $\left\{x \in \Gamma^n : \sum_{\beta \in H^m} f^x(\beta) = \alpha\right\}$.

The techniques need to be adapted, however, with one key distinction between zk-sumcheck and zk-pep : the prover now must make many (algebraic) commitments, each of which is used in a pair of decommitments; moreover, the commitments cannot be sent in parallel anymore, owing to dependencies between messages in contiguous rounds. Intuitively, neither of these should pose too great a challenge: computing fingerprints of a set of messages whose commitment is sent sequentially should be no easier than when they are sent in parallel (indeed, for one-way communication protocols they are exactly equivalent); and if one algebraic decommitment does not leak a significant amount of information, two should not do so either.

The protocol follows. We note that (differently from Section 6) $\chi(\rho)$ denotes the vector of Lagrange polynomials over \mathbb{F} for degree- d univariate polynomials with interpolating set $[d+1]$, i.e., $\chi(\rho) = (\chi_i(\rho) : i \in [d+1]) \in \mathbb{F}^{d+1}$.

7.2 Analysis of the protocol

We now show that zk-sumcheck is a valid (i.e., complete and sound) streaming interactive proof, and compute its space and communication complexities.

► **Theorem 47.** Let f be such that an evaluation of the \mathbb{F}_q -polynomial f^x can be computed by streaming x in $O(m^2 \log q)$ space. For any $\alpha \in \mathbb{F}_q$, Figure 11 is an SIP for $\text{SUMCHECK}(f, \alpha)$ with space complexity $s = O(m^2 \log q)$, communication complexity $O(q^m m \log^2 q)$ in the setup and $O(q^{\log \log q} d m \log q) = q^{\log \log q} \text{poly}(q)$ in the interactive phase.

Proof. As in Theorem 35, we first show completeness and soundness, then compute the complexities.

Completeness. Recall that $\text{decommit}(\beta, w, k)$ with correction γ accepts if (the fingerprint matches the LDE of w and) $\gamma + w_k = \beta$. Therefore, when P and V are both honest, the first $m-1$ decommitments of Step 4 accept, since

$$\begin{aligned}
& \theta \cdot \gamma^{(i)} - \chi(\rho_{i-1}) \cdot \gamma^{(i-1)} + \left(\theta \cdot y^{(i)} - \chi(\rho_{i-1}) \cdot y^{(i-1)} \right)_k \\
&= \sum_{j=1}^{d+1} \left(\theta_j (\gamma_j^{(i)} + y_{jk}^{(i)}) - \chi_j(\rho_{i-1}) (\gamma_j^{(i-1)} + y_{jk}^{(i-1)}) \right) \\
&= \sum_{j=1}^{d+1} \theta_j f_i(j) - \sum_{j=1}^{d+1} \chi_j(\rho_{i-1}) f_{i-1}(j) \\
&= \left(\sum_{\beta \in H} f_i(\beta) \right) - f_{i-1}(\rho_{i-1}) \\
&= 0.
\end{aligned}$$

Input: Explicit access to \mathbb{F} , element $\alpha \in \mathbb{F}$, degree d , dimension m , evaluation domain $H \subset \mathbb{F}$ and mapping $x \mapsto f^x$; streaming access to $x \in \Gamma^n$.

Parameters:

Field size $q = |\mathbb{F}|$ satisfying $dm = o(q)$;

Commitment lengths $v = q^m(\log m + \log \log q)/96$ and $p = q^{\log \log q}$.

Step 0: Temporal commitment

P: Send a string $z \sim ((\mathbb{F} \setminus [d+1])^m)^v$.

V: Sample $\rho \sim (\mathbb{F} \setminus [d+1])^m$ and stream z . Check if $z_i = \rho$ for each i , storing $\ell := i$ if so.

Reject if $\rho \neq z_i$ for all $i \in [v]$.

Step 1: Input streaming

V: Stream x and compute $f^x(\rho) \in \mathbb{F}$.

Step 2: Algebraic commitments

P: Compute $f_1(T) = \sum_{\beta_2, \dots, \beta_m \in H} f^x(T, \beta_2, \dots, \beta_m)$ and sample $k \sim [p]$.

V: Sample $\sigma^{(1)}, \dots, \sigma^{(m+1)} \sim \mathbb{F}^m$. Compute $\chi(\rho_1), \dots, \chi(\rho_m)$ and the linear coefficients θ such that $\sum_{\beta \in H} g(\beta) = \sum_i \theta_i g(i)$ when g is a degree- d univariate polynomial.

Repeat, from $i = 1$ to m :

P: Send $y^{(i)} \sim \mathbb{F}^{(d+1) \times p}$ and $\gamma^{(i)} = (f_i(j) - y_{jk}^{(i)} : j \in [d+1])$.

V: Compute the fingerprints $\hat{y}^{(i)}(\sigma^{(i)}, \chi(\rho_i))$ and $\hat{y}^{(i)}(\sigma^{(i+1)}, \theta)$, as well as the dot products $\chi(\rho_i) \cdot \gamma^{(i)}$ and $\theta \cdot \gamma^{(i)}$.

Send ρ_i .

P: If $i < m$, compute $f_{i+1}(T) = \sum_{\beta_{i+2}, \dots, \beta_m \in H} f^x(\rho_1, \dots, \rho_i, T, \beta_{i+2}, \dots, \beta_m)$.

P: Send k .

Step 3: Temporal decommitment

V: Send ℓ .

P: Check that $z_\ell = \rho \in (\mathbb{F} \setminus [d+1])^m$, aborting otherwise.

Step 4: Algebraic decommitments

V: For all $1 < i \leq m$, run

decommit $(0, \theta \cdot y^{(i)} - \chi(\rho_{i-1}) \cdot y^{(i-1)}, k)$, with

fingerprints $\hat{y}^{(i)}(\sigma^{(i)}, \theta) - \hat{y}^{(i-1)}(\sigma^{(i)}, \chi(\rho_{i-1}))$ and correction $\theta \cdot \gamma^{(i)} - \chi(\rho_{i-1}) \cdot \gamma^{(i-1)}$.

Run decommit $(\alpha, \theta \cdot y^{(1)}, k)$ with fingerprint $\hat{y}^{(1)}(\sigma^{(1)}, \theta)$ and correction $\theta \cdot \gamma^{(1)}$.

Run decommit $(f^x(\rho), \chi(\rho_m) \cdot y^{(m)}, k)$ with fingerprint $\hat{y}^{(m)}(\sigma^{(m+1)}, \chi(\rho_m))$ and correction $\chi(\rho_m) \cdot \gamma^{(m)}$.

Accept if all decommitments accept, and reject otherwise.

■ **Figure 11** Protocol zk-sumcheck(f, α).

Likewise, the last two decommitments accept because

$$\begin{aligned}
\boldsymbol{\theta} \cdot \boldsymbol{\gamma}^{(1)} + \left(\boldsymbol{\theta} \cdot \boldsymbol{y}^{(1)} \right)_k &= \sum_{j=1}^{d+1} \boldsymbol{\theta}_j (\boldsymbol{\gamma}_j^{(1)} + \boldsymbol{y}_{jk}^{(1)}) \\
&= \sum_{j=1}^{d+1} \boldsymbol{\theta}_j f_1(j) \\
&= \sum_{\beta \in H} f_1(\beta) \\
&= \sum_{\beta \in H^m} f(\beta) \\
&= \alpha
\end{aligned}$$

and

$$\begin{aligned}
\boldsymbol{\chi}(\boldsymbol{\rho}_m) \cdot \boldsymbol{\gamma}^{(m)} + \left(\boldsymbol{\chi}(\boldsymbol{\rho}_m) \cdot \boldsymbol{y}^{(m)} \right)_k &= \sum_{j=1}^{d+1} \chi_j(\boldsymbol{\rho}_m) (\boldsymbol{\gamma}_j^{(m)} + \boldsymbol{y}_{jk}^{(m)}) \\
&= \sum_{j=1}^{d+1} \chi_j(\boldsymbol{\rho}_m) f_m(j) \\
&= f_m(\boldsymbol{\rho}_m) \\
&= f^x(\boldsymbol{\rho}),
\end{aligned}$$

respectively. The verifier thus accepts unless $\boldsymbol{\rho} \neq \{z_i : i \in [v]\}$ in Step 0, an event with probability

$$\left(1 - \frac{1}{q-d-1} \right)^v \leq e^{-v/(q-d-1)^m} \leq e^{-v/q^m} = o(1).$$

Soundness. We divide the behaviour of a malicious prover into three cases. The first (and simplest) is when \tilde{P} commits to f_i for all i and decommits with polynomials whose evaluations at 0 yield the same values as the honest prover (i.e., in $\text{decommit}(\beta, w, k)$ with γ as the correction, \tilde{P} replies with a polynomial g such that $g(0) = w_k + \gamma$). Then, since $\sum_{\beta \in H^m} f(\beta) \neq \alpha$, the verifier rejects in $\text{decommit}(\alpha, \boldsymbol{\theta} \cdot \boldsymbol{y}^{(1)}, k)$ with probability 1.

The second case is when \tilde{P} commits to a sequence of polynomials g_1, \dots, g_m such that $g_i \neq f_i$ for some i , and decommits honestly. Then V accepts if and only if the set $\{g_i\}$ leads the verifier in the standard sumcheck protocol to accept; by the soundness of that protocol, V accepts with probability at most $dm/(q-d-1) = o(1)$.

The only remaining case is when \tilde{P} commits to a sequence of polynomials $\{g_i\}$ (which may or may not coincide with $\{f_i\}$) and, in at least one decommitment with respect to a string w where \tilde{P} receives the line L , the prover replies with a degree- dm polynomial g such that $g(0) \neq w_k = \hat{w}_L(0)$. Then, since V has a fingerprint $\hat{w}(\boldsymbol{\sigma})$ with $\boldsymbol{\sigma} \sim \mathbb{F}^m$ and a field element $\sigma \sim \mathbb{F}$ such that $L(\sigma) = \boldsymbol{\sigma}$, we have $g(\sigma) \neq \hat{w}(\boldsymbol{\sigma}) = \hat{w}_L(\sigma)$ with probability $dm/q = o(1)$ by Lemma 18 (Schwartz-Zippel), and soundness follows.

Space and communication complexities. The communication of the setup (Step 0, the temporal commitment) is $q^m(\log m + \log \log q)m \log q = O(q^m m \log^2 q)$ bits. The communication of the interactive phase (Steps 2–4) is dominated by the m algebraic commitments to elements of \mathbb{F}^{d+1} with length $p = q^{\log \log q}$ each, for a total of $O(q^{\log \log q} dm \log q) \leq q^{\log \log q + 2}$ bits.

The verifier's space complexity is dominated by computing $f^x(\rho)$ and storing $O(m)$ elements of \mathbb{F}^m (i.e., ρ and $\sigma^{(i)}$ for $i \in [m+1]$), so that it is bounded by $O(m^2 \log q)$. ◀

7.3 Zero-knowledge

Having shown that zk-sumcheck is a valid streaming interactive proof, we now show it is also zero-knowledge.

► **Theorem 48.** *Figure 11 is zero-knowledge against $\text{poly}(q)$ -space streaming distinguishers. The simulator has space complexity $\text{poly}(q)$.*

Proof. We shall prove indistinguishability as we have done earlier: with the simulator S shown in Figure 12, we assume towards contradiction that there exists $\alpha \in \mathbb{F}$, an input $x \in \mathbb{F}^n$, internal randomness r , a space- $O(m^2 \log q)$ verifier \tilde{V} and a $\text{poly}(q)$ -space distinguisher D that accepts $\text{View}_{P, \tilde{V}}(x, r)$ with probability $\varepsilon = \Omega(1)$ above that with which D accepts $S(\tilde{V}, x, r)$. Then, via Lemma 25, we construct a one-way protocol for INDEX with impossibly large success probability.

The space complexity of S is dominated by its storing of $O(m^2 \log q) = \text{poly}(q)$ elements of $\mathbb{F}^m \times [v]$ and by the computation of the partial sums $(g_i : i \in [m])$. Note that the naive strategy of sampling g and computing the corresponding partial sums requires $\Omega(d^m)$ space; however, [5] constructs an algorithm that can sample from the same distribution in $\text{poly}(q)$ time, and thus space.²⁹ Note, moreover, that the alphabet over which z is taken has size

$$\begin{aligned} (q - d - 1)^m &= q^m \left(1 - \frac{d+1}{q}\right)^m \\ &\geq q^m \left(1 - \frac{1}{m}\right)^m \\ &\geq \frac{q^m}{3} \\ &\geq \frac{32v}{\log \log v}, \end{aligned}$$

so that Theorem 33 applies. (The conditions $(q - d - 1)^m = \Theta\left(\frac{v}{\log \log v}\right)$ and $\log q \leq s = \text{polylog}(q)$ are also clearly satisfied.)

We fix a string z (and thus the set C of the verifier's likely decommitments) along with bits of the verifier's random string r that ensure distinguishing bias at least $\varepsilon/2$ and $o(1)$ probability of simulation failure (recall that failure corresponds to the event $(\rho, \ell) \notin C$). Consider the following (linear) mapping between \mathbb{F} -vector spaces: from polynomials $g : \mathbb{F}^m \rightarrow \mathbb{F}$ of degree at most d that satisfy the $s+1$ linear constraints of the fingerprints and subcube sum (i.e., $g(\rho) = z_i$ for all $(z_i, i) \in C$ and $\sum_{\beta \in H^m} g(\beta) = \alpha$) to the sequence of univariate (partial sum) polynomials $\sum_{\beta_{i+1}, \dots, \beta_m \in H} g(\rho_1, \dots, \rho_{i-1}, T, \beta_{i+1}, \dots, \beta_m)$ for all $i \in [m]$ and evaluation points ρ in C .

Let $\ell \leq (d+1)m$ be the dimension of the image of this mapping, and let $\xi = \xi(\rho) \in \mathbb{F}^{(d+1)m \times \ell}$ be the linear coefficients that map vectors in \mathbb{F}^ℓ to partial sums (given by $d+1$ evaluations) with respect to ρ . We now proceed to Alice's strategy, who receives $w \in \mathbb{F}^{\ell \times p}$

²⁹ More precisely, the algorithm of [5] allows us to sample from the distributions $g_i(\beta)$ for any β and i under the uniform distribution of g satisfying a set of constraints. To sample (g_1, \dots, g_m) , we begin with the set of constraints induced by C and, after sampling $g_i(j)$, include the corresponding constraint before the next sample.

Input: Whitebox access to \tilde{V} ; oracle access to random bit string of length $q^{m+\log \log q} \text{poly}(q)$ interpreted as the concatenation of $z \in (\mathbb{F}^m)^v$ and $y^{(i)} \in \mathbb{F}^{(d+1) \times p}$ for all $i \in [m]$.

Output: View $(z, x, (y^{(i)}, \gamma^{(i)} : i \in [m]), k, (h_i : i \in [m+1]))$ with $z \in ((\mathbb{F} \setminus [d+1])^d)^v$, $y^{(i)} \in \mathbb{F}^{(d+1) \times p}$, $\gamma^{(i)} \in \mathbb{F}^{d+1}$, $k \in [p]$ and $h_i : \mathbb{F} \rightarrow \mathbb{F}$ of degree dm .

Step 0: Temporal commitment

S: Send $z \in ((\mathbb{F} \setminus [d+1])^m)^v$.

\tilde{V} : Simulate until the end of this step and let $b \in \{0, 1\}^s$ be the resulting snapshot of \tilde{V} .
Use the whitebox oracle \mathcal{W} to determine the set $C \subset \{(z_i, i) : i \in [v]\}$ of size s with the largest $\mathcal{W}(b, (z_i, i))$.

Step 1: Input streaming

\tilde{V} : Stream x , simulating the verifier while computing and storing $f^x(z_i)$ for all $(z_i, i) \in C$.

Step 2: Algebraic commitments

S: Take $g_1 : \mathbb{F} \rightarrow \mathbb{F}$ of degree (at most) d under the distribution determined by sampling $g : \mathbb{F}^m \rightarrow \mathbb{F}$ subject to the constraints $\sum_{\beta \in H^m} g(\beta) = \alpha$ and $g(z_i) = f^x(z_i)$ for all $(z_i, i) \in C$, then outputting $g_1(T) = \sum_{\beta_2, \dots, \beta_m \in H} g(T, \beta_2, \dots, \beta_m)$.
Sample $k \sim [p]$.

\tilde{V} : Simulate until the end of the step.

Repeat, from $i = 1$ to m :

S: Send $y^{(i)}$ and $\gamma^{(i)} = (g_i(j) - y_{jk}^{(i)} : j \in [d+1])$.

\tilde{V} : Simulate until ρ_i is sent (or until the end of the step when $i = m$).

S: If $i < m$, sample g_{i+1} under the distribution given by taking g randomly and outputting $g_{i+1}(T) = \sum_{\beta_{i+2}, \dots, \beta_m \in H} g(\rho_1, \dots, \rho_i, T, \beta_{i+2}, \dots, \beta_m)$.

S: Compute θ such that $\sum_{\beta \in H} h(\beta) = \sum_i \theta_i h(i)$ when h is a degree- d univariate polynomial, and send k .

Step 3: Temporal decommitment

\tilde{V} : Simulate until \tilde{V} sends $\ell \in [v]$.

S: Abort if $z_\ell \neq \rho$, $\rho \notin (\mathbb{F} \setminus [d+1])^m$ or $(\rho, \ell) \notin C$.

Step 4: Algebraic decommitments

For all $1 < i \leq m$,

\tilde{V} : Simulate until \tilde{V} sends a line $L_i : \mathbb{F} \rightarrow \mathbb{F}^m$.

S: Abort if $L_i(0) \neq k$, and otherwise send

$$\left((\theta \cdot \hat{y}^{(i)} - \chi(\rho_i) \cdot \hat{y}^{(i-1)}) \circ L_i(j) : j \in [dm+1] \right).$$

\tilde{V} : Simulate until \tilde{V} sends a line $L_1 : \mathbb{F} \rightarrow \mathbb{F}^m$.

S: Abort if $L_1(0) \neq k$, and otherwise send

$$\left((\theta \cdot \hat{y}^{(1)}) \circ L_1(j) : j \in [dm+1] \right).$$

\tilde{V} : Simulate until \tilde{V} sends a line $L_{m+1} : \mathbb{F} \rightarrow \mathbb{F}^m$.

S: Abort if $L_{m+1}(0) \neq k$, and otherwise send

$$\left((\chi(\rho_m) \cdot \hat{y}^{(m)}) \circ L_{m+1}(j) : j \in [dm+1] \right).$$

■ **Figure 12** Simulator for Figure 11.

as input and uses a random $y'^{(i)}$ shared with Bob for each commitment string $y^{(i)}$. She will also use $t^{(i)}$ for each pair $y^{(i-1)}, y^{(i)}$; additionally, $t^{(1)}$ and $t^{(m+1)}$ will be used for $y^{(1)}$ and $y^{(m)}$, respectively. The $t^{(i)}$ will ensure Bob knows the linear combination of every algebraic decommitment.

More precisely, Alice runs S (with the fixed string z and partially fixed r) until the end of Step 0, determines the set C , samples $\rho' \sim F = \{z_i : (z_i, i) \in C\}$ and sets $\xi = \xi(\rho')$. For every $(i, j) \in [m-1] \times [d]$ and $(i, j) \in \{m\} \times [d-1]$, she sets $y_j^{(i)} = y_j'^{(i)} + (\xi \cdot w)_{(i-1)d+j}$. She also sets the remaining rows (i.e., $y_{d+1}^{(i)}$ for all i as well as $y_d^{(m)}$) to satisfy

$$\begin{aligned} \theta \cdot y^{(1)} &= t^{(1)}, \\ \theta \cdot y^{(i)} - \chi(\rho'_{i-1}) \cdot y^{(i-1)} &= t^{(i)} \quad \text{for } 1 < i \leq m \text{ and} \\ \chi(\rho'_m) \cdot y^{(m)} &= t^{(m+1)}. \end{aligned}$$

Note that these are $m+1$ linear constraints on $m+1$ row vectors of dimension p , and since θ_{d+1} and $\chi_d(\rho'_m)$ are nonzero, there is at least one solution.³⁰ (If some constraint is not independent from the others, Alice replaces it with a “canonical” constraint to ensure a unique solution, e.g., setting the linear coefficients for $y_{d+1}^{(i)}$ with the smallest bit representation that makes the constraint independent.) She then simulates Step 1 and the part of Step 2 until \tilde{V} (and D) finish streaming the $y^{(i)}$, sending the resulting snapshots of S , \tilde{V} and D to Bob along with ρ' in a poly(q)-bit message.

Bob reads his input (η, k) and sets the correction tuples $\gamma^{(i)} \in \mathbb{F}^{d+1}$ so as to satisfy constraints with the same linear coefficients as $y^{(i)}$: he sets $\gamma_j^{(i)} = (\xi \cdot \eta)_{(i-1)d+j} - y_{jk}'^{(i)}$ for $(i, j) \in [m-1] \times [d]$ and $(i, j) \in \{m\} \times [d-1]$; then sets the coordinates $i = d+1$ and $j \in [m]$ as well as $(i, j) = (d, m)$ to satisfy

$$\begin{aligned} \theta \cdot \gamma^{(1)} &= \alpha - t_k^{(1)}, \\ \theta \cdot \gamma^{(i)} - \chi(\rho'_{i-1}) \cdot \gamma^{(i-1)} &= -t_k^{(i)} \quad \text{for } 1 < i \leq m \text{ and} \\ \chi(\rho'_m) \cdot \gamma^{(m)} &= f^x(\rho') - t_k^{(m+1)}. \end{aligned}$$

Bob then finishes the simulation of Step 2 with the coordinate $k \in [p]$.

In Step 3, if $\rho' \neq \rho$ or the simulation fails (i.e., $z_\ell = \rho$ but $(\rho, \ell) \notin C$), Bob accepts or rejects uniformly at random. Otherwise, he simulates Step 4 until the protocol terminates (which his access to the shared random strings $t^{(i)}$ enables him to). At the end of the simulation, Bob accepts if and only if D accepts.

Note that, when $\eta = \tau - (w_{ik} : i \in [\ell])$ for a vector τ that maps to the polynomials $(g_i : i \in [m])$ via $\xi = \xi(\rho)$, then $\gamma_j^{(i)}$ satisfies

$$\begin{aligned} \gamma_j^{(i)} &= (\xi \cdot \eta)_{(i-1)d+j} - y_j'^{(i)} \\ &= g_i(j) - (\xi \cdot w)_{(i-1)d+j,k} - y_{jk}'^{(i)} \\ &= g_i(j) - y_{jk}^{(i)} \end{aligned}$$

for all i, j in $[m-1] \times [d]$ and $\{m\} \times [d-1]$ (equivalently, for all i, j such that $y_j^{(i)}$ includes a linear combination of the rows of w). Then the linear constraints satisfied by the other $m+1$ pairs ensures the equality extends to all (i, j) : for $i \in [m], j = d+1$ and $(i, j) = (m, d)$, we have

³⁰The condition $\chi_d(\rho'_m) \neq 0$ follows from choosing $\rho'_m \notin [d+1]$, and we assume the last entry of θ is nonzero without loss of generality. Note that if θ is the zero vector the problem trivialises: in this case the verifier does not need assistance from a prover (or even to stream x), accepting if and only if $\alpha = 0$.

$$\begin{aligned}
\boldsymbol{\theta} \cdot \boldsymbol{\gamma}^{(1)} &= \alpha - t_k^{(1)} \\
&= \sum_{j=1}^{d+1} \boldsymbol{\theta}_j g(j) - t_k^{(1)} \\
&= \sum_{j=1}^{d+1} \boldsymbol{\theta}_j \left(g(j) - y_{jk}^{(1)} \right), \\
\boldsymbol{\chi}(\boldsymbol{\rho}_m) \cdot \boldsymbol{\gamma}^{(m)} &= f^x(\boldsymbol{\rho}) - t_k^{(m+1)} \\
&= g_m(\boldsymbol{\rho}_m) - t_k^{(m+1)} \\
&= \sum_{j=1}^{d+1} \chi_j(\boldsymbol{\rho}_m) \left(g_m(j) - y_{jk}^{(m)} \right),
\end{aligned}$$

and, for $1 < i \leq m$,

$$\begin{aligned}
\boldsymbol{\theta} \cdot \boldsymbol{\gamma}^{(i)} - \boldsymbol{\chi}(\boldsymbol{\rho}_{i-1}) \cdot \boldsymbol{\gamma}^{(i-1)} &= -t_k^{(i)} \\
&= \sum_{j=1}^{d+1} \left(\chi_j(\boldsymbol{\rho}_{i-1}) y_{jk}^{(i-1)} - \boldsymbol{\theta}_j y_{jk}^{(i)} \right) \\
&= \sum_{j=1}^{d+1} \boldsymbol{\theta}_j \left(g_i(j) - y_{jk}^{(i)} \right) + \sum_{j=1}^{d+1} \chi_j(\boldsymbol{\rho}_{i-1}) \left(g_{i-1}(j) - y_{jk}^{(i-1)} \right).
\end{aligned}$$

That is, since the $\boldsymbol{\gamma}^{(i)}$ satisfy the same linear constraints as the vectors $(g_i(j) - y_{jk}^{(i)} : j \in [d+1])$, it follows that they are equal. Therefore the resulting view is distributed *exactly* as $\text{View}_{P, \tilde{V}}(x, r)$ when $\boldsymbol{\tau}$ maps to the partial sums of f^x (and thus $\boldsymbol{\xi}(\boldsymbol{\rho}) \cdot \boldsymbol{\tau}$ maps to the partial sums with respect to $\boldsymbol{\rho}$); and if $\boldsymbol{\eta} \sim \mathbb{F}^\ell$, it is distributed as $S(\tilde{V}, x, r)$ (unless the simulation fails or $\boldsymbol{\rho} \neq \boldsymbol{\rho}'$).

This one-way protocol achieves bias 0 when the simulation fails (an $o(1)$ -probability event) or the verifier’s temporal decommitment $\boldsymbol{\rho}$ is in C (i.e., the simulation succeeds) but $\boldsymbol{\rho} \neq \boldsymbol{\rho}'$, an event with conditional probability $1 - \frac{1}{|C|} = 1 - \frac{1}{s}$. Otherwise, it achieves a bias of $\varepsilon/2$. We thus have

$$\begin{aligned}
&\mathbb{P}_{\substack{w \sim \mathbb{F}^{\ell \times p} \\ k \sim [p]}} [B(A(w), (f^x(i) - w_{ik} : i \in [\ell]), k) \text{ accepts}] \\
&\quad - \mathbb{P}_{\substack{w \sim \mathbb{F}^{\ell \times p} \\ k \sim [p] \\ \boldsymbol{\eta} \sim \mathbb{F}^\ell}} [B(A(w), \boldsymbol{\eta}, k) \text{ accepts}] \\
&= o(1) \cdot 0 + (1 - o(1)) \cdot \left(1 - \frac{1}{s}\right) \cdot 0 + (1 - o(1)) \cdot \frac{1}{s} \cdot \frac{\varepsilon}{2} \\
&\geq \frac{\varepsilon}{3s}.
\end{aligned}$$

Applying Lemma 25 yields a one-way binary INDEX protocol for strings of length $p = q^{\log \log q}$ with messages of length $\frac{s^2 \ell^2 \log_3^2 q}{\varepsilon^2} \text{poly}(q) = \text{poly}(q)$ and constant bias. But this contradicts Proposition 21’s upper bound of $O\left(\sqrt{\text{poly}(q)/p}\right) = o(1)$. \blacktriangleleft

7.4 Applications: FREQUENCY-MOMENT and INNER-PRODUCT

We now proceed to applications of zk-sumcheck. The first is a zkSIP that (exactly) computes frequency moments of order $k > 1$ (commonly denoted F_k) for a stream over an alphabet of size ℓ , a problem known to require $\Omega(\ell)$ space without a prover [2].

► **Definition 49.** Fix $k \in \mathbb{N}$. For every $\ell \in [n]$ and $t \in [n^k]$, the language $\text{FREQUENCY-MOMENT}_k(t)$ is $\left\{x \in [\ell]^n : \sum_{i \in [\ell]} \varphi_i(x)^k = t\right\}$, where $\varphi_i(x) := |\{j \in [n] : x_j = i\}|$.

► **Corollary 50.** Fix $1 < k \in \mathbb{N}$ and $\delta \in (0, 1]$. For every $\ell \in [n]$ and $t \in [n^k]$, there exists a zero-knowledge SIP for $\text{FREQUENCY-MOMENT}_k(t)$ with space complexity $O(\log^2 n / \log \log n)$. The communication complexity is $O(n^{1+\delta})$ in the setup and $n^{o(1)}$ in the interactive phase, and the protocol is secure against $\text{polylog}(n)$ -space distinguishers.

Proof. We set parameters analogously to Corollary 39, but take into account the factor- k blowup in the degree of f^x : set degree $d = k \log^{\frac{2}{\delta}} n = O\left(\log^{\frac{2}{\delta}} n\right)$, dimension $m = \frac{\delta \log n}{2 \log \log n}$, and take a field \mathbb{F} of size $|\mathbb{F}| = q = \Theta\left(\log^{1+\frac{2}{\delta}} n\right)$. The mapping $x \mapsto f^x$ is defined as follows: viewing $[\ell] \hookrightarrow [d+1]^m \hookrightarrow \mathbb{F}^m$ and defining the frequency vector $\varphi = \varphi(x) := (\varphi_i(x) : i \in [\ell])$, set $f^x(\alpha) := \sum_{i \in [d+1]^m} \hat{\varphi}(i, \alpha)^k$ for $\alpha \in \mathbb{F}^{m-1}$, where $\hat{\varphi}$ is the degree- d/k extension of φ . Note that f^x is a $(m-1)$ -variate degree- d polynomial.

Using $O(dm \log q) = O(m^2 \log q)$ bits of space (recall that k is constant), the verifier can compute all the low-degree extensions $\hat{\varphi}(i, \rho) \in \mathbb{F}$ (by adding $\chi_{x_j}(i, \rho)$ to each running sum upon reading x_j); then, after the stream, V raises each LDE to the k^{th} power and adds the results to obtain $f^x(\rho)$.

Applying Figure 11, the verifier checks whether

$$\sum_{\alpha \in [d+1]^{m-1}} f^x(\alpha) = \sum_{\beta \in [d+1]^m} \hat{\varphi}(\beta)^k = \sum_{i \in [\ell]} \varphi_i^k$$

is equal to t . The space complexity is $O(m^2 \log q) = O(\log^2 n / \log \log n)$; the communication complexity of the setup step is of order

$$q^m m \log^2 q = n^{1+\frac{\delta}{2}} \text{polylog}(n) = O(n^{1+\delta}),$$

and $q^{\log \log q} \text{poly}(q) = n^{o(1)}$ in the interactive phase. Lastly, the protocol is secure against distinguishers with space $\text{poly}(q) = \text{polylog}(n)$. ◀

Our second and last application is a small modification of the F_2 protocol that allows us to compute inner products.

► **Definition 51.** For every $\ell \in [n]$, $t \in [n^2 \ell]$ and field \mathbb{F} , the language $\text{INNER-PRODUCT}(t)$ is defined as $\left\{(x, y) \in \mathbb{F}^n \times \mathbb{F}^n : \varphi(x) \cdot \varphi(y) = \sum_{i \in [\ell]} \varphi_i(x) \varphi_i(y) = t\right\}$.

► **Corollary 52.** For every $\delta \in (0, 1]$, $\ell \in [n]$, $t \in [n^2 \ell]$ and field \mathbb{F}_q with $q = \Theta\left(\log^{1+\frac{2}{\delta}} n\right)$, there exists a zkSIP for $\text{INNER-PRODUCT}(t)$ with space complexity $O(\log^2 n / \log \log n)$ and communication complexities $O(n^{1+\delta})$ and $n^{o(1)}$ in the setup and communication phases, respectively.

Proof. We use the same parameter settings as Corollary 50 and define

$$f^{x,y}(\alpha) = \sum_{i \in [d+1]} \widehat{\varphi(x)}(i, \alpha) \widehat{\varphi(y)}(i, \alpha),$$

a polynomial of degree $2d = 2 \log^{\frac{2}{3}} n$ whose evaluation the verifier computes by saving $\widehat{\varphi(x)}(i, \rho)$ and $\widehat{\varphi(y)}(i, \rho)$ for $i \in [d+1]$. Figure 10 enables the verifier to check that $\sum_{i \in [\ell]} \varphi_i(x) \varphi_i(y)$ equals t , as desired, with complexities of the same order as in Corollary 50. ◀

We remark that while one might reduce inner product to F_2 , by taking the difference between the second moment of $\varphi(x) + \varphi(y)$ and the second moments of $\varphi(x)$ and $\varphi(y)$, the resulting protocol leaks these values, and is therefore not zero-knowledge.

References

- 1 Amirali Abdullah, Samira Daruki, Chitradeep Dutta Roy, and Suresh Venkatasubramanian. Streaming verification of graph properties. In Seok-Hee Hong, editor, *27th International Symposium on Algorithms and Computation, ISAAC 2016, December 12-14, 2016, Sydney, Australia*, volume 64 of *LIPICs*, pages 3:1–3:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICs.ISAAC.2016.3. 6
- 2 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and system sciences*, 58(1):137–147, 1999. 2, 5, 16, 50, 58
- 3 Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009. URL: <http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264>. 50
- 4 Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. 2
- 5 Eli Ben-Sasson, Alessandro Chiesa, Michael A. Forbes, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Zero knowledge protocols from succinct constraint detection. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography – 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 172–206. Springer, 2017. doi:10.1007/978-3-319-70503-3_6. 54
- 6 Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474, 2014. 2
- 7 Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 103–128, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-17653-2_4. 2
- 8 Itay Berman, Ron D. Rothblum, and Vinod Vaikuntanathan. Zero-knowledge proofs of proximity. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPICs*, pages 19:1–19:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.ITCS.2018.19. 6
- 9 Manuel Blum. Coin flipping by telephone a protocol for solving impossible problems. *ACM SIGACT News*, 15(1):23–27, 1983. 22
- 10 Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford university press, 2013. 19

- 11 Amit Chakrabarti, Graham Cormode, Navin Goyal, and Justin Thaler. Annotations for sparse data streams. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 687–706, 2014. 2, 6
- 12 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Annotations in data streams. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikolettseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, volume 5555 of *Lecture Notes in Computer Science*, pages 222–234. Springer, 2009. doi:10.1007/978-3-642-02927-1_20. 6
- 13 Amit Chakrabarti, Graham Cormode, Andrew McGregor, and Justin Thaler. Annotations in data streams. *ACM Trans. Algorithms*, 11(1):7:1–7:30, 2014. doi:10.1145/2636924. 6, 50
- 14 Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian. Verifiable stream computation and Arthur–Merlin communication. In *30th Conference on Computational Complexity (CCC 2015)*, 2015. 2, 5, 6
- 15 Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian. Verifiable Stream Computation and Arthur–Merlin Communication. *SIAM Journal on Computing*, 48(4):1265–1299, January 2019. doi:10.1137/17M112289X. 5, 6, 8, 25, 48
- 16 Amit Chakrabarti and Prantar Ghosh. Streaming verification of graph computations via graph structure. *APPROX/RANDOM 2019, September 20-22, 2019*, 2019. 2
- 17 Amit Chakrabarti, Prantar Ghosh, and Justin Thaler. Streaming verification for graph problems: Optimal tradeoffs and nonlinear sketches. *arXiv preprint*, 2020. arXiv:2007.03039. 2, 6
- 18 Alessandro Chiesa, Michael A. Forbes, Tom Gur, and Nicholas Spooner. Spatial isolation implies zero knowledge even in a quantum world. *Journal of The ACM*, 69(2):15:1–15:44, 2022. doi:10.1145/3511100. 6
- 19 Alessandro Chiesa and Tom Gur. Proofs of proximity for distribution testing. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, 2018. 6
- 20 Graham Cormode. Applications of sketching and pathways to impact. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS '23*, pages 5–10, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3584372.3589937. 1
- 21 Graham Cormode and Chris Hickey. Cheap checking for cloud computing: Statistical analysis via annotated data streams. In *AISTATS*, 2018. 2, 6
- 22 Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Practical verified computation with streaming interactive proofs. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 90–112, 2012. 2, 3, 6
- 23 Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Streaming graph computations with a helpful advisor. *Algorithmica. An International Journal in Computer Science*, 65(2):409–442, 2013. 2, 6
- 24 Graham Cormode, Justin Thaler, and Ke Yi. Verifying computations with streaming interactive proofs. *Proc. VLDB Endow.*, 5(1):25–36, 2011. doi:10.14778/2047485.2047488. 2, 6, 20, 24, 50
- 25 Marcel Dall’Agnol, Tom Gur, Subhayan Roy Moulik, and Justin Thaler. Quantum proofs of proximity. *Quantum*, 6:834, October 2022. doi:10.22331/q-2022-10-13-834. 6
- 26 Samira Daruki, Justin Thaler, and Suresh Venkatasubramanian. Streaming verification in data analysis. In *International Symposium on Algorithms and Computation*, pages 715–726, 2015. 2, 6
- 27 Yevgeniy Dodis, Willy Quach, and Daniel Wichs. Authentication in the bounded storage model. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022 – 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 737–766. Springer, 2022. doi:10.1007/978-3-031-07082-2_26. 6

- 28 Yevgeniy Dodis, Willy Quach, and Daniel Wichs. Speak much, remember little: Cryptography in the bounded storage model, revisited. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023 – 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23–27, 2023, Proceedings, Part I*, volume 14004 of *Lecture Notes in Computer Science*, pages 86–116. Springer, 2023. doi:10.1007/978-3-031-30545-0_4. 6
- 29 Prantar Ghosh. New verification schemes for frequency-based functions on data streams. In Nitin Saxena and Sunil Simon, editors, *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2020, December 14–18, 2020, BITS Pilani, K K Birla Goa Campus, Goa, India (Virtual Conference)*, volume 182 of *LIPICs*, pages 22:1–22:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.FSTTCS.2020.22. 6
- 30 Oded Goldreich. Zero-Knowledge twenty years after its invention. *IACR Cryptol. ePrint Arch.*, 2002:186, 2002. 2
- 31 Oded Goldreich. Computational complexity: A conceptual perspective. *ACM Sigact News*, 39(3):35–39, 2008. 2
- 32 Oded Goldreich and Tom Gur. Universal locally verifiable codes and 3-round interactive proofs of proximity for CSP. *Theor. Comput. Sci.*, 878–879:83–101, 2021. doi:10.1016/j.tcs.2021.05.030. 6
- 33 Oded Goldreich, Tom Gur, and Ron D. Rothblum. Proofs of proximity for context-free languages and read-once branching programs. *Inf. Comput.*, 261:175–201, 2018. doi:10.1016/j.ic.2018.02.003. 6
- 34 Shafi Goldwasser, Yael Tauman Kalai, and Guy N Rothblum. Delegating computation: Interactive proofs for muggles. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 113–122, 2008. 3, 6, 21, 50
- 35 Jiaxin Guan and Mark Zhandry. Simple schemes in the bounded storage model. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019 – 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 500–524. Springer, 2019. doi:10.1007/978-3-030-17659-4_17. 6
- 36 Tom Gur. *On Locally Verifiable Proofs of Proximity*. PhD thesis, The Weizmann Institute of Science (Israel), 2017. 6
- 37 Tom Gur, Yang P. Liu, and Ron D. Rothblum. An exponential separation between MA and AM proofs of proximity. *Comput. Complex.*, 30(2):12, 2021. doi:10.1007/s00037-021-00212-3. 6
- 38 Tom Gur and Ran Raz. Arthur–Merlin streaming complexity. *Information and Computation*, 243:145–165, 2015. 6, 48
- 39 Tom Gur and Ron D. Rothblum. A hierarchy theorem for interactive proofs of proximity. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9–11, 2017, Berkeley, CA, USA*, volume 67 of *LIPICs*, pages 39:1–39:43. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICs.ITCS.2017.39. 6
- 40 Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. *Comput. Complex.*, 27(1):99–207, 2018. doi:10.1007/s00037-016-0136-9. 6
- 41 Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory, 2012. 18, 34
- 42 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. 4
- 43 Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography. In *30th Annual Symposium on Foundations of Computer Science*, pages 230–235, 1989. 4

- 44 Yuval Ishai and Mor Weiss. Probabilistically checkable proofs of proximity with zero-knowledge. In Yehuda Lindell, editor, *Theory of Cryptography – 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 121–145. Springer, 2014. doi:10.1007/978-3-642-54242-8_6. 6
- 45 Andrew McGregor. Graph stream algorithms: A survey. *Sigmod Record*, 43(1):9–20, May 2014. doi:10.1145/2627692.2627694. 1
- 46 Shanmugavelayutham Muthukrishnan. *Data Streams: Algorithms and Applications*. Now Publishers Inc, 2005. 1
- 47 Moni Naor. Bit commitment using pseudorandomness. *Journal of cryptology*, 4(2):151–158, 1991. 4
- 48 Michael O Rabin. *Fingerprinting by Random Polynomials*. Center for Research in Computing Techn., Aiken Computation Laboratory, Univ., 1981. 24
- 49 Anup Rao and Amir Yehudayoff. *Communication Complexity: And Applications*. Cambridge University Press, 2020. 5, 10, 26, 62
- 50 Omer Reingold, Guy N Rothblum, and Ron D Rothblum. Constant-round interactive proofs for delegating computation. *SIAM Journal on Computing*, 50(3):STOC16–255, 2019. 50
- 51 Guy N. Rothblum and Ron D. Rothblum. Batch verification and proofs of proximity with polylog overhead. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography – 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 108–138. Springer, 2020. doi:10.1007/978-3-030-64378-2_5. 6, 50
- 52 Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: Delegating computation in sublinear time. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC’13, Palo Alto, CA, USA, June 1-4, 2013*, pages 793–802. ACM, 2013. doi:10.1145/2488608.2488709. 6, 50
- 53 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of The Acm*, 27(4):701–717, October 1980. doi:10.1145/322217.322225. 24
- 54 Justin Thaler. Time-optimal interactive proofs for circuit evaluation. In *Advances in Cryptology–CRYPTO 2013*, pages 71–89. Springer, 2013. 2
- 55 Justin Thaler. Semi-streaming algorithms for annotated graph streams. *arXiv preprint*, 2014. arXiv:1407.3462. 2, 6
- 56 Salil Vadhan. The complexity of zero knowledge. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 52–70, 2007. 2
- 57 Salil Pravin Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, 1999. 2

A Deferred proofs

A.1 Proof of Proposition 21

► **Proposition 53** (Proposition 21, restated). *Any one-way communication protocol for SEARCH-INDEX with input $(x, j) \sim \Gamma^p \times [p]$ that sends an s -bit message succeeds with probability at most $\frac{1}{|\Gamma|} + O\left(\sqrt{s/p}\right)$.*

Proof. Define, for ease of notation, $\gamma = |\Gamma|$. We follow the strategy used in [49] for the binary case. First, note that by the minimax theorem we may assume Alice’s and Bob’s strategies are deterministic; i.e., that Alice sends $A(x) \in \{0, 1\}^s$ and Bob outputs $B(A(x), j) \in \Gamma$ for some functions A and B .

Let λ be the distribution of Alice’s message $A = A(x)$ induced by the (uniform) distribution of x , partitioning Γ^p into $\{P_a\}$ where $P_a = A^{-1}(a) = \{x \in \Gamma^p : A(x) = a\}$. Note that the distribution of x conditioned on $A = a$ is uniform over P_a , and that $\mathbb{P}_{A \sim \lambda}[A = a] = |P_a|/\gamma^p$. Then,

$$\begin{aligned} \mathbb{P}_{x \sim \Gamma^p}[\text{Bob outputs } x_j] &= \sum_{a \in \{0,1\}^s} \mathbb{P}_{x \sim \Gamma^p}[A(x) = a] \cdot \mathbb{P}_{x \sim \Gamma^p}[b(a, j) = x_j \mid A(x) = a] \\ &= \sum_{a \in \{0,1\}^s} \mathbb{P}_{A \sim \lambda}[A = a] \cdot \mathbb{P}_{x \sim P_a}[b(a, j) = x_j] \\ &= \mathbb{E}_{A \sim \lambda} \left[\mathbb{P}_{x \sim P_A}[b(A, j) = x_j] \right] \\ &\leq \mathbb{E}_{A \sim \lambda} \left[\max_{\alpha \in \Gamma} \{\mathbb{P}_{x \sim P_A}[x_j = \alpha]\} \right], \end{aligned} \quad (14)$$

so that we only need to bound the latter expression; note that the inequality shows Bob’s optimal strategy is to output the most frequent symbol at the j^{th} coordinate in P_A .

Now, define μ as the uniform distribution over Γ and $\mu_{i,a}$ as the distribution of x_i when $x \sim P_a$ (i.e., the distribution of x_i when $x \sim \Gamma^p$ conditioned on $A(x) = a$). Then, by Pinsker’s inequality (Equation 10), for all $a \in \text{Im } A$ and $i \in [p]$ we have

$$\|\mu_{i,a} - \mu\|^2 \leq \frac{\text{KL}(\mu_{i,a} \parallel \mu)}{2 \ln 2}$$

(where we use $\|\cdot\|$ as shorthand for the 2-norm $\|\cdot\|_2$). Since the inequality holds for all a and i , then it also holds for the convex combination corresponding to taking $A \sim \lambda$ and $j \sim [p]$ independently (i.e., whose coefficients are $\mathbb{P}[A = a, j = i] = \frac{|P_a|}{\gamma^p p}$). Therefore,

$$\begin{aligned} \mathbb{E}_{A \sim \lambda} \left[\mathbb{E}_{j \sim [p]} \left[\|\mu_{j,A} - \mu\|^2 \right] \right] &= \frac{1}{p} \sum_{i=1}^p \mathbb{E}_{A \sim \lambda} \left[\|\mu_{i,A} - \mu\|^2 \right] \\ &\leq \frac{1}{2p \ln 2} \sum_{i=1}^p \mathbb{E}_{A \sim \lambda} [\text{KL}(\mu_{i,A} \parallel \mu)] \\ &= \frac{1}{2p \ln 2} \sum_{i=1}^p I(A : x_i), \end{aligned}$$

where the last equality follows by the definition of mutual information (Equation 11). By convexity of $z \mapsto z^2$, we have

$$\begin{aligned} \mathbb{E}_{A \sim \lambda} \left[\mathbb{E}_{j \sim [p]} \left[\|\mu_{j,A} - \mu\|^2 \right] \right] &\leq \mathbb{E}_{A \sim \lambda} \left[\|\mu_{j,A} - \mu\|^2 \right] \\ &\leq \frac{1}{2p \ln 2} \sum_{i=1}^p I(A : x_i). \end{aligned}$$

Recall that $\mu_{i,a}(\alpha) = \mathbb{P}_{x \sim P_a}[x_i = \alpha]$. Comparing this value with the average mass $1/\gamma$, we have

$$\begin{aligned}
 \mathbb{E}_{\substack{A \sim \lambda \\ j \sim [p]}} \left[\max_{\alpha \in \Gamma} \{ \mathbb{P}_{x \sim P_A} [x_j = \alpha] \} \right] - \frac{1}{\gamma} &= \mathbb{E}_{\substack{A \sim \lambda \\ j \sim [p]}} \left[\max_{\alpha \in \Gamma} \left\{ \mu_{j,A}(\alpha) - \frac{1}{\gamma} \right\} \right] \\
 &\leq \mathbb{E}_{\substack{A \sim \lambda \\ j \sim [p]}} \left[\max_{\alpha \in \Gamma} \left\{ \left| \mu_{j,A}(\alpha) - \frac{1}{\gamma} \right| \right\} \right] \\
 &\leq \mathbb{E}_{\substack{A \sim \lambda \\ j \sim [p]}} [\| \mu_{j,A} - \mu \|] \\
 &\leq \sqrt{\frac{\sum_{i=1}^p I(A : x_i)}{2p \ln 2}},
 \end{aligned}$$

so that using Equation 14 and rearranging,

$$\mathbb{P}_{\substack{x \sim \Gamma^p \\ j \sim [p]}} [\text{Bob outputs } x_j] \leq \frac{1}{\gamma} + \sqrt{\frac{\sum_{i=1}^p I(A : x_i)}{2p \ln 2}}.$$

The theorem thus reduces to showing $\sum_{i=1}^p I(A : x_i) \leq s$. By standard information-theoretic equivalences and inequalities,

$$\begin{aligned}
 \sum_{i=1}^p I(A : x_i) &= \sum_{i=1}^p (H(x_i) - H(x_i|A)) && \text{(by Equation 11)} \\
 &= H(x) - \sum_{i=1}^p H(x_i|A) && \text{(by Equation 7)} \\
 &\leq H(x) - \sum_{i=1}^n H(x_i|x_1, \dots, x_{i-1}, A) && \text{(by Equation 6)} \\
 &= H(x) - H(x|A) && \text{(by Equation 8)} \\
 &= I(A : x) \leq H(A) && \text{(by Equation 11)} \\
 &\leq s && \text{(by Equation 5)}
 \end{aligned}$$

and the result follows. \blacktriangleleft

A.2 Proof of Theorem 24

► **Theorem 54** (Theorem 24, restated). *Figure 6 (algebraic-commit) and Figure 5 (decommit) form a streaming commitment protocol with space complexity $s = O((\ell + m) \log q)$ if $p = q^{3\ell}$ and $dm = \text{polylog}(q)$. The scheme is secure against $\text{poly}(s)$ -space adversaries and communicates $O(\ell q^{3\ell} \log q)$ bits.*

Furthermore, if each linear coefficient can be computed in $O(m \log q)$ space, then $s = O(m \log q)$.

Proof. We follow the same steps of Theorem 22, beginning with the binding property: using $y^{(i)}$ to denote the i^{th} column of y , when P is honest, i.e., sends the correction tuple $\gamma = \alpha - y^{(k)}$ in the commit stage and the polynomial $\hat{z}_{|L}$ where $z = \beta \cdot y$ in the decommit stage, then V accepts as $\hat{z}_{|L}(\rho) = \hat{z}(\rho) = \hat{y}(\rho, \beta)$ and $\hat{z}_{|L}(0) + \gamma = z_k + \beta \cdot \gamma = \alpha \cdot \beta$. (Recall that the line L is such that $L(0) = k$ and $L(\rho) = \rho$.)

Now, suppose P replies with a polynomial g such that $g(0) \neq \sum_{i \in [l]} \beta_i y_{ik} = z_k = \hat{z}_{|L}(0)$; then the Schwartz-Zippel lemma implies $g(\rho) \neq \hat{z}_{|L}(\rho)$ except with probability $dm/q = o(1)$, in which case V rejects. As the verifier only needs to store the evaluation point $\rho \in \mathbb{F}^m$, the coordinate $k \in [p]$ and a constant number of additional field elements, its space complexity

is $O(m \log q)$ as long as each β_i can be computed in this space (e.g., when $\beta_i = \beta_i(\rho)$ is the evaluation of an m -variate polynomial over \mathbb{F}); if β must be stored in its entirety, the complexity becomes $O((\ell + m) \log q)$.

To show the hiding property, assume towards contradiction that there exists a streaming algorithm D with space $\text{poly}(s) = \text{poly}(\ell, \log q)$ that distinguishes commitments between some $\alpha \in \mathbb{F}^\ell$ and $\alpha' \in \mathbb{F}^\ell \setminus \{\alpha\}$ with constant bias: that is,

$$\mathbb{P}_{\substack{y \sim \mathbb{F}^{\ell \times p} \\ k \sim [p]}} \left[D(y, \alpha - y^{(k)}, k) \text{ accepts} \right] - \mathbb{P}_{\substack{y \sim \mathbb{F}^{\ell \times p} \\ k \sim [p]}} \left[D(y, \alpha' - y^{(k)}, k) \text{ accepts} \right] \geq \varepsilon$$

for some $\varepsilon = \Omega(1)$. Now consider the following one-way communication protocol for SEARCH-INDEX over the alphabet \mathbb{F}^ℓ with input $(x, j) \in (\mathbb{F}^\ell)^p \times [p]$: Alice, viewing x as an element of $\mathbb{F}^{\ell \times p}$, simulates D on the stream (x, γ) , where $\gamma \sim \mathbb{F}^\ell$, and sends the $\text{polylog}(p)$ -bit snapshot of D to Bob, who finishes the simulation with j ; if D accepts output $\alpha - \gamma$, and otherwise output $\alpha' - \gamma$. Note that Bob outputs correctly exactly when $\gamma = \alpha - y^{(k)}$ and D accepts, or $\gamma = \alpha' - y^{(k)}$ and D rejects. We will now show that the protocol solves SEARCH-INDEX with a bias that is too large, contradicting Proposition 21.

$$\begin{aligned} & \mathbb{P}_{\substack{x \sim (\mathbb{F}^\ell)^p \\ j \sim [p]}} [\text{Bob outputs } x_j] \\ &= \frac{1}{q^\ell} \cdot \mathbb{P}_{\substack{x \sim (\mathbb{F}^\ell)^p \\ j \sim [p]}} [D(x, \alpha - x_j, j) \text{ accepts}] + \frac{1}{q^\ell} \cdot \mathbb{P}_{\substack{x \sim (\mathbb{F}^\ell)^p \\ j \sim [p]}} [D(x, \alpha' - x_j, j) \text{ rejects}] \\ &= \frac{1}{q^\ell} \left(1 + \mathbb{P}_{\substack{x \sim (\mathbb{F}^\ell)^p \\ j \sim [p]}} [D(x, \alpha - x_j, j) \text{ accepts}] - \mathbb{P}_{\substack{x \sim (\mathbb{F}^\ell)^p \\ j \sim [p]}} [D(x, \alpha' - x_j, j) \text{ accepts}] \right) \\ &\geq \frac{1 + \varepsilon}{q^\ell} \\ &= \frac{1}{q^\ell} + \Omega\left(\frac{1}{q^\ell}\right). \end{aligned}$$

Since $q^{-\ell} = \Omega\left(\sqrt{q^\ell/p}\right) = \omega\left(\sqrt{\text{poly}(s)/p}\right)$, owing to $s = \text{poly}(\ell, \log q)$, the result follows. The communication complexity of the protocols is dominated by the prover sending ℓp field elements, for a total of $O(\ell q^{3\ell} \log q)$ bits. \blacktriangleleft

A.3 Proof of Claim 30

\triangleright **Claim 55 (Claim 30, restated).** Let $p, q \in [0, 1]^v$ be probability vectors and $t \in [v]$ a positive integer. There exists a set $C \subseteq [v]$ of size t such that $\sum_{i \in [v] \setminus C} p_i q_i \leq 1/t$.

Proof. We reduce the claim to proving an upper bound on a certain optimisation problem. Namely, let $\Delta = \{x \in [0, 1]^v : \sum_i x_i = 1\}$ and $\Delta' = \Delta \cap \{x \in [0, 1]^v : x_1 \geq \dots \geq x_v\}$ be the v -dimensional simplex and the simplex with ordered coordinates, respectively. Define the function $f : \Delta' \times \Delta \rightarrow \mathbb{R}_+$ by $f(p, q) = \sum_{i=1}^v i p_i q_i$.

Under the assumption that $f(p, q) \leq 1$ for all $p \in \Delta'$ and $q \in \Delta$, we conclude as follows: since $p_1 \geq p_2 \geq \dots \geq p_v$ without loss of generality (permuting the vectors to satisfy the condition does not affect the truth of the claim), for any $t \in [v]$

$$1 \geq f(p, q) = \sum_{i=1}^v \left(\sum_{j=i}^v p_j q_j \right) \geq \sum_{i=1}^t \left(\sum_{j=i}^v p_j q_j \right)$$

implies the existence of $i \in [t]$ such that $\sum_{j=i}^v p_j q_j \leq 1/t$. Taking $C = [i - 1]$ completes the proof.

2:66 Streaming Zero-Knowledge Proofs

We now proceed to show $f(p, q) \leq 1$. Since f is continuous with compact domain, there exists a pair (p^*, q^*) that maximises f . Let $\ell \in [v]$ be the largest nonzero coordinate of p^* . Then $q_i^* > 0$ for all $i \leq \ell$, as otherwise moving the mass p_i^* onto p_1^* would contradict maximality; and $q_i^* = 0$ for all $i > \ell$, or moving q_i^* onto (say) q_1^* likewise leads to a contradiction.

Now, suppose (towards contradiction) $\ell > 1$, take $1 < j \leq \ell$ and consider the pair (p^*, q') with $q'_1 = 0$, $q'_i = q_1^* + q_i^*$ and $q'_j = q_j^*$ otherwise. Then $f(p^*, q') \leq f(p^*, q^*)$ implies

$$ip_i^*(q_1^* + q_i^*) \leq p_1^*q_1^* + ip_i^*q_i^*,$$

and thus $ip_i^* \leq p_1^*$ (since $q_1^* \neq 0$). But then

$$f(p^*, q^*) = \sum_{i=1}^{\ell} ip_i^*q_i^* \leq p_1^* \sum_{i=1}^{\ell} q_i^* = p_1^* < 1,$$

a contradiction, as the delta distributions at 1 achieve value 1.

We thus conclude that $\ell = 1$, so the maximisers p^*, q^* are the delta distributions at 1 and $f(p, q) \leq f(p^*, q^*) = 1$, as desired. \triangleleft

Solving Unique Games over Globally Hypercontractive Graphs

Mitali Bafna  

Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, USA

Dor Minzer  

Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

We study the complexity of affine Unique-Games (UG) over *globally hypercontractive graphs*, which are graphs that are not small set expanders but admit a useful and succinct characterization of all small sets that violate the small-set expansion property. This class of graphs includes the Johnson and Grassmann graphs, which have played a pivotal role in recent PCP constructions for UG, and their generalizations via high-dimensional expanders.

We show new rounding techniques for higher degree sum-of-squares (SoS) relaxations for worst-case optimization. In particular, our algorithm shows how to round “low-entropy” pseudodistributions, broadly extending the algorithmic framework of [5]. At a high level, [5] showed how to round pseudodistributions for problems where there is a “unique” good solution. We extend their framework by exhibiting a rounding for problems where there might be “few good solutions”.

Our result suggests that UG is easy on globally hypercontractive graphs, and therefore highlights the importance of graphs that lack such a characterization in the context of PCP reductions for UG.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Complexity theory and logic

Keywords and phrases unique games, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.3

Related Version *Full Version:* <https://arxiv.org/abs/2304.07284> [10]

Funding *Dor Minzer:* Supported by a Sloan Research Fellowship, NSF CCF award 2227876 and NSF CAREER award 2239160.

1 Introduction

The main goal of this paper is to design efficient algorithms that solve instances of the Unique-Games problem whose underlying graph is globally hypercontractive graphs, an extension of the class of small set expanders. The motivation for our investigation is three-fold.

Candidate hard instances for Unique-Games

Recent progress towards the UGC [31, 21, 22, 32] showed that it is NP-hard to distinguish $1/2$ -satisfiable instances of UG from ε -satisfiable instances. These works crucially relied on the use of globally hypercontractive graphs. Our algorithms allow us to examine the hard instances arising from their reduction. We try to identify the source of hardness and thus suggest a natural class of graphs that might be hard for UG (Section 1.2.1).

New rounding techniques for Higher Degree SoS

In doing so we build new rounding techniques for higher degree SoS. The study of algorithms for UG has led to the development of general algorithmic techniques such as sophisticated graph partitioning tools [1] and new rounding techniques for SoS [16, 13]. These techniques



© Mitali Bafna and Dor Minzer;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 3; pp. 3:1–3:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



have in turn led to breakthroughs in robust statistics [14, 37, 12] and other average case problems [17]. The setting of SoS for worst-case optimization is much less understood though. We give new techniques that might be useful for other problems in the worst case setting. We elaborate on our rounding techniques in Section 1.2.2.

The emergence of Unique-Games instances in other contexts

Unique-Games instances on structured graphs naturally appear in other contexts in theoretical computer science, and the tools developed by trying to design algorithms are often helpful. In the context of the current paper, the type of Unique-Games instance we study turn out to be crucial in the field of high-dimensional expanders. Indeed, in [9, 18] the authors investigate a conjecture due to Dinur and Kaufman [20], which asks whether one can construct sparse, low soundness, 2-query direct product testers using high dimensional expanders. The results of [9, 18] assert that a sufficient condition for a high-dimensional expander to admit such direct product testers is the existence of certain local algorithm to approximate affine instance of Unique-Games defined on graphs associated with the complex; the authors refer to this property as UG coboundary expansion. In a follow-up work in progress, the authors and Lifshitz [8] have constructed complexes that are UG coboundary expanders, and some of the ideas developed herein are crucial. See Section 1.2.3 for more details.

1.1 Unique-Games

The Unique Games Conjecture (UGC in short) is a central open problems in Complexity Theory [27]. In short, the UGC says that distinguishing between almost satisfiable (value $\geq 1 - \varepsilon$) and highly unsatisfiable (value $\leq \varepsilon$) instances of a certain 2-variable constraint satisfaction problem (CSP) called *Unique Games* is NP-hard. The primary reason for the interest in UGC is that, if true, it implies a large number of hardness of approximation results that are often times tight [33, 29, 4, 34, 40] (see [28, 43]). One of the most striking consequences of UGC is that it implies that a class of semi-definite programs (SDP), namely the basic SDP or degree 2 Sum-of-Squares, achieves the best possible approximation ratio (among all efficient algorithms) for all CSPs [40].

► **Definition 1.** *A instance of Unique-Games Ψ consists of a graph $G = (V, E)$, a finite alphabet Σ and a collection of constraints, $\Phi = \{\Phi_e\}_{e \in E}$, one for each edge in G . For all $e \in E$, the constraint Φ_e takes the form $\Phi_e = \{(\sigma, \phi_e(\sigma)) \mid \sigma \in \Sigma\}$, where $\phi_e: \Sigma \rightarrow \Sigma$ is a 1-to-1 map.*

The goal in the Unique-Games problem is to find an assignment $A: V \rightarrow \Sigma$ that satisfies the maximum number of constraints possible, that is, satisfies that $(A(u), A(v)) \in \Phi_e$ for the largest number of edges $e = (u, v) \in E$ as possible. We define the value of the instance Ψ by:

$$\text{val}(\Psi) = \max_{A: V \rightarrow \Sigma} \frac{\#\{e \mid A \text{ satisfies } e\}}{|E|}.$$

With this in mind, the Unique-Games Conjecture is the following statement:

► **Conjecture 2.** *For all $\varepsilon, \delta > 0$ there is $k \in \mathbb{N}$ such that given a Unique-Games instance Ψ with alphabet size at most k , it is NP-hard to distinguish between:*

YES case: $\text{val}(\Psi) \geq 1 - \varepsilon$.

NO case: $\text{val}(\Psi) \leq \delta$.

It turns out that the topology of the underlying graph G plays a crucial role in the complexity of the UG instance defined over it. In particular, it turns out that UG over expander graphs is easy:

► **Definition 3.** Given a regular graph $G = (V, E)$ and a set of vertices $S \subseteq V$, the edge expansion of S is defined by:

$$\Phi(S) = \Pr_{u \in S, v \in \Gamma(u)} [v \notin S].$$

The results of [3, 38, 2] assert that UG instances with completeness close to 1 over expanders are easy. A graph G is called a (γ, ξ) -small set-expander (SSE) if for every $S \subseteq V$ of size at most $\xi|V|$ it holds that $\Phi(S) \geq \gamma$. In [5], it is shown that UG is easy over “certifiable” small-set expanders, that in fact captures all currently known small-set expanders. Thus to find hard instances of UG, one must look beyond graphs that are expanders and small set expanders.

1.1.1 NP-hardness Reduction for 2-2 Games and Global Hypercontractivity

Indeed, recent progress towards UGC [31, 21, 22, 32] has utilized graphs which are not small-set expanders. In these works it is proved that 2-to-1-Games are NP-hard (which is a very similar problem to UG, except that each one of the maps ϕ_e defining the constraints is a 2-to-1 map). This implies that for all $\varepsilon > 0$, given a UG instance Ψ over sufficiently large alphabet, it is NP-hard to distinguish between the case that $\text{val}(\Psi) \geq 1/2$ and the case that $\text{val}(\Psi) \leq \varepsilon$. To prove these results, these works use graphs that are not small set expanders in two different ways:

1. Smooth Parallel Repetition: A key step in the reduction of [31, 21, 22, 32] is an application of the Parallel Repetition Theorem [42] to get a hardness result for a sufficiently smooth outer PCP construction. Roughly speaking, this step in the process may be associated with the Johnson graph with a *large intersection* parameter. That is, with the graph $J(n, \ell, t)$ in which the vertices are $\binom{[n]}{\ell}$, and two vertices A and B are adjacent if $|A \cap B| = \ell - t$, and we think of t as much smaller than ℓ (say, $t = \sqrt{\ell}$).
2. Composition with the Grassmann encoding: The Grassmann encoding is an encoding of linear functions based on the Grassmann graph $\text{Grass}(n, \ell)$ over \mathbb{F}_2 . The Grassmann graph over \mathbb{F}_2 is the graph whose vertices are all ℓ -dimensional subspaces of \mathbb{F}_2^n , denoted by $\binom{[n]}{\ell}$, and two vertices L and L' are adjacent if $\dim(L \cap L') = \ell - 1$.

Both of the graphs above, namely the Johnson graph with large intersection sizes, as well as the Grassmann graph, are not small set expanders. However very importantly, the class of small sets in the Grassmann graph with bad expansion has a succinct and intuitive characterization, and the proof of the 2-to-1 Games Theorem heavily relies on this characterization.

Though the term is not formally defined, we refer to graphs such as the Grassmann graph above as globally hypercontractive graphs. By that, we mean that there is a collection of “obviously-non-expanding local sets”, such that any small set that doesn’t expand well must have a large intersection with one of the sets from the collection (see the full version of the paper for a semi-formal definition). Aside from the Grassmann graph, this class of graphs includes Johnson graphs with small intersection sizes [30], certain Cayley graphs over the symmetric group [23], p -biased cubes for $p = o(1)$, other product domains [26] as well as high dimensional expanders [24, 7].

Thus a natural approach to isolating hard instances of UG is to study the complexity of UG over graphs that are globally hypercontractive, in particular the Johnson and Grassmann graphs. The study of this question was initiated in [5] for the class of *Affine Unique-Games* (Definition 4) over Johnson graphs. Unfortunately their algorithm gave parameters that were insufficient to shed light on the source of hardness in reduction above and therefore to derive any of the consequences of our results (see full version).

1.2 Our Results

Our results improve upon [5] in two main aspects: we are able to deal with instances with arbitrarily small (but constant) completeness, and most importantly, their algorithm gets a soundness guarantee that degrades with other parameters of the graph (which in all PCP constructions grow with the alphabet size), whereas ours doesn't. To describe our results we start with the definition of Affine Unique Games.

► **Definition 4.** *An instance of Affine-UG is an instance of Unique-Games in which the alphabet is the ring of integers modulo q , \mathbb{Z}_q , and all of the constraint maps ϕ_e are affine shifts, that is, ϕ_e of the form $\phi_e(\sigma) = \sigma + b_e$ for some $b_e \in \mathbb{Z}_q$.*

An equivalent but slightly different way to view the Affine-UG problem is as a system of linear equations (X, E) over \mathbb{Z}_q . Each equation in E is of the form $x_i - x_j = b$ where $x_i, x_j \in X$ are variables and $b \in \mathbb{Z}_q$ is some constant. Despite looking very restrictive, it is known [29] that the UGC is true if and only if it holds for the class of Affine UG and furthermore this class captures many interesting optimization problems such as Max-Cut and graph coloring, thus we shall focus our attention on Affine UG henceforth.¹

Our main result asserts that there is a polynomial time algorithm for solving Affine UG over globally hypercontractive graphs. As the term globally hypercontractive graph is not formally defined, below are some concrete instances of graphs on which this applies. In the full version we give a semi-formal definition of globally hypercontractive graphs and also show how our algorithm and analysis can be abstracted to solve UG on such graphs, as long as one is provided with an SoS certificate of global hypercontractivity.

We first consider the Johnson graph with small intersection sizes, which we henceforth refer to as the noisy-Johnson graph. This is the regime in which a characterization theorem for non-expanding sets holds. Formally the “ α -noisy” Johnson graph is $J(n, \ell, t)$ in the case that $t = \alpha\ell$, for $\alpha \in (0, 1)$ bounded away from 0 and thought of as a fixed constant independent of ℓ . The first result for Johnson graphs addresses the case that the completeness of the instance is close to 1, in which case our algorithm matches the guarantee of the algorithm of [5] for certifiable small-set expanders, and in particular the α -noisy hypercube graph:

► **Theorem 5.** *There is $\varepsilon_0 > 0$ such that for all $\alpha \in (0, 1)$ the following holds for all $0 < \varepsilon \leq \varepsilon_0$. There exists an algorithm whose running time is $n^{\text{poly}(\ell, |\Sigma|, 1/\varepsilon)}$ which, on input Ψ which is an affine UG instance over $J(n, \ell, \alpha\ell)$ promised to be at least $(1 - \varepsilon)$ -satisfiable, finds an assignment that satisfies at least $2^{-O(\frac{\sqrt{\varepsilon}}{\alpha})}$ -fraction of the constraints in Ψ .*

The second result addresses the case of UG instances with arbitrarily small (but bounded away from 0) completeness, in which case our algorithm satisfies a constant fraction of the constraints:

► **Theorem 6.** *For all $\alpha \in (0, 1)$ and $c > 0$, there is $\delta > 0$ such that the following holds. There exists an algorithm whose running time is n^D with $D = \ell^{\text{poly}(|\Sigma|\ell^{1/c})} 2$ which on input Ψ , an affine UG instance over $J(n, \ell, \alpha\ell)$ promised to be at least c -satisfiable, finds an assignment that satisfies at least δ -fraction of the constraints in Ψ .*

¹ We remark that the reduction of [29] does not preserve the topology of the graph. We are therefore not able to translate our results directly to the class of general UG, and believe this is an interesting direction for further study.

² We note that we have not optimized for D and the $\exp(\ell)$ -dependence arises due to the degree of the SoS proofs. We used a blackbox statement to convert some of the proofs to SoS proofs, and we conjecture that one can in fact improve the SoS degree to $O(\ell)$ when done carefully.

We remark that the soundness guarantee in the theorems above does not depend on ℓ (when $\alpha = \Omega(1)$), which in most PCP constructions grows with the alphabet size of the instance. But note that this guarantee degrades when $\alpha = o(1)$ and becomes useless if α depends on the alphabet size. In Section 1.2.1 below we discuss why this is interesting – in fact $o(1)$ -Noisy Johnson graphs are a natural candidate for hard instances of UG.

We can get similar results given any of the globally hypercontractive graphs mentioned earlier. Below we give a corollary for the Grassmann graph. We show that there is a polynomial time algorithm solving affine UG over the Grassmann graph, even on instances with small completeness:

► **Theorem 7.** *For all $c > 0$ there exists $\delta > 0$ such that the following holds. There exists an algorithm whose running time is n^D with $D = \ell^{\text{poly}(|\Sigma|\ell^{1/c})}$ which on input Ψ , an affine UG instance over $\text{Grass}(n, \ell)$ promised to be at least c -satisfiable, finds an assignment that satisfies at least δ -fraction of the constraints in Ψ .*

Note that since the spectral gap of the Grassmann graph is $1/2$, UG algorithms over expanders already imply Theorem 7 for $c \gg 1/2$. Thus, the main contribution of Theorem 7 is the algorithm on Grassmann graphs that works for arbitrarily small completeness.

Below we state our result for random walks on high dimensional expanders (HDX), a large class of graphs that generalize the Johnson graphs but do not necessarily possess its strong symmetries. These include graphs stemming from cut-offs of [36]’s construction of Ramanujan complexes, or [25]’s construction of coset complex expanders. These graphs exhibit the nice high-dimensional expansion properties (e.g. global hypercontractivity) of the Johnson graphs yet are substantially different in other aspects, such as being of bounded degree.

► **Theorem 8.** *For all $\alpha \in (0, 1)$ and $c > 0$, there exists $\delta > 0$ such that the following holds. Let X be any d -dimensional two-sided γ -local-spectral expander with $\gamma \ll o_\ell(1)$ and $d > \ell$. There exists an algorithm whose running time is n^D with $D = \ell^{\text{poly}(|\Sigma|\ell^{1/c})}$ which on input Ψ , an affine UG instance over the canonical walk M on $X(\ell)$ of depth α , promised to be at least c -satisfiable, finds an assignment that satisfies at least δ -fraction of the constraints in Ψ .*

Since we have not defined any of the HDX terminology, let us note that this is indeed a generalization of Theorem 6. The Johnson graph corresponds to the complete complex X (which is the simplest instantiation of a two-sided local spectral expander), and the α -noisy Johnson graph $J(n, \ell, \alpha\ell)$ corresponds to a “canonical” random-walk on $X(\ell)$ that goes down $\alpha\ell$ -levels and comes back up randomly to $X(\ell)$ while ensuring that it changes exactly $\alpha\ell$ elements in a vertex. In fact, in the above theorem we can allow M to be any complete random walk on $X(\ell)$ and our soundness guarantee will only depend on c and certain parameters of M that are inherently independent of ℓ ³.

1.2.1 Candidate Hard Instances for Unique Games

Our results suggest that the hardness in the instances of UG obtained via the reduction of [31, 21, 22, 32] does not come from the Grassmann graph (which is globally hypercontractive), but rather from the smooth parallel repetition step. Recall that this step uses a Johnson

³ Concretely it depends on the stripped threshold rank of M above a certain threshold as defined in [6]. For example, when M is the canonical random walk with depth α on $X(\ell)$, and the completeness is $c = 1 - \varepsilon$, this quantity is $r(M) = O(\sqrt{\varepsilon}/\alpha)$ and our soundness guarantee is $\exp(-r)$, matching that of Theorem 5.

graph with a large intersection parameter ($J(n, \ell, \alpha \ell)$ with $\alpha \approx 0$), that is not globally-hypercontractive. Therefore combining the knowledge from the reduction and our algorithm we get that the α -noisy-Johnson graphs should be hard for UG when $\alpha = o(1)$ and become easy when α is bounded away from 0. This also explains why our soundness guarantee decays with α . Indeed, we would be able to make a stronger assertion provided that our results held for general UG (as opposed to only affine UG) or if the reduction above produced instances of Affine UG. Though we believe an algorithm for general UG should exist along the lines of our algorithm, we do not know how to prove so and leave this as an interesting direction to investigate.

Albeit, ignoring the subtlety between general and affine UG, this means that the $o(1)$ -noisy Johnson graphs and shallow random walks on HDX provide a natural candidate for constructing SoS lower bounds for UG.

1.2.2 New Rounding Scheme for Higher Degree SoS

Our algorithms are obtained via a novel rounding scheme and analysis for the standard higher degree Sum-of-Squares SDP relaxation for Unique Games. Raghavendra’s [40] groundbreaking result showing the optimality of the basic SDP for all CSPs under the UGC, led to efforts to refute the UGC using higher degree SoS relaxations [35, 39]. The study of SoS algorithms has since produced numerous algorithmic advances across many fronts: high-dimensional robust statistics [14, 37, 11, 12], quantum computation [15] and algorithms for semi-random models [17], to name a few. Most of these works use the sum-of-squares method for *average-case* problems though and unfortunately there remains a dearth of techniques for analysing higher degree SoS relaxations for *worst-case* optimization problems. The handful of techniques known for worst-case rounding are the *global correlation rounding* technique from [16, 41] and its generalization via reweightings in [15].

There is an intuitive reason for why this is the case: all aforementioned algorithms for average-case problems rely on a strong “uniqueness” property for the solution space. That is, given an average-case optimization problem, the key observation in the analysis is that the solution to the problem is *unique* upto small perturbations. These algorithms then proceed by converting a proof of uniqueness into an SoS algorithm for finding such a solution, via the proofs-to-algorithms framework for designing Sum-of-Squares algorithms [13].

Such strong uniqueness properties are too much to expect for worst-case problems. Recently [5] showed how to round UG instances on certifiable SSEs. The key property of such instances was a certain “weak uniqueness” of the solution space: any two solutions to the UG instance on an SSE are weakly correlated to each other, i.e. they “agree” on 1% of the vertices. [5] then exploited this observation to give a novel analysis of a higher degree SoS rounding.

For many worst-case problems though the solution space might not be so structured and in fact could allow for many distinct solutions. It turns out that this is precisely the case for UG on globally hypercontractive graphs. Our main technical contribution is to strengthen and broadly extend the [5] framework. At a high-level we show that in our case, the solution space is supported over “few good solutions”. That is, there is a small list of solutions such that every good solution is 1% correlated with one of these. We give a new rounding for higher degree SoS that exploits this “weak few good solutions” property. This turns out to be significantly more challenging than the case where we have “weak uniqueness”. We expect that with this strengthening, the framework of weak uniqueness to algorithms should be broadly applicable for other worst-case optimization problems. In Section 2 we provide a detailed overview of our techniques, starting out with the framework of [5].

1.2.3 The Emergence of Unique-Games Instances in Other Contexts

Affine instances of Unique-Games naturally appear in the context of high-dimensional expanders. For instance, given a graph $G = (V, E)$ and a labeling $\Pi: E \rightarrow \mathbb{F}_2$, one may think of (G, Π) as an instance of Unique-Games, wherein the goal is to find a labeling $A: V \rightarrow \mathbb{F}_2$ such that $A(u) - A(v) = \Pi(u, v)$ for as many edges $(u, v) \in E$ as possible. Note that if (u, v, w) is a triangle in G and $\Pi(u, v) + \Pi(v, w) + \Pi(w, u) \neq 0$, then no assignment can simultaneously satisfy all of the edges (u, v) , (v, w) and (w, u) . We call such triangles inconsistent triangles. The coboundary constant of G (with coefficients in \mathbb{F}_2) is defined as the ratio

$$\max_{\Pi} \frac{1 - \text{val}(G, \Pi)}{\text{fraction of inconsistent triangles in } G}.$$

The coboundary expansion of a graph (and its higher degree analogs for simplicial complexes) are important notions of topological expansion. These notions are inherently different from the more traditional spectral-type expansion notions studied for graphs (and simplicial complexes), and therefore they provide us additional understanding of graphs/ complexes. For instance, recently the works [9, 18] proved that spectral expansion of simplicial complexes is insufficient if one wishes to construct low soundness direct product testers. Instead, one needs spectral expansion as well as coboundary expansion with respect to some non-Abelian groups. The connection between Unique-Games and expansion is useful in studying these new notions of coboundary expansion, and in a recent work we use it to construct such coboundary expanders [8, 19].

1.3 Open Problems

We end this introductory section by stating a few open directions that are of interest for future research. Perhaps the most pertinent question that arises out of our work is whether one can build better integrality gaps for UG:

► **Problem 1.** *Can we get higher degree SoS lower bounds for UG using non-globally hypercontractive graphs such as the Johnson graph in the $o(1)$ -noise regime?*

The second problem asks whether our results continue to hold for non-affine unique games:

► **Problem 2.** *For globally hypercontractive graphs G such as the Johnson graph (with small intersection size) and the Grassmann graph, is there a polynomial time algorithm that given a UG instance Ψ over G with $\text{val}(\Psi) \geq 1 - \varepsilon$ (where $\varepsilon > 0$ is thought of as small), finds an assignment satisfying at least δ fraction of the constraints in Ψ ? How about the case that $\text{val}(\Psi) \geq c$, where c is bounded away from 1?*

The third problem asks whether there are other combinatorial optimization problems for which our techniques may yield improved algorithms. Informally, we show how to round SoS relaxations for problems that admit a few good solutions. We believe that this technique should be useful outside the context of UG – given any problem for which one can prove (in SoS) that there are only a “few good solutions”, one can apply similar rounding techniques to obtain one such solution.

► **Problem 3.** *Can one use the low-entropy rounding framework to get improved run-time for other combinatorial optimization problems, such as coloring 3-colorable graphs using as few colors as possible or improved subexponential time algorithms for Max-Cut?*

2 Overview of Our Techniques

We now elaborate on our techniques starting with the framework of [5]. They proposed a new technique for rounding relaxations of UG that have “low-entropy” measured via a function called the shift-partition size. Given two fixed assignments for the instance, their shift-partition size is roughly defined as the fraction of variables on which these assignments agree (upto symmetry). Taking the equivalent view of the SDP solution as a distribution \mathcal{D} over non-integral solutions, called a pseudodistribution, the expected shift-partition size of two random assignments drawn from \mathcal{D} is then roughly equal to an average of local collision probabilities under \mathcal{D} and thus a proxy for the *entropy* of \mathcal{D} . Their analysis proceeds by showing: (1) when the expected shift-partition size (equivalently collision probability) is large, one can round to a high-valued solution, and moreover (2) when the graph is a certifiable small-set expander, the pseudodistribution always has large shift-partition size! They were not able to extend this idea to get high-valued solutions for the broader class of globally hypercontractive graphs though, since in this case the pseudodistribution might be supported over multiple assignments and therefore does not have high collision probability. It turns out though that even in this harder case, the pseudodistribution $\mathcal{D} \times \mathcal{D}$ has large expected shift-partition size after *conditioning* on an event E . But they could not exploit this property since after conditioning the shift-partition could be large for trivial reasons⁴ and therefore is no longer a good proxy for the collision probability/entropy of the distribution \mathcal{D} .

To get around this barrier, we show that after a suitable preprocessing step on the pseudodistribution, one can in fact condition on any event E (with not too small probability) while preserving most of the desired local independence properties of the distribution. Thus, even after conditioning on E , the expected shift-partition size of $\mathcal{D} \times \mathcal{D} \mid E$ being large signifies that the pseudodistribution \mathcal{D} has high collision probability. One can then use a simple rounding procedure to obtain a high-valued UG solution. Conditioning pseudodistributions is one of the few ways we know of harnessing the power of higher-degree pseudodistributions, hence we believe that the idea of gaining structural control over the distribution after conditioning may be applicable in the analysis of other SoS algorithms too. To explain further details, we start by describing the approach of [5].

2.1 The Approach of [5]: Rounding analysis via the Shift Partition

Fix an Affine Unique-Games instance $\Psi = (G = (V, E), \mathbb{F}_q, \Phi)$. In the SoS relaxation of the Unique-Games problem we have a collection of variables $X_{v,\sigma}$, one for pair of vertex $v \in V$ and label to it $\sigma \in \Sigma$. The output of the program is a pseudoexpectation operator $\tilde{\mathbb{E}}$, which assigns to each monomial involving at most d of the variables a real-number, under which:

1. The value is high:

$$\tilde{\mathbb{E}} \left[\sum_{(u,v) \in E} \sum_{\sigma \in \Sigma} X_{v,\sigma} X_{u,\phi_{u,v}(\sigma)} \right] \geq c \cdot |E|.$$

2. $\tilde{\mathbb{E}}$ is a linear, positive semi-definite operator (when viewed as a matrix over $\mathbb{R}^{M \times M}$ where M is the set of monomials of degree at most $d/2$) satisfying various Booleanity constraints on $X_{u,\sigma}$.
3. Scaling: $\tilde{\mathbb{E}}[1] = 1$.

⁴ In the worst case, the event E could collapse the product distribution over two random assignments to set the second random assignment to be always equal to the first one. In this case a pair of assignments drawn from $\mathcal{D} \times \mathcal{D} \mid E$ being equal does not say anything about the collision probability of \mathcal{D} .

Morally, the pseudoexpectation $\tilde{\mathbb{E}}$ should be thought of in the following way: there is an unknown distribution \mathcal{D} over assignments A_1, \dots, A_m that each have value at least c . For the assignment A_i we think of Boolean valued assignment to the variables $X_{u,\sigma}$ that assigns to a variable 1 if and only if $A_i(u) = \sigma$, and associate with it the expectation operator \mathbb{E}_i which maps monomials to Boolean values in the natural way according to A_i . The operator $\tilde{\mathbb{E}}$ then is the average of the operators \mathbb{E}_i according to $i \sim \mathcal{D}$.⁵

Shift-partition

Given $\tilde{\mathbb{E}}$, one can construct a different pseudoexpectation operator that allows access to moments of two assignments $X = A_i, X' = A_j$ where $i, j \sim \mathcal{D}$ are chosen independently. In expectation, we get that at least c^2 fraction of the edges get satisfied by both X and X' ; the algorithm attempts to satisfy these edges. Towards this end, given two fixed assignments X and X' we define the shift-partition of the vertices of V : $V = \cup_{s \in \mathbb{F}_q} F_s$ where for each $s \in \mathbb{F}_q$ we define

$$F_s(X, X') = \{v \in V \mid X(v) - X'(v) = s\}.$$

The shift-partition size is then defined as:

$$\tilde{\mathbb{E}}_{X, X' \sim \mathcal{D}} \left[\sum_{s \in \Sigma} \left(\frac{|F_s(X, X')|}{|V(G)|} \right)^2 \right].$$

After rearranging, we get that when X and X' are independent, this expression is an average of some local collision probabilities (precisely $\mathbb{E}_{u,v}[CP(X_u - X_v)]$), and hence the shift-partition size being large in expectation turns out to be useful for rounding.

On the other hand, observe that if an edge $(u, v) \in E$ is satisfied by both X and X' , then $X(u) - X(v) = X'(u) - X'(v)$ and rearranging we conclude that u and v are in the same part F_s of the shift partition. We therefore conclude that in expectation over $X, X' \sim \mathcal{D}$ at least c^2 fraction of the edges of G stay inside the same part of the shift partition, implying that the expansion of the shift-partition is small.

Small-set expanders

If the graph G is a small-set expander, then the above implies that at least one of the sets F_s /the shift-partition size is large and the following rounding procedure works in such cases:

1. Sample a vertex $v \in V$ and choose $A(v) = \sigma$ according to the distribution $p(\sigma) = \tilde{\mathbb{E}}[X_{v,\sigma}]$.
2. For any $u \in V$, sample $A(u)$ according to the distribution $p(a) = \frac{\tilde{\mathbb{E}}[X_{u,a}X_{v,\sigma}]}{\tilde{\mathbb{E}}[X_{v,\sigma}]}$.

To get an understanding to why this rounding scheme works, think of X as fixed and X' as random. Thus, the fact that part s of the shift partition is large implies that $X' = X + s$ on a constant fraction of the vertices. Therefore, once we sampled the assignment to v in the first part of the algorithm, the value of s is determined. In the second step we are sampling the assignment to other nodes conditioned on the value of v . However, there is one value for u which is much more likely than others – namely $X(u) + s$, and so we can expect that $X'(u) = X(u) + s$ for a constant fraction of the vertices u . In particular, for any edge (u, w) inside F_s that is satisfied by X , we will have that the assignments sampled for u and w are

⁵ Formally speaking, when given $\tilde{\mathbb{E}}$ we are not guaranteed that there exists an actual distribution \mathcal{D} over good assignments as above, however this intuition will be good enough for the sake of this informal presentation.

3:10 Solving Unique Games over Globally Hypercontractive Graphs

$X(u) + s$ and $X(w) + s$ respectively with constant probability, in which case we manage to satisfy (u, w) . To analyse this rounding strategy formally, [5] crucially use the independence of X and X' .

In essence, the above asserts that the shift-partition being large implies that the solution space of X must have high collision probability, which can then be used for rounding. By that, we mean that our distribution essentially consists of only one assignment (upto shift-symmetry) and its perturbations.

Non small-set expanders

Consider a graph which is not a small set expander, say that G is the Johnson graph $J(n, \ell, t = \ell/2)$. In that case the above reasoning no longer works as F_s may indeed be all small sets. However, as explained earlier, using global hypercontractivity we can infer that one of the sets F_s must possess a certain structure – it must have large density inside one of the canonical non-expanding sets. In the case of the Johnson graph specifically, these canonical sets take the following form:

$$H_R = \left\{ A \in \binom{[n]}{\ell} \mid A \supseteq R \right\},$$

for $R \subset [n]$ and $|R| = r = O(1)$. In fact, global hypercontractivity gives the following stronger structural property: the set $H = \bigcup_{R \in \mathcal{R}} H_R$ where \mathcal{R} consists of all R 's inside which some part F_s is dense, has a constant measure. Doing simple accounting, it follows that $|\mathcal{R}| \geq \Omega(n^r/\ell^r)$ and as there are at most $\binom{n}{r}$ different canonical sets it follows that $|\mathcal{R}|$ contains an $\Omega(1/\ell^r)$ fraction of these sets.

For each choice of X and X' though we may have a different collection of dense subcubes \mathcal{R} . But since \mathcal{R} contains an $\Omega(1/\ell^r)$ fraction of all the subcubes, we get that there must be at least one subcube H_R that is dense with probability $\Omega(1/\ell^r)$ over $X, X' \sim \mathcal{D}$. Let H_R be such a subcube and $E_R(X, X')$ be the event that H_R is dense. Ideally, at this point one would like to condition on E_R so that one of the parts inside the shift partition F_s becomes large inside H_R , and then hope that as was the case for small-set expanders, we can satisfy many of the edges inside $F_s \cap H_R$.

Unfortunately, this hope does not materialize – after conditioning on E_R even though the shift-partition is large, the rounding strategy above may break. Indeed, for the rounding procedure we wanted the values of $X(u)$ and $X'(u)$ for $X, X' \sim \mathcal{D}$ to be independent for every vertex u . However, after conditioning the joint distribution $\mathcal{D} \times \mathcal{D} \mid E_R$ over (X, X') might have correlations between X and X' . In particular this distribution could even be supported on pairs (X, X') that are always equal to each other, in which case the shift-partition is large because of trivial reasons and therefore its large size doesn't imply anything about the collision probability/entropy of \mathcal{D} .

Hence in [5] the authors don't manage to do this conditioning, and instead settle for satisfying an $\Omega\left(\frac{1}{\ell^{2r}}\right)$ -fraction of the constraints on H_R . After that they iterate this algorithm many times to satisfy an $\Omega\left(\frac{1}{\ell^{2r}}\right)$ -fraction of the constraints of the whole graph.

2.2 Our Approach: Conditioning on the Event E via (Eliminating) Global Correlations

Our main contribution to the above framework is to show that by adding an additional preprocessing step, we can ensure that even after conditioning on the event E_R above, the assignments X and X' will remain highly independent. In particular, the fact that some part in the shift partition becomes large must happen – just like in the case of small-set expanders – due to the fact that our distribution has high collision probability.

As the event $E = E_R(X, X')$ has probability at least $\Omega(\frac{1}{\ell^r})$, if we are sufficiently high up in the SoS hierarchy ($\Theta(\ell^r)$ levels will do, for an overall running time of $n^{\Theta(\ell^r)}$), we do have access to the conditional pseudoexpectation

$$\tilde{\mathbb{E}}[Y \mid E] = \frac{\tilde{\mathbb{E}}[Y 1_E]}{\tilde{\mathbb{E}}[1_E]}.$$

This means that we can sample labels of vertices conditioned on the event E . To make this useful though, we must change the rounding procedure. To get some intuition consider the extreme case in which after conditioning on $E(X, X')$ there are huge correlations between X and X' that remain in our distribution.

Namely, suppose that after conditioning on E it holds that $X(u) = X'(u)$ for almost all vertices u . In that case, if we sampled X, X' from $\mathcal{D} \times \mathcal{D}$ (not conditioned on E), we would get that with probability at least $\Pr[E] \geq \Omega(\frac{1}{\ell^r})$ the event E holds, in which case X and X' agree on almost all vertices. This means that if \mathcal{D} was an actual distribution the assignments have a large global correlation: fix $X' = X_0$ for X_0 that satisfies $\Pr_{\mathcal{D}}[E(X, X_0) = 1] \geq \Pr_{\mathcal{D} \times \mathcal{D}}[E]$. Once E holds, we have that $X(u) - X(v) = X_0(u) - X_0(v)$ for almost all pairs of vertices, hence the values of the assignment X to the vertices u and v is correlated across \mathcal{D} . Therefore, a natural idea is to avoid this issue by transforming \mathcal{D} to another distribution lacking global correlations, in the sense that the assignments to a typical pair of vertices u and v are almost independent.

For this purpose we use an idea from [41], which adapted to our setting says that for any $\tau > 0$ there is $d = d(\tau, |\Sigma|)$ such that conditioning $\tilde{\mathbb{E}}$ on the values of d randomly chosen vertices ensures that the global correlation is at most τ . That is, the values of $X(u)$ and $X(v)$ for two typical vertices u and v are at most τ -correlated, and the same holds for X' . In the full version of the paper we then show that if we start with such a pseudodistribution that lacks global correlations, then one can condition on the event E and retain near independence between the assignments X and X' , at least on most vertices. To be more precise, we show that for $Y_{u,v} = (X(u), X(v))$ and $Y'_{u,v} = (X'(u), X'(v))$, the statistical distance between $Y_{u,v}, Y'_{u,v} \mid E$ and $Y_{u,v}, Y'_{u,v}$ is small for almost all pairs of vertices u, v .⁶

Using this idea we are able to get an $\Omega(1)$ -valued solution on some basic set H_R . To summarize, we first preprocess the pseudodistribution to eliminate global correlations. We can then find an event $E(X, X')$, corresponding to the fact that some part F_s in the shift partition has become dense in some basic set H_R . Furthermore, conditioning on E most pairs $(X(u), X(v)), (X'(u), X'(v))$ remain almost-independent. Then running a simple rounding procedure on H_R (as in [5]), we are able to satisfy a good fraction of the edges inside H_R . H_R might be a $o(1)$ fraction of the graph though, therefore like [5] we repeat this procedure multiple times to get an $\Omega(1)$ -valued solution for the whole graph. This gives an efficient algorithm for affine UG over the Johnson graphs as in Theorem 5.

To prove Theorem 6 (namely, the regime where c is not close to 1) more work is needed. Indeed, in the case that c is close to 1 we are able to conclude that essentially all edges stay within some part F_s of the shift partition. Thus, as long as our sets H_R cover a constant fraction of the edges that stay within some F_s , they are automatically guaranteed to cover a constant fraction of the edges that are satisfied by both X and X' , and these are the edges our rounding procedure manages to satisfy. If c is just bounded away from 0 we can no longer make such an argument, and it is no longer even clear that the sets H_R cover some edges that we have a hope of satisfying.

⁶ To make our rounding succeed we need to use a more complicated version of $Y_{u,v}$ (see the full version of the paper).

2.3 Getting Small Completeness: Capturing all of the Non-expanding Edges

To design our algorithm for the case when the completeness c is just guaranteed to be bounded away from 0 we must first argue that in the shift partition, we are able to capture almost all of the edges that stay within a part F_s using the basic sets H_R (so as to ensure we are including the edges that X and X' both satisfy).

Towards this end we require a more refined corollary of global hypercontractivity, asserting that if we have a small set of vertices F in the Johnson graph that has edge expansion at most $1 - \eta$, then we can find a collection \mathcal{R} of basic sets such that:

1. **Bounded and dense:** each $R \in \mathcal{R}$ has size $|R| = O(1)$ and F is dense inside each H_R . That is, $\delta(F \cap H_R) \geq \Omega_\eta(\delta(H_R))$ for each $R \in \mathcal{R}$.
2. **Maximally dense:** For all $R \in \mathcal{R}$ and all $R' \subsetneq R$, F is not very dense in $H_{R'}$.
3. **Capture almost all non-expanding edges:** Almost all the edges that stay inside F also stay inside H_R for some $R \in \mathcal{R}$.

Indeed, we show that a global hypercontractive inequality such as the one in [30] can be used to prove such a result (in a black-box manner).

Using this result, we are able to argue that the edges that stay inside the subcubes H_R for $R \in \mathcal{R}$ cover most of the edges that stay within the same part in the shift partition. There are several subtleties here that one has to deal with, for example, “regularity issues” such as, how many different R ’s cover a given edge. The goal of the second item above is to handle such concerns, and it roughly says that no vertex nor edge gets over-counted by a lot. After that, we are able to condition on an event E , where as before E indicates that some part F_s becomes dense inside some basic set H_R , so that the resulting distribution has a large shift-partition inside H_R . At this point, we are (morally) back to the problem of rounding the SoS solution on a set with a large shift-partition, except that now our solution has value $c' > 0$ (as opposed to close to 1). We remark that again, we use the “elimination of global correlations” idea presented earlier to retain near independence after conditioning. With more care, we use a similar analysis to the one presented for completeness close to 1 to finish the proof when c is arbitrarily small.

References

- 1 Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *J. ACM*, 62(5):42:1–42:25, 2015. doi:10.1145/2775105.
- 2 Sanjeev Arora, Russell Impagliazzo, William Matthews, and David Steurer. Improved algorithms for unique games via divide and conquer. *Electron. Colloquium Comput. Complex.*, 17:41, 2010. URL: <http://eccc.hpi-web.de/report/2010/041>, arXiv:TR10-041.
- 3 Sanjeev Arora, Subhash Khot, Alexandra Kolla, David Steurer, Madhur Tulsiani, and Nish-eeth K. Vishnoi. Unique games on expanding constraint graphs are easy: extended abstract. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 21–28, 2008. doi:10.1145/1374376.1374380.
- 4 Per Austrin. Balanced max 2-sat might not be the hardest. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 189–197. ACM, 2007. doi:10.1145/1250790.1250818.
- 5 Mitali Bafna, Boaz Barak, Pravesh K. Kothari, Tselil Schramm, and David Steurer. Playing unique games on certified small-set expanders. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1629–1642. ACM, 2021.

- 6 Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. High dimensional expanders: Eigenstripping, pseudorandomness, and unique games. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 – 12, 2022*, pages 1069–1128, 2022.
- 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.
- 8 Mitali Bafna, Noam Lifshitz, and Dor Minzer. Constant degree direct product testers with small soundness. *arXiv preprint*, 2024. [arXiv:2402.00850](https://arxiv.org/abs/2402.00850).
- 9 Mitali Bafna and Dor Minzer. Characterizing direct product testing via coboundary expansion. *Electron. Colloquium Comput. Complex.*, TR23-120, 2023. [arXiv:TR23-120](https://arxiv.org/abs/2304.07284).
- 10 Mitali Bafna and Dor Minzer. Solving unique games over globally hypercontractive graphs. *CoRR*, abs/2304.07284, 2023. [doi:10.48550/arXiv.2304.07284](https://doi.org/10.48550/arXiv.2304.07284).
- 11 Ainesh Bakshi, Ilias Diakonikolas, Samuel B. Hopkins, Daniel Kane, Sushrut Karmalkar, and Pravesh K. Kothari. Outlier-robust clustering of gaussians and other non-spherical mixtures. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 149–159. IEEE, 2020.
- 12 Ainesh Bakshi, Ilias Diakonikolas, He Jia, Daniel M. Kane, Pravesh K. Kothari, and Santosh S. Vempala. Robustly learning mixtures of k arbitrary gaussians. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 – 24, 2022*, pages 1234–1247. ACM, 2022.
- 13 Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 – 22, 2012*, pages 307–326, 2012. [doi:10.1145/2213977.2214006](https://doi.org/10.1145/2213977.2214006).
- 14 Boaz Barak, Jonathan A. Kelner, and David Steurer. Rounding sum-of-squares relaxations. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 31–40, 2014. [doi:10.1145/2591796.2591886](https://doi.org/10.1145/2591796.2591886).
- 15 Boaz Barak, Pravesh K. Kothari, and David Steurer. Quantum entanglement, sum of squares, and the log rank conjecture. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 975–988, 2017. [doi:10.1145/3055399.3055488](https://doi.org/10.1145/3055399.3055488).
- 16 Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 472–481, 2011. [doi:10.1109/FOCS.2011.95](https://doi.org/10.1109/FOCS.2011.95).
- 17 Rares-Darius Buhai, Pravesh K. Kothari, and David Steurer. Algorithms approaching the threshold for semi-random planted clique. *CoRR*, abs/2212.05619, 2022. [doi:10.48550/arXiv.2212.05619](https://doi.org/10.48550/arXiv.2212.05619).
- 18 Yotam Dikstein and Irit Dinur. Agreement theorems for high dimensional expanders in the small soundness regime: the role of covers. *Electron. Colloquium Comput. Complex.*, TR23-119, 2023. [arXiv:TR23-119](https://arxiv.org/abs/2304.07284).
- 19 Yotam Dikstein, Irit Dinur, and Alexander Lubotzky. Low acceptance agreement tests via bounded-degree symplectic hdxs, 2024. [arXiv:2402.01078](https://arxiv.org/abs/2402.01078).
- 20 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 974–985. IEEE Computer Society, 2017. [doi:10.1109/FOCS.2017.94](https://doi.org/10.1109/FOCS.2017.94).
- 21 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. 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 940–951. ACM, 2018. [doi:10.1145/3188745.3188806](https://doi.org/10.1145/3188745.3188806).

- 22 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.
- 23 Yuval Filmus, Guy Kindler, Noam Lifshitz, and Dor Minzer. Hypercontractivity on the symmetric group. *arXiv preprint*, 2020. [arXiv:2009.05503](https://arxiv.org/abs/2009.05503).
- 24 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.
- 25 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.
- 26 Peter Keevash, Noam Lifshitz, Eoin Long, and Dor Minzer. Global hypercontractivity and its applications. *arXiv preprint*, 2021. [arXiv:2103.04604](https://arxiv.org/abs/2103.04604).
- 27 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775, 2002. [doi:10.1145/509907.510017](https://doi.org/10.1145/509907.510017).
- 28 Subhash Khot. On the unique games conjecture (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 99–121. IEEE Computer Society, 2010. [doi:10.1109/CCC.2010.19](https://doi.org/10.1109/CCC.2010.19).
- 29 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O’Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable csp? *SIAM J. Comput.*, 37(1):319–357, 2007. [doi:10.1137/S0097539705447372](https://doi.org/10.1137/S0097539705447372).
- 30 Subhash Khot, Dor Minzer, Dana Moshkovitz, and Muli Safra. Small set expansion in the johnson graph. *Electron. Colloquium Comput. Complex.*, TR18-078, 2018. [arXiv:TR18-078](https://arxiv.org/abs/1801.07801).
- 31 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, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 576–589, 2017. [doi:10.1145/3055399.3055432](https://doi.org/10.1145/3055399.3055432).
- 32 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. *Annals of Mathematics*, 198(1):1–92, 2023.
- 33 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. [doi:10.1016/j.jcss.2007.06.019](https://doi.org/10.1016/j.jcss.2007.06.019).
- 34 Guy Kindler, Assaf Naor, and Gideon Schechtman. The UGC hardness threshold of the L_p grothendieck problem. *Math. Oper. Res.*, 35(2):267–283, 2010. [doi:10.1287/moor.1090.0425](https://doi.org/10.1287/moor.1090.0425).
- 35 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. Optim.*, 11(3):796–817, 2000/01. [doi:10.1137/S1052623400366802](https://doi.org/10.1137/S1052623400366802).
- 36 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of ramanujan complexes of type ad. *European Journal of Combinatorics*, 26(6):965–993, 2005.
- 37 Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–446. IEEE, 2016.
- 38 Konstantin Makarychev and Yury Makarychev. How to play unique games on expanders. In Klaus Jansen and Roberto Solis-Oba, editors, *Approximation and Online Algorithms – 8th International Workshop, WAOA 2010, Liverpool, UK, September 9-10, 2010. Revised Papers*, volume 6534 of *Lecture Notes in Computer Science*, pages 190–200. Springer, 2010. [doi:10.1007/978-3-642-18318-8_17](https://doi.org/10.1007/978-3-642-18318-8_17).
- 39 Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.
- 40 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 245–254, 2008. [doi:10.1145/1374376.1374414](https://doi.org/10.1145/1374376.1374414).

- 41 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*, pages 373–387. SIAM, 2012.
- 42 Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. doi: 10.1137/S0097539795280895.
- 43 Luca Trevisan. On khot’s unique games conjecture. *Bulletin (New Series) of the American Mathematical Society*, 49(1), 2012.

Derandomizing Logspace with a Small Shared Hard Drive

Edward Pyne   

MIT, Cambridge, MA, USA

Abstract

We obtain new catalytic algorithms for space-bounded derandomization. In the catalytic computation model introduced by (Buhrman, Cleve, Koucký, Loff, and Speelman STOC 2013), we are given a small worktape, and a larger catalytic tape that has an arbitrary initial configuration. We may edit this tape, but it must be exactly restored to its initial configuration at the completion of the computation. We prove that

$$\mathbf{BSPACE}[S] \subseteq \mathbf{CSPACE}[S, S^2]$$

where $\mathbf{BSPACE}[S]$ corresponds to randomized space S computation, and $\mathbf{CSPACE}[S, C]$ corresponds to catalytic algorithms that use $O(S)$ bits of workspace and $O(C)$ bits of catalytic space. Previously, only $\mathbf{BSPACE}[S] \subseteq \mathbf{CSPACE}[S, 2^{O(S)}]$ was known. In fact, we prove a general tradeoff, that for every $\alpha \in [1, 1.5]$,

$$\mathbf{BSPACE}[S] \subseteq \mathbf{CSPACE}[S^\alpha, S^{3-\alpha}].$$

We do not use the algebraic techniques of prior work on catalytic computation. Instead, we develop an algorithm that branches based on if the catalytic tape is conditionally random, and instantiate this primitive in a recursive framework. Our result gives an alternate proof of the best known time-space tradeoff for $\mathbf{BSPACE}[S]$, due to (Cai, Chakaravarthy, and van Melkebeek, Theory Comput. Sys. 2006). As a final application, we extend our results to solve search problems in $\mathbf{CSPACE}[S, S^2]$. As far as we are aware, this constitutes the first study of search problems in the catalytic computing model.

2012 ACM Subject Classification Theory of computation \rightarrow Pseudorandomness and derandomization

Keywords and phrases Catalytic computation, space-bounded computation, derandomization

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.4

Funding Supported by a Jane Street Graduate Research Fellowship.

Acknowledgements I am grateful to William Hoza for bringing the reference [5] to my attention. I thank Dean Doron, Ian Mertz, Roei Tell, Ryan Williams, and anonymous reviewers for helpful discussions and comments on the manuscript.

1 Introduction

In the catalytic logspace (**CL**) model, introduced by Buhrman, Cleve, Koucký, Loff, and Speelman [3], there is a machine M with $O(\log n)$ bits of standard working memory, and n^c bits of catalytic memory. This catalytic memory has an arbitrary initial configuration (perhaps data on a shared hard drive), and must be returned to exactly this configuration at the end of the computation. Remarkably, [3] showed that **CL** is likely to be strictly more powerful than **L**. In particular, it contains logspace-uniform \mathbf{TC}^1 and thus **NL**. Motivated by this striking result, there have been several further works exploring the power of catalytic computation [4, 12, 10, 8, 2, 9].

We parameterize catalytic computation by time, space, and catalytic space (similar notions have been considered before, e.g. [2]).



© Edward Pyne;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 4; pp. 4:1–4:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



► **Definition 1.** Let $\mathbf{CTISP}[T(n), S(n), C(n)]$ be the set of languages recognized by catalytic machines that use $O(S(n))$ workspace and $O(C(n))$ catalytic space on inputs of size n , and run in time $\text{poly}(T(n))$ in the worst case.

Note that the worst-case runtime must hold over every catalytic tape, as well as every input.

Prior work has studied the ability of catalytic space to substitute for *randomness*, in particular in the setting of derandomizing space-bounded computation. Let \mathbf{BPL} be the set of languages recognized by randomized machines that run in space $O(\log n)$ on inputs of size n , and make two-sided error. The result of [3] implies that

$$\mathbf{BPL} \subseteq \mathbf{CTISP}[n, \log n, n^c]$$

for some constant c . In fact, we are aware of two other proofs of this fact. An unpublished result (see the recent survey of Mertz [15] for a sketch) proves it by treating the catalytic tape as a set of random walks, and the third follows from recent work on certified derandomization for \mathbf{BPL} [20, 11].

As our main result, we improve the amount of catalytic space needed to simulate \mathbf{BPL} by a superpolynomial amount.

► **Theorem 2.**

$$\mathbf{BPL} \subseteq \mathbf{CTISP}[n, \log n, \log^2 n].$$

Our simulation of \mathbf{BPL} is as time- and space- efficient as the frontier result of Nisan [18], which proves that $\mathbf{BPL} \subseteq \mathbf{TISP}[n, \log^2 n]$, and moreover almost all the space used is catalytic. Next, we incorporate this algorithm into a recursive framework to derive a (time-efficient) tradeoff between the catalytic- and non-catalytic space consumption.

► **Theorem 3.** For every $\alpha \in [1, 1.5]$,

$$\mathbf{BPL} \subseteq \mathbf{CTISP}\left[2^{\log^\alpha(n)}, \log^\alpha(n), \log^{3-\alpha}(n)\right].$$

This result immediately gives a new proof of the best known result on time-space tradeoffs for \mathbf{BPL} due to Cai, Chakaravathy, and van Melkebeek [5]. They prove \mathbf{BPL} is contained in $\mathbf{TISP}[2^{\log^\alpha n}, \log^{3-\alpha} n]$ for every $\alpha \in [1, 1.5]$. In fact, our result shows that for every $\alpha < 1.5$, we can achieve a simulation with equivalent time and total space, but where the majority of the space used can be made catalytic.

Interestingly, while previous work on algorithms for catalytic computation [3, 8] primarily used algebraic techniques involving reversible computation over a ring, our results take a completely different approach based on conditional compressibility. We also develop a new approach for efficient composition of catalytic algorithms, building on the well-known composition of space-bounded algorithms. We hope that our techniques will have broader applications, both inside and beyond the model of catalytic computation. As a final application, we show that our ideas can be used to give nontrivial catalytic search algorithms.

1.1 Proof Overview for Theorem 2

A canonical (promise)- \mathbf{BPL} complete problem is that of estimating transition probabilities in read-once branching programs:

► **Definition 4.** A *read-once branching program (ROBP)* \bar{B} of width w and length n and alphabet $\{0, 1\}^t$ is defined by a function $B : [w] \times \{0, 1\}^t \rightarrow [w]$.¹ For $x \in (\{0, 1\}^t)^n$, define

$$\bar{B}[i, x] = B[B[\dots B[B[i, x_1], x_2] \dots], x_{n-1}], x_n].$$

It is well known that to derandomize **BPL**, it suffices to estimate $\Pr_{x \leftarrow U_n}[\bar{B}[1, x] = 1]$ up to error $1/3$ for an ROBP \bar{B} of length n , width n and alphabet $\{0, 1\}$.

The Result of Nisan

We now recall the result of [18], which itself begins with the PRG of [16]. In this PRG, we draw $\ell = \log n$ hash functions h_1, \dots, h_ℓ from a pairwise independent hash family on $t = O(\log nw)$ bits. We recursively define the PRG as follows. Let $\text{NIS}_0(x) = x$ for $x \in \{0, 1\}^t$, and let $\text{NIS}_{i+1}(x) = (\text{NIS}_i(x), \text{NIS}_i(h_{i+1}(x)))$. To analyze this PRG, fix a branching program $\bar{B} : (\{0, 1\}^t)^n \rightarrow \{0, 1\}$ of width w , with transition function B . Viewing this construction from the bottom up, the first hash function h_1 is good if for every every $a, b \in [w]$,

$$\Pr_{x, x' \leftarrow U_t}[B[B[a, x], x'] = b] \approx \Pr_{x \leftarrow U_t}[B[B[a, x], h_1(x)] = b],$$

i.e. the distribution $(x, h_1(x))$ is indistinguishable from the distribution (x, x') by the composition of B with itself. Since B can only pass $\log w$ bits of information from the first to the second half, this occurs with probability $1 - w^{-c}$ over $h_1 \leftarrow \mathcal{H}$ (assuming $t = O(\log w)$ is sufficiently large). The ultimate PRG is analyzed recursively using ℓ applications of essentially the same idea. Concretely, at the second level of the construction, we now want a hash function h_2 that fools the length $n/2$ program with transition function $B'[a, x] = B[B[a, x], h_1(x)]$.

While the Nisan PRG randomly selects ℓ hash functions at once, the insight of [18] was that, given a specific program \bar{B} that we want to fool, we can *search* for good hash functions level by level. At level i , we find a hash function h_i that fools the relevant transition function. As this test is easy to implement in time $2^{O(t)}$ for a fixed h and there are $2^{O(t)}$ such h to test, we can find such a good hash function in time $2^{O(t)} = \text{poly}(n)$ per level, giving a polynomial runtime overall.

An Algorithm From Conditional Compression

We transform this algorithm into a *catalytic* algorithm as follows. Suppose we have a branching program \bar{B} of width w and length $n = 2^\ell$, and a catalytic tape \mathbf{w} , with an arbitrary initial configuration. We interpret \mathbf{w} as holding 2ℓ hash functions $h_1, \dots, h_{2\ell}$, each over $t = O(\log nw)$ bits (and note that each function can have description size exactly $2t$, as there exists a pairwise independent hash family on t bits of size 2^{2t}). Let $V \in \{0, 1, *\}^{2\ell}$ and initialize $V = *^{2\ell}$ to indicate the status of each block. We then iterate through this list. Letting the i th hash function be \tilde{h} and the previous good hash functions be \vec{h}_p , we check if \tilde{h} is a good hash function, using the test as before.

- If \tilde{h} is good, we set $V_i = 1$, indicating \tilde{h} is part of the list of good hashes.
- If \tilde{h} is not good, it must lie in the set $\mathbf{BAD}(\vec{h}_p)$ of hash functions that fail to fool the current transition function. But as almost all h are good, the index of \tilde{h} in $\mathbf{BAD}(\vec{h}_p)$ is a concise description of \tilde{h} ! We can then replace \tilde{h} with this index, and free up $\Omega(\log nw)$ bits on this block of the tape. Finally, set $V_i = 0$ to indicate we have compressed this block.

¹ The standard definition of ROBPs permits the transition function to differ between the layers. However, as we will always be dealing with programs where $w \geq n$, and we are insensitive to polynomial losses in the width, we can assume all transition functions are the same for clarity.

At the end of this phase, we have either found ℓ good hash functions, or have freed up $\ell \cdot \Omega(\log nw)$ bits on the tape. In the latter case, we can simply search for a good set of hash functions (on slightly fewer bits), exactly as in the algorithm of [18], and store these in the free space of the compressed blocks. Thus, in both cases we obtain a sequence of hash functions that together constitute a good PRG for \overline{B} , and hence can construct a generator NIS that does a good job estimating walk probabilities on \overline{B} . The final step of estimating these walks can be performed in space $O(t + \log nw) = O(\log nw)$ with read-only access to the tape \mathbf{w} .

Finally, to return the tape to its original configuration, we work backwards over the compressed blocks, i.e. indices i where $V_i = 0$. For each block, we determine the preceding good hash functions \vec{h}_p , read the index of the original hash (i.e. tape configuration) in $\mathbf{BAD}(\vec{h}_p)$, then find the hash with this index by enumeration and write it to the tape.

1.2 Proof Overview for Theorem 3

To obtain a smooth tradeoff between the catalytic and non-catalytic space, our next idea is to unify this with efficient *composition* of catalytic algorithms:

Composition of Catalytic Algorithms

Recall that in the conventional composition of space-bounded algorithms, we can compute the composition of two algorithms running in space $S(n)$ in space $c \cdot S(n)$, for some constant $c > 1$. Our key observation is that for catalytic algorithms, we can obtain composition with *no* increase in the length of the catalytic tape:

► **Theorem 5** (Composition of Catalytic Space-Bounded Algorithms). *Given two catalytic algorithms $\mathcal{M}_1, \mathcal{M}_2$ computing f_1, f_2 respectively, each using space $S(n) \geq \log n$, catalytic space $C(n)$, and time $T(n)$, there is a catalytic algorithm \mathcal{M}' using time $\text{poly}(T(n))$, space $O(S(n))$, and catalytic space $C(n)$ that computes $f_2 \circ f_1$.*

The proof of this result modifies the standard composition of space-bounded algorithms. To compute $\mathcal{M}_2(\mathcal{M}_1(x))$, we begin to simulate $\mathcal{M}_2^{\mathbf{w}}(f_1(x))$ (where the superscript notation denotes running the machine with catalytic tape \mathbf{w}). Whenever \mathcal{M}_2 reads a bit of the input, we simulate $\mathcal{M}_1^{\mathbf{w}'}(x)$ to obtain the relevant bit of $f_1(x)$, where \mathbf{w}' is the current configuration of the catalytic tape of \mathcal{M}_2 . Since \mathcal{M}_1 is guaranteed to produce the correct answer for every starting tape, we have that $\mathcal{M}_1^{\mathbf{w}'}(x) = f_1(x)$. Moreover, as \mathcal{M}_1 is catalytic, it resets the tape to \mathbf{w}' before returning, so \mathcal{M}_2 does not notice the call has occurred, and can continue its computation.

We remark that we are not able to apply this theorem as-is due to issues with (essentially) f_1 being a relation with multiple valid outputs, so the actual statement we prove is more involved. In particular, we must deal with safety reverting the catalytic tape if an intermediate call to \mathcal{M}_1 fails.

Derandomization via Repeated Powering

Going from this to Theorem 3 requires a further ingredient, which is given by a variant of the Saks-Zhou recursive powering scheme. Saks-Zhou [21] divides computing the n th power of an $n \times n$ stochastic matrix M (a **prBPL** complete problem) into r_2 iterations of computing the 2^{r_1} th power, for any $r_1 r_2 = \log n$. For convenience, let $M_0 = M$ and $M_i = M^{2^{r_1 \cdot i}}$ for $i \in [r_2]$. In the original algorithm, all levels share a single set of hash functions $\vec{h} = (h_1, \dots, h_{r_1})$, each on $O(\log n)$ bits. A random set of hash functions will do a good job computing M_i

from M_{i-1} for every i , and so we can reuse this fixed set of hash functions at every level.² Unfortunately, such an argument is incompatible with searching for good hash functions one by one. Since we use every hash function to produce an approximation to M_1 , if we later discover a hash function is bad at powering M_i for $i \geq 1$, seemingly we must destroy all partial progress and try a new set of hash functions. Thus, the Saks-Zhou algorithm must enumerate over $\vec{h} = (h_1, \dots, h_{r_1})$ all at once, incurring a runtime of $2^{\Omega(r_1 \log n)}$. As the algorithm incurs a runtime of $2^{\Omega(r_2 \log n)}$ merely from the recursive composition of space bounded algorithms, the total runtime is at least $2^{\Omega(\max\{r_1, r_2\} \log n)} = 2^{\Omega(\log^{3/2} n)}$ for any setting of parameters r_1 and r_2 . We note that the work of [5] also avoids this issue, and we explain their differing approach in more detail in Section 1.2.

Composing Conditional Compression Algorithms

Our catalytic algorithm allows for a more efficient approach. We follow the same recursive powering scheme as Saks-Zhou, but at each level use the algorithm of Theorem 2 that treats \mathbf{w} as a list of $2r_1$ candidate hash functions.³ Whenever we request an entry of a smaller power, we call the next level algorithm. If that level sees that the hash functions currently on the tape are good, it uses them to compute the requested entry. If not, it temporarily compresses the tape, finds good hash functions in time $\text{poly}(n)$, uses them to compute the requested entry, then resets the tape to exactly the same configuration the calling algorithm was expecting before returning. Thus, every level can either use the tape as-is, if it is suitable, or quickly compute a better set of hash functions on the fly and revert before returning control. This eliminates the $2^{r_1 \log n}$ term in the runtime. Moreover, the $O(r_1 \log n)$ bits used to store the hash functions can be treated as catalytic space, resulting in an algorithm that uses only $O(r_2 \log n)$ bits of workspace.

Finally, for every $\alpha \in [1, 1.5]$, we can choose $r_1 = \log^{2-\alpha}(n)$ and $r_2 = \log^{\alpha-1}(n)$ and obtain a algorithm that uses $O(r_1 \log n) = O(\log^{3-\alpha} n)$ catalytic space, $O(r_2 \log n) = O(\log^\alpha n)$ workspace, and runs in time $\text{poly}(n^{r_2}) = 2^{O(\log^\alpha n)}$, as claimed.

Such an approach runs into a subtle technical issue. Since the algorithm at level i may be called many times with different starting catalytic tapes, we must ensure that the algorithm returns the *same* approximate power each time, as otherwise the composition would not be well defined. To fix this, we first define a notion of catalytic algorithms that are allowed to return \perp for some initial catalytic tapes, in addition to a fixed output that is independent of the catalytic tape. We then show how these algorithms can be composed, while still maintaining the ability to revert the tape to the original configuration in the worst case. Finally, we adopt the strategy of Saks and Zhou [21], and randomly perturb (or “shift”) the matrices at each level. In our case, if a level of the algorithm determines that a shift is bad (i.e. could produce ambiguous behavior) it aborts and returns \perp . We show with high probability over the shifts, this will never occur (i.e. we will not return \perp) no matter the tape, and so we can compose the algorithm with itself and find the desired output.

Showing that we can successfully avoid permanent damage to the tape in the case that the shifts are bad requires further work. In particular, we ensure that our catalytic algorithms can be *reverted* from any point, where our notion of reversibility requires that we do not introduce

² There are additional complications from reusing the hash functions, but they are not the primary reason for the high time complexity.

³ We give a “non-black-box” explanation of the final algorithm here as it illustrates the actual idea, but our proof uses a black-box statement regarding composition of catalytic algorithms.

any new configurations of the catalytic tape. We show that we can achieve this notion without a substantial time cost, and moreover it is compatible with recursive composition. Using this tool, we are able to return to the original tape configuration of a subroutine ever returns \perp .

Comparison With [5]

We briefly overview the techniques of [5], which achieves the best known time-space tradeoff for **BPL**, but in which all $O(\log^{3-\alpha} n)$ bits of space must be standard workspace. They likewise give a version of the Saks-Zhou result that does not incur the n^{r^2} factor in runtime, which we now explain. Their result follows the following recursive framework. We start with a set of hash functions $\vec{h} = (h_1, \dots, h_\ell)$ that produce a good approximation of M_i (which we denote \widetilde{M}_i) from M_{i-1} for every $i \leq r$, but does not necessarily produce a good approximation of M_{r+1} from M_r . We then search for a new set of hash functions $\vec{h}' = (h'_1, \dots, h'_\ell)$ with the following two properties. First, \vec{h}' is good at approximately powering M_i for every $i \leq r$ (in particular, it produces a good approximation \widetilde{M}'_{r+1}). Second, after applying the random shift and round operation to the approximations $\widetilde{M}_i, \widetilde{M}'_i$ for $i \leq r$ produced by the old and new sets of hash functions, we obtain *the same* matrices. After doing so, we replace \vec{h} with \vec{h}' and increment r . The latter requirement allows us to make progress, as we can gradually find sets of hash functions that are good for greater powers, without destroying progress by altering the “results” of prior computation. However, this approach does not give a catalytic algorithm (in particular, it does not exploit the fact that bad hash functions are compressible).

1.3 Search Problems in Catalytic Space

Finally, we show how our compression-based techniques can be extended to solve search problems in catalytic space. As far as we are aware, we are the first to study catalytic search algorithms. Previous work of Sivakumar [22] showed that many search problems, such as producing a Johnson-Lindenstrauss sketch of a collection of vectors, can be reduced in logspace to solving the following problem, which we call “mutual ROBP hitting”.

► **Definition 6** (Mutual ROBP Hitting). *Given a list of branching programs $(\overline{B}^1, \dots, \overline{B}^n)$ each of length and width n , such that $\mathbb{E}[\overline{B}^j(U_n)] \geq 1 - 1/n^2$ for every j , produce a fixed $x \in \{0, 1\}^n$ such that*

$$\bigwedge_{j \in [n]} \overline{B}^j(x) = 1.$$

We remark that this problem is (as of now) possibly harder than the task of producing x that satisfies a single ROBP, as it is not known to lie in **BPL**. Despite this, we show that we can solve this problem in the same asymptotic bound as Theorem 2:

► **Theorem 7.** *There is a **CTISP** $[n, \log n, \log^2 n]$ algorithm that solves the mutual ROBP hitting problem.*

Previously, this problem was known to be solveable in **TISP** $[n, \log^2 n]$, so we again show that almost all of the required space can be made catalytic.

We can immediately apply the reduction of [22] to obtain search algorithms for well-studied problems. As one application, we obtain a catalytic algorithm which produces a Johnson-Lindenstrauss transform: a low-dimensional embedding of a collection of vectors that approximately preserves their ℓ_2 distance. The problem of deterministically producing such an embedding has been extensively studied [22, 1, 13, 14].

► **Corollary 8.** *There is a CTISP $[n, \log n, \log^2 n]$ algorithm that, given $\varepsilon > 0$ and a collection of vectors $v_1, \dots, v_n \in \mathbb{R}^n$, outputs vectors⁴ $\tilde{v}_1, \dots, \tilde{v}_n \in \mathbb{R}^{O(\log(n)/\varepsilon^2)}$ such that for every $i, j \in [n]$, we have*

$$(1 - \varepsilon)\|v_i - v_j\|_2^2 \leq \|\tilde{v}_i - \tilde{v}_j\|_2^2 \leq (1 + \varepsilon)\|v_i - v_j\|_2^2.$$

We prove Theorem 7 by extending our compress-or-random approach to producing a set of hash functions that is good for a *polynomial number* of ROBPs at once. In more detail, suppose the algorithm of Theorem 2 is now given a list of branching programs (B^1, \dots, B^n) . The algorithm is structured as before, but now for each new hash function \tilde{h} , we test if \tilde{h} is good for all of the input ROBPs. If yes, we again add it to our good sublist. Otherwise, let B^j be the first program that \tilde{h} was not good for, and let \mathbf{BAD}^j be the set of bad hash functions for this program, given the current prefix (and note that the prefix is by construction good for all programs). We now compress \tilde{h} by recording j (which we require for the decompression algorithm to recover the hash function) together with the index of \tilde{h} in \mathbf{BAD}^j . Again as before, once we have processed all blocks, either we have found a set of hash functions that are good for all ROBPs, or we have freed up $\Omega(\log^2 n)$ space. In that case, we again search for a set of hash functions that is good for every program simultaneously. This only incurs a mild constant factor loss in the length of each hash function, so our algorithm has the same asymptotic performance.

1.4 Roadmap

In Section 2, we formally define the catalytic computation model, and prove Theorem 5 and Theorem 7. In Section 3, we prove Theorem 2, and in Section 4, we prove Theorem 3. In Appendix A we provide proofs of some cited lemmas.

2 Catalytic Machines and Composition

We first formally define a catalytic Turing machine.

► **Definition 9** (Catalytic Turing Machine [3]). *A Turing machine \mathcal{M} is a **catalytic machine** using time $T(n)$, workspace $S(n)$, and catalytic space $C(n)$ if it has a work tape, a read-only input tape, a write-only output tape, and a catalytic tape \mathbf{w} . We require that for every input x with $|x| = n$ and every \mathbf{w} , $\mathcal{M}^{\mathbf{w}}(x)$ halts in time at most $T(n)$, using at most $S(n)$ cells on the worktape and $C(n)$ cells on \mathbf{w} . Moreover, the final configuration of \mathbf{w} must be equal to its initial configuration, for every x and \mathbf{w} .*

We now define the notion of a catalytic machine that computes a function. We furthermore define the notion of partially computing a function, where on some tapes \mathbf{w} the machine can output a special failure symbol \perp .

► **Definition 10.** *For a function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$, we say a catalytic machine \mathcal{M} (**catalytically**) **computes** f if for every x and \mathbf{w} , $\mathcal{M}^{\mathbf{w}}(x) = f(x)$, and at the end of the computation \mathbf{w} is in its original state. We say that \mathcal{M} **partially (catalytically) computes** f if for every x and \mathbf{w} , $\mathcal{M}^{\mathbf{w}}(x) \in \{\perp, f(x)\}$, and at the end of the computation (no matter the output) \mathbf{w} is in its original state.*

⁴ We assume the input and output are specified to $O(\log n)$ bits of precision.

Partial catalytic computation is trivial without further restrictions (as \mathcal{M} can always output \perp), but we require it as an intermediate step in our analyses. We require a further condition on our machines, that they can revert the catalytic tape at any time without the catalytic tape traversing any new configurations:⁵

► **Definition 11.** *A catalytic machine \mathcal{M} is **reversible** if for every x and initial configuration \mathbf{w} , at any point during the execution of $\mathcal{M}^{\mathbf{w}}(x)$, the machine can receive an external REVERT signal. Let $P = P(\mathbf{w})$ denote all prior configurations of the catalytic tape during the execution of $\mathcal{M}^{\mathbf{w}}(x)$. After this signal, \mathcal{M} must reset \mathbf{w} to the original configuration, and moreover every intermediate configuration of \mathbf{w} during this process must lie in P . We require any time bound on \mathcal{M} to hold even in the case that \mathcal{M} is given the REVERT command at an arbitrary point.*

2.1 Composition of Catalytic Algorithms

We state the main result of this section, which is that catalytic algorithms can be composed without increasing the *catalytic* space usage. We must be careful when dealing with partial catalytic machines, and in this case we only obtain composition if the machines are reversible (Definition 11).

► **Theorem 12** (Composition of Partial Catalytic Machines). *Suppose reversible catalytic machines $\mathcal{M}_1, \mathcal{M}_2$ partially compute $f_1, f_2 : \{0, 1\}^n \rightarrow \{0, 1\}^n$ respectively using workspace $S(n) \geq \log(n)$, catalytic space $C(n)$, and time $T(n)$. Then there is a reversible catalytic machine \mathcal{M} that partially computes $f_2 \circ f_1$ using workspace $2S(n) + O(\log(Sn))$, catalytic space C , and time $\text{poly}(T(n))$. Moreover, $\mathcal{M}^{\mathbf{w}}(x) = \perp$ only if $\mathcal{M}_2^{\mathbf{w}}(f_1(x)) = \perp$, or there exists \mathbf{w}' such that $\mathcal{M}_1^{\mathbf{w}'}(x) = \perp$.*

Proof. We proceed roughly following the standard proof for composition of space-bounded algorithms. We maintain two sections on the worktape of size S for \mathcal{M}_1 and \mathcal{M}_2 (and auxiliary state of size $O(\log(Sn))$ to keep track of the location of read and write heads, and the FSM configuration of both machines), and a single catalytic tape \mathbf{w} .

The Simulation. We now begin to simulate $\mathcal{M}_2^{\mathbf{w}}$. We first verify that $\mathcal{M}_1^{\mathbf{w}}(x) \neq \perp$. As in the conventional composition of space-bounded algorithms, every time \mathcal{M}_2 reads its input, we run $\mathcal{M}_1^{\mathbf{w}}(x)$ on a separate section of the worktape and return the relevant bit of its output, where \mathbf{w} is the same catalytic tape used by \mathcal{M}_2 , in whatever its current configuration is at the time of the tape read. Moreover, every time \mathcal{M}_2 writes to the catalytic tape, resulting in a configuration \mathbf{w}' , we run $\mathcal{M}_1^{\mathbf{w}'}(x)$ and verify that it does not produce \perp .

Computing the Function. In the case that $\mathcal{M}_1^{\mathbf{w}'}(x) = f_1(x)$ for every configuration \mathbf{w}' that is encountered in this simulation and $\mathcal{M}_2^{\mathbf{w}}(f_1(x)) = f_2(f_1(x))$, it is easy to see that $\mathcal{M}^{\mathbf{w}}(x) = f_2(f_1(x))$. Moreover, it is clear that in this case we successfully reset the tape. Otherwise, consider the first point at which $\mathcal{M}_1^{\mathbf{w}'}(x) = \perp$. We first undo the most recent change to the catalytic tape, and send the REVERT command to \mathcal{M}_2 . Once \mathcal{M}_2 has finished reverting, return \perp . We claim that \mathcal{M}_2 successfully reverts the tape. This follows from the reversibility of \mathcal{M}_2 , and the fact that all calls \mathcal{M}_2 makes to its input during this

⁵ There are existing results related to transforming catalytic algorithms into reversible catalytic algorithms [15]. However, they do not appear to maintain worst-case runtime over the catalytic tape, which is crucial for our results.

process are correctly answered by \mathcal{M}_1 . The latter property follows as every time \mathcal{M}_2 queries its input during the revert process, \mathbf{w} is in a state that was encountered during the forward pass, and hence $\mathcal{M}_1(x)$ produced $f_1(x)$ when initialized with this catalytic configuration (as otherwise we would have aborted sooner).

Reversibility. Essentially the same argument establishes that \mathcal{M} is reversible. If we receive the REVERT command, let $P(\mathbf{w})$ be the states of the catalytic tape that have been encountered so far. First send REVERT to \mathcal{M}_1 (if operating), and once it has completed send the REVERT command to \mathcal{M}_2 . Moreover, while \mathcal{M}_2 is reverting, we claim that \mathbf{w} remains in $P(\mathbf{w})$. This follows from the fact that \mathcal{M}_2 is reversible (as any configurations it creates will lie in $P(\mathbf{w})$), and moreover every time \mathcal{M}_2 queries its input, any computation done by \mathcal{M}_1 will likewise keep \mathbf{w} in $P(\mathbf{w})$, as we already called \mathcal{M}_1 with this starting configuration in the forward pass (and \mathcal{M}_1 will not produce \perp , as otherwise we would have already aborted). Thus, the composed algorithm is reversible.

Time and Space. We now argue the space and time are as claimed. There are a constant number of pointers (which we maintain on the worktape) to track the number of bits output by \mathcal{M}_1 , current tape heads, and other information. The fact that the catalytic tape size is preserved is immediate. The call overhead adds at most a polynomial factor in the runtime, as we run \mathcal{M}_1 at most once per step of \mathcal{M}_2 . Finally, if \mathcal{M}_1 computes $f_1(x)$ for every catalytic tape and \mathcal{M}_2 computes correctly on \mathbf{w} , we successfully compute $f_2 \circ f_1$ as claimed. ◀

We derive an easy corollary in the case of multiple composition:

► **Corollary 13.** *Suppose a reversible catalytic machine \mathcal{M} partially computes $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$ using workspace $S(n) \geq \log n$, catalytic space C , and time 2^S . Then there exists a reversible catalytic machine \mathcal{M}' that partially computes f^ℓ using workspace $O(\ell \cdot S)$, catalytic space C , and time $2^{O(\ell \cdot S)}$. Moreover, for x where $\mathcal{M}^{\mathbf{w}}(f^i(x)) = f(f^i(x))$ for every \mathbf{w} and $i \in \{0, \dots, \ell - 1\}$, $\mathcal{M}'^{\mathbf{w}}(x) = f^\ell(x)$.*

3 Catalytic Derandomization From Conditional Compression

In this section we prove Theorem 2. We state all our results in terms of catalytic algorithms for the stochastic matrix powering problem, as it is easily compatible with the recursive framework we implement later. Recall a nonnegative matrix is **stochastic** (resp. **substochastic**) if all row sums are 1 (resp. at most 1). For a set S , let U_S be the uniform distribution over S , and let $U_n = U_{\{0,1\}^n}$.

► **Theorem 14.** *There is a CTISP $[n, \log n, \log^2 n]$ algorithm that, given n and a stochastic matrix $M \in [0, 1]^{n \times n}$ where each entry is specified with $O(\log n)$ bits of precision, outputs $\widetilde{M} \in [0, 1]^{n \times n}$ such that $\left\| \widetilde{M} - M^n \right\|_1 \leq 1/n$.*

We recall the existence of efficient algorithms which canonicalize (sub)stochastic matrices, essentially reducing the stochastic matrix powering problem to producing a PRG that fools a branching program.

► **Lemma 15** ([21, 19, 7]). *There is a constant $c > 0$ and a space $O(\log nw/\varepsilon)$ algorithm which, given $\varepsilon > 0$ and $n, w \in \mathbb{N}$ where $w \geq n$ and a substochastic matrix $M \in [0, 1]^{w \times w}$ with $O(\log w)$ bits of precision, returns a branching program*

$$\overline{B} : [(w/\varepsilon)^c] \times \{0, 1\}^m \rightarrow [(w/\varepsilon)^c]$$

4:10 Derandomizing Logspace with a Small Shared Hard Drive

of width $w' = (w/\varepsilon)^c$ and length $m = n \cdot O(\log(w/\varepsilon))$ where m is a power of two. Moreover, letting $\widetilde{M} \in [0, 1]^{w \times w}$ be the (substochastic) matrix where for $i, j \in [w]$ we define⁶ $\widetilde{M}_{i,j} = \Pr_{x \leftarrow U_n}[\overline{B}[i, x] = j]$, we have $\left\| \widetilde{M} - M^n \right\|_1 \leq \varepsilon$.

As this is not the way these results are stated, we provide a translation in Appendix A. We next define the Nisan PRG, and recall several auxiliary lemmas.

The Nisan PRG

Given a branching program, we first define the larger alphabet program obtained from duplicating each edge:

► **Definition 16.** For $t \in \mathbb{N}$, for $\overline{B} : [w] \times \{0, 1\}^n \rightarrow [w]$ of width w , let $\overline{B}_t : [w] \times (\{0, 1\}^t)^n \rightarrow [w]$ be the branching program of length n and width w over alphabet $\{0, 1\}^t$ with transition function $B_t[a, y] = B[a, y_1]$, where y_1 is the first bit of $y \in \{0, 1\}^t$. Note that \overline{B}_t can be constructed in space $O(\log t n w)$ given \overline{B} , and furthermore for every $i, j \in [w]$, $\Pr_{x \leftarrow U_n}[\overline{B}[i, x] = j] = \Pr_{x \leftarrow U_{(\{0, 1\}^t)^n}}[\overline{B}_t[i, x] = j]$.

We recall a pairwise independent hash family with a very efficient description:

► **Observation 17.** For every $t \in \mathbb{N}$, there exists a pairwise independent hash family $\mathcal{H} : \{0, 1\}^t \rightarrow \{0, 1\}^t$ such that $|\mathcal{H}| = 2^{2t}$, and $h \in \mathcal{H}$ (which we associate with $h \in \{0, 1\}^{2t}$) can be evaluated in space $O(t)$.

Given a (hash) function $h : \{0, 1\}^t \rightarrow \{0, 1\}^t$ and a program \overline{B} , we define an operator that applies a single level of the Nisan construction with hash function h .

► **Definition 18.** Given $\overline{B}_t : [w] \times (\{0, 1\}^t)^n \rightarrow [w]$ and $h : \{0, 1\}^t \rightarrow \{0, 1\}^t$, let $\overline{B}_{t,h} : [w] \times (\{0, 1\}^t)^{n/2} \rightarrow [w]$ be the width w , length $n/2$ program with transition function

$$B_{t,h}[a, x] = B_t[B_t[a, x], h(x)].$$

Using a recursive application of hash functions, we can define the Nisan PRG as follows.

► **Definition 19.** For $(h_1, \dots, h_\ell) \in \mathcal{H}_t$, define $\text{NIS}_{(h_1, \dots, h_\ell)} : \{0, 1\}^t \rightarrow \{0, 1\}^{t \cdot n}$ inductively as follows. Let $\text{NIS}_0(x) = x_1$, and for $j \in [\ell]$

$$\text{NIS}_{(h_1, \dots, h_j)}(x) = (\text{NIS}_{(h_1, \dots, h_{j-1})}(x) \parallel \text{NIS}_{(h_1, \dots, h_{j-1})}(h_j(x))).$$

Note that $B[\cdot, \text{NIS}_{(h_1, \dots, h_\ell)}(\cdot)]$ and $B_{t, (h_1, \dots, h_\ell)}[\cdot, \cdot]$ (as defined in Definition 18) are equal as functions.

To analyze the Nisan PRG, we define the notion of a hash function being good for composing two functions, and a PRG being good for a function.

► **Definition 20.** For every $n, w, t \in \mathbb{N}$ and $\delta > 0$ and $\overline{f} : [w] \times (\{0, 1\}^t)^n \rightarrow [w]$ and $G : \{0, 1\}^t \rightarrow (\{0, 1\}^t)^n$, we say that G is δ -good for \overline{f} if for every $i, j \in [w]$,

$$\left| \Pr_{x \leftarrow U_t}[\overline{f}[i, G(x)] = j] - \Pr_{x \leftarrow U_{(\{0, 1\}^t)^n}}[\overline{f}[i, x] = j] \right| \leq \delta.$$

Moreover, we say $h \in \mathcal{H}$ is δ -good (and δ -bad otherwise) for $f : [w] \times \{0, 1\}^t \rightarrow [w]$ if $G(x) = (x \parallel h(x))$ is δ -good for $\overline{f}[i, (x, y)] = f[i, x, y]$.

⁶ Note that this truncates the size from w' back to w .

We recall that a random hash function is good with high probability.

► **Lemma 21** ([16]). *For every f , $\Pr_{h \leftarrow \mathcal{H}_t}[h \text{ is } \delta\text{-good for } f] \geq 1 - w^5(1/\delta)^2/2^t$.*

(We provide a proof in Appendix A.) Moreover, a hybrid argument establishes the following.

► **Lemma 22** ([16]). *For every $\overline{B}_t : [w] \times (\{0, 1\}^t)^n \rightarrow [w]$ and $\vec{h} = (h_1, \dots, h_\ell)$, suppose for every $i \in [\ell]$, h_i is δ -good for $B_{t, h_1, \dots, h_{i-1}}$. Then $\text{NIS}_{\vec{h}}$ is $\delta \cdot nw$ -good for \overline{B} .*

Catalytic Derandomization

We now state the main result that powers both of our derandomizations.

► **Theorem 23.** *There is a pair of reversible catalytic algorithms \mathcal{A}, \mathcal{D} that run in workspace $O(\log nw/\varepsilon)$, catalytic space $O(\log(n) \cdot \log(nw/\varepsilon))$, and time $\text{poly}(nw/\varepsilon)$ and act as follows. Given $\varepsilon > 0$ and a length $n = 2^\ell$, width w ROBP $\overline{B} : [w] \times \{0, 1\}^n \rightarrow [w]$ where $w \geq n$:*

■ *The machine $\mathcal{A}^{\mathbf{w}}(\overline{B})$ outputs $V \in \{0, 1\}^{2^\ell}$ and $t = O(\log nw/\varepsilon)$ and sets the catalytic tape to \mathbf{w}' , such that (\mathbf{w}', V) contains a (read-only) data structure supporting access to hash functions $\vec{h} = (h_1, \dots, h_\ell)$ each on t bits, such that $\text{NIS}_{\vec{h}}$ is ε -good for \overline{B} .*

■ *The machine $\mathcal{D}^{\mathbf{w}'}$ (\overline{B}, V) sets the final catalytic tape configuration to \mathbf{w} .*

For the extension to search problems, we require a version which takes in a list of ROBP, and constructs a set of hash functions that is simultaneously good for all of them. Our proof is for this more general notion, which immediately implies Theorem 23:

► **Theorem 24.** *There is a pair of reversible catalytic algorithms \mathcal{A}, \mathcal{D} that run in workspace $O(\log nw/\varepsilon)$, catalytic space $O(\log(n) \cdot \log(nw/\varepsilon))$, and time $\text{poly}(nw/\varepsilon)$ and act as follows. Given $\varepsilon > 0$ and a set of w length $n = 2^\ell$, width w RBPs*

$$\mathcal{B} = (\overline{B}^1, \dots, \overline{B}^w)$$

with $\overline{B}^j : [w] \times \{0, 1\}^n \rightarrow [w]$ where $w \geq n$:

■ *The machine $\mathcal{A}^{\mathbf{w}}(\mathcal{B})$ outputs $V \in \{0, 1\}^{2^\ell}$ and $t = O(\log nw/\varepsilon)$ and sets the catalytic tape to \mathbf{w}' , such that (\mathbf{w}', V) contains a (read-only) data structure supporting access to hash functions $\vec{h} = (h_1, \dots, h_\ell)$ each on t bits, such that $\text{NIS}_{\vec{h}}$ is ε -good for \overline{B}^j for every $j \in [w]$.*

■ *The machine $\mathcal{D}^{\mathbf{w}'}$ (\mathcal{B}, V) sets the final catalytic tape configuration to \mathbf{w} .*

To make our compression and decompression algorithms work, we require that we can determine if a hash function is good for a branching program at a certain level of the Nisan construction, given pointers to the hash functions at the previous levels:

► **Proposition 25.** *There is a space $O(t + \log(w/\delta))$ algorithm that, given $n, w, t \in \mathbb{N}$ with $w \geq n$ and $\vec{h} \in \{0, 1\}^{2t}$ and $\overline{B} : [w] \times \{0, 1\}^n \rightarrow [w]$ and (read only) \mathbf{w} and pointers p_1, \dots, p_r such that $\mathbf{w}_{[p_i, \dots, p_i+2t]} = h_i$ represents a hash function on t bits, returns if \vec{h} is δ -good for $B_{t, (h_1, \dots, h_r)}$.*

We give a proof in Appendix A, as it essentially follows from the argument of [18]. We can then prove the theorem:

Proof of Theorem 24. We assume without loss of generality that n, w and $1/\varepsilon$ are powers of two. Set

$$t_0 = 60 \log(w/\varepsilon), \quad t_1 = 25 \log(w/\varepsilon), \quad \delta = \varepsilon/nw \geq \varepsilon/w^2,$$

4:12 Derandomizing Logspace with a Small Shared Hard Drive

and note that we choose t_1 large enough such that a good series of hash functions (for all \overline{B}^j simultaneously) on t_1 bits always exists. The algorithm works as follows. First, virtually divide the catalytic tape as:

$$\mathbf{w} = (\mathbf{w}^1 || \mathbf{w}^2 || \dots || \mathbf{w}^{2^\ell})$$

where $|\mathbf{w}^i| = 2t_0$, which we think of as initially holding $h : \{0, 1\}^{t_0} \rightarrow \{0, 1\}^{t_0}$. Note that $|\mathbf{w}| = \ell \cdot 4t_0 = O(\log(n) \log(nw/\varepsilon))$ as claimed.

Next, initialize $V \in \{0, 1, *\}^{2^\ell}$ to indicate if each block is compressed, uncompressed, or unprocessed respectively. The first two cases correspond to the following two formats of the block:

$$\mathbf{w}^i = \begin{cases} h & V_i = 1 \\ (z || j || 0^{50 \log(w/\varepsilon)}) & V_i = 0 \end{cases}$$

Informally, the first corresponds to block i originally containing a good hash function for \mathcal{B} , and the second corresponds to block i originally containing a bad hash function for \overline{B}^j , which is thus compressible (in fact, z represents a compressed version of the original data). We define notation for the set of blocks in each configuration:

► **Definition 26.** For $b \in \{0, 1\}$, let $I^b(V) \subseteq [2^\ell]$ correspond to the indices such that $V_i = b$, and let $S^b(V) = |I^b(V)|$.

Next, we initialize a counter $i = 1$ for the current block. We then iterate over $i = \{1, \dots, 2^\ell\}$ until $\max\{S^1(V), S^0(V)\} = \ell$.⁷ For each i , the algorithm works as follows. Let $\tilde{h} = \mathbf{w}^i$ be the hash function (on t_0 bits) obtained from the current block. We then test if \tilde{h} is δ -good for

$$f^j = B_{t_0, \vec{h}_p}^j, \text{ for every } j \in [w].$$

Where

$$\vec{h}_p = (\mathbf{w}^{I^1(V)_1}, \dots, \mathbf{w}^{I^1(V)_{S^1(V)}})$$

corresponds to the hash functions on the preceding good blocks, and B_{t_0, \vec{h}_p}^j is defined as in Definition 18 applied to \overline{B}^j and \vec{h}_p . As the index set $I^1(V)$ is easy to generate given V , this test can be performed in space $O(\log nw/\varepsilon)$ without modifying the catalytic tape (and hence also in time $\text{poly}(nw/\varepsilon)$), by Proposition 25.

Given the results of this test, we break into cases depending on if \tilde{h} is good:

- If \tilde{h} is δ -good for f^j for every $j \in [w]$, set $V_i = 1$.
- If \tilde{h} is δ -bad for some f^j , set $V_i = 0$ and let j be the first index for which this holds. Next, by enumeration over strings $h \in \{0, 1\}^{2t_0}$ (which we can do using the workspace), determine the index of \tilde{h} in the set

$$\mathbf{BAD}_{i,j} = \left\{ h \in \{0, 1\}^{2t_0} : h \text{ is } \delta\text{-bad for } B_{t_0, \vec{h}_p}^j \right\}$$

where we again perform this test using Proposition 25. Letting the index of \tilde{h} in this set be z , write

$$\mathbf{w}^i = (z || j || 0^{50 \log(w/\varepsilon)})$$

⁷ If we exit before $i = 2^\ell$, set the remaining indices of V to an arbitrary value, which we ignore for clarity of presentation.

(we perform this write operation left to right, and will revert it right to left). We denote the final $50 \log(w/\varepsilon)$ bits as free space.

Finally, we claim that we can in fact write these quantities in space $|\mathbf{w}^i| = 2t_0$. The index j requires $\log(w)$ bits. Moreover, we have

$$\begin{aligned} |\mathbf{BAD}_{i,j}| &= 2^{2t_0} \cdot \Pr_{h \leftarrow \mathcal{H}} [h \text{ is } \delta\text{-bad for } f] \\ &\leq 2^{2t_0} \cdot w^5 (1/\delta)^2 / 2^{t_0} && \text{(Lemma 21)} \\ &\leq 2^{2t_0} \cdot (w/\varepsilon)^{7-60} \end{aligned}$$

And thus $\log |\mathbf{BAD}_{i,j}| \leq 2t_0 - 51 \log(w/\varepsilon) = |\mathbf{w}^i| - 51 \log(w/\varepsilon)$. Therefore, we can record all required information as claimed.

After processing all blocks, we obtain a catalytic tape \mathbf{w}' and one of two cases:

- If $S^1(V) = \ell$, there exist $\vec{h} = (h_1, \dots, h_\ell)$ corresponding to the hash functions (on t_0 bits) in $I^1(V)$, and these functions are easy to recover from V .
- Else, we must have $S^0(V) = \ell$.

For $i \in I^0(V)$, let F_i be the $50 \log(w/\varepsilon)$ free bits in \mathbf{w}^i . Note that a description of a hash function $h : \{0, 1\}^{t_1} \rightarrow \{0, 1\}^{t_1}$ is of size $|F_i|$. Iterating over $i \in I^0(V)$ in increasing order, we find (via brute force enumeration) a hash function \tilde{h} that is δ -good for

$$f^j = B_{t_1, h_p}^j, \text{ for every } j \in [w].$$

Where

$$\vec{h}_p = \left(\mathbf{w}_{F_{I^0(V)_1}}, \dots, \mathbf{w}_{F_{I^0(V)_{i-1}}} \right)$$

corresponds to the (δ -good) hash functions stored on the free space in the preceding indices of $I^0(V)$. Next, store \tilde{h} in \mathbf{w}_{F_i} . Such a good hash function always exists, by our choice of t_1 and Lemma 21, and moreover testing if each candidate is good can be computed in the desired space and time by Proposition 25. After this processing,

$$\left(\mathbf{w}_{F_{I^0(V)_1}}, \dots, \mathbf{w}_{F_{I^0(V)_\ell}} \right)$$

contain ℓ good hash functions, which we can clearly access in read-only fashion given V and \mathbf{w}' .

Thus, in both cases we obtain a set of hash functions $\vec{h} = (h_1, \dots, h_\ell)$ on $t = O(\log nw/\varepsilon)$ bits that is δ -good for every one of the relevant tests, so by Lemma 22 we have that $\text{NIS}_{\vec{h}}$ is $\delta \cdot nw \leq \varepsilon$ -good for \vec{B}^j for every j .

Decompression and Reversibility. It suffices to show that at any point, the algorithm can revert the tape to the original configuration \mathbf{w} (and then $\mathcal{D}(\mathcal{B}, V)$ simply issues the REVERT command). No matter the present configuration, we iterate through $I^0(V)$ in descending order. Letting the current index be $i \in I^0(V)$, recall this block is of form $(\mathbf{w}')^i = (z || j || *)$. First write $0^{50 \log(w/\varepsilon)}$ to the last indices (in reverse order to satisfy reversibility), such that we reach the configuration after compressing the block. Then enumerate over $h \in \{0, 1\}^{2t_0}$ using workspace $O(t_0 + \log(w/\varepsilon))$, until we find the hash with index z in $\mathbf{BAD}_{i,j}$, where $\mathbf{BAD}_{i,j}$ and B_{t_0, h_p}^j are defined as before (which we still have access to because we reset the tape in reverse order), and we determine membership by Proposition 25.

Once we find this h , write $(\mathbf{w}')^i = h$ (in the reverse order to satisfy reversibility) and proceed to the next highest index in $I^0(V)$. After this process has completed, it is clear from construction that \mathbf{w} has been reset to the original configuration, and that the tape never reaches a new intermediate configuration during this process.

4:14 Derandomizing Logspace with a Small Shared Hard Drive

Time and Space. In every step of the computation, we perform at most $\text{poly}(2^{t_0}nw/\varepsilon)$ work to determine if a hash function is good, find the index of a bad hash function, or find a good hash function. Moreover, as at every point we store at most a constant number of hash functions on the worktape, the space consumption follows. ◀

It is easy to go from Theorem 23 to Theorem 14.

Proof of Theorem 14. Let $\overline{B} : [\text{poly}(n)] \times \{0, 1\}^{n^c} \rightarrow [\text{poly}(n)]$ be the ROBP obtained from applying Lemma 15 to M with $n = n$, $w = n$, and $\varepsilon = 1/2n$. We then call Theorem 23 with $\overline{B} = \overline{B}$ and $\varepsilon = 1/2n^2$. Let $\vec{h} = (h_1, \dots, h_{c \log n})$ be the hash functions obtained from this call, which we have implicit access to via the current state of the catalytic tape \mathbf{w}' and V , and let $t = O(\log n)$ be the domain of the hash functions. Then enumerate over $x \in \{0, 1\}^t$ and for $i, j \in [w]$ let

$$\widetilde{M}_{i,j} = \Pr_{x \leftarrow U_t} [\overline{B}[i, \text{NIS}_{\vec{h}}(x)] = j].$$

By Lemma 15 and Theorem 23, we have the guarantee that

$$\left\| \widetilde{M} - M^n \right\|_1 \leq \frac{1}{2n} + n \cdot \frac{1}{2n^2} = 1/n. \quad \blacktriangleleft$$

Finally, we can use Theorem 24 to prove Theorem 7.

Proof of Theorem 7. The algorithm works as follows. Given $\mathcal{B} = (\overline{B}^1, \dots, \overline{B}^n)$ of width and length n where $\mathbb{E} [\overline{B}^j(U_n)] \geq 1 - 1/n^2$ for every j , we call Theorem 24 with $\mathcal{B} = \mathcal{B}$ and $\varepsilon = 1/2n^2$. Let $\vec{h} = (h_1, \dots, h_{\log n})$ be the hash functions obtained from this call, which we have implicit access to via the current state of the catalytic tape \mathbf{w}' and V , and let $t = O(\log n)$ be the length of the domain of the hash functions. We then claim there is some $x \in \{0, 1\}^t$ such that for all $j \in [n]$,

$$\overline{B}^j(\text{NIS}_{\vec{h}}(x)) = 1.$$

We have this as

$$\begin{aligned} \Pr_{x \leftarrow U_t} \left[\bigwedge_{j \in [n]} \overline{B}^j(\text{NIS}_{\vec{h}}(x)) \right] &\geq 1 - \sum_{j \in [n]} \Pr_{x \leftarrow U_t} [\overline{B}^j(\text{NIS}_{\vec{h}}(x)) = 0] \\ &\geq 1 - \sum_{j \in [n]} \left(\Pr_{x \leftarrow U_t} [\overline{B}^j(U_n) = 0] + \frac{1}{2n^2} \right) \\ &\geq 1 - n(2/n^2) > 0 \end{aligned}$$

where the second inequality follows from our choice of ε . Then it is simple to find such an x by enumeration, whereupon we compute and return $\text{NIS}_{\vec{h}}(x) \in \{0, 1\}^n$, which is precisely the solution for the mutual ROBP hitting problem. Finally, we use Theorem 24 to reset the tape. ◀

4 Catalytic Recursive Matrix Powering

We now transform Theorem 23 into a parameterized algorithm for matrix powering.

► **Theorem 27.** *There is a catalytic machine that, given r_1, r_2 such that $r_1 r_2 = \log(n)$ and a stochastic matrix $M \in [0, 1]^{n \times n}$ where each entry is specified with $l = O(\log n)$ bits of precision, uses workspace $O(r_2 \cdot \log(n))$, catalytic space $O(r_1 \cdot \log(n))$, and time $n^{O(r_2)}$, and outputs \tilde{M} such that $\|\tilde{M} - M^n\| \leq 1/n$.*

Theorem 27 immediately implies Theorem 3 by setting $r_2 = \log^{\alpha-1}(n)$ and $r_1 = \log^{2-\alpha}(n)$ for $\alpha \in [1, 1.5]$, and using the standard transformation of estimating the acceptance probability of a **BPL** machine via stochastic matrix powering.

We first prove there exists an algorithm which computes a single intermediate power. We must be careful to ensure that the algorithm satisfies the requirements of (partial) catalytic computation. In particular, if the machine ever outputs an answer (rather than \perp), this must be the only possible answer for this input, over all possible catalytic tapes. Simultaneously, we must ensure that the vast majority of inputs never return \perp no matter the initial catalytic tape configuration.

We achieve this dual guarantee using an idea from Saks and Zhou [21]. For a given input matrix M , we additionally take in a shift $s \in [0, \delta]$ for $\delta = 1/\text{poly}(w/\varepsilon)$. After computing an approximate power of M , we add s to each entry, and then truncate each entry to $O(\log w/\varepsilon)$ bits of precision. In fact, we first verify that our shifted approximate power is sufficiently far from the rounding threshold, and if not return \perp . By doing so, we algorithmically verify that we will never round to different thresholds over different \mathbf{w} . Unfortunately, for some pairs (M, s) it may be the case that we detect possible mis-rounding for some tapes \mathbf{w} , even if all possible approximations lie inside a single rounding interval. This can result in returning \perp on some tapes and a (consistent) value otherwise. However, we show that we can choose the magnitude of s such that with high probability over s this does not occur, and we always return a (consistent) value.

► **Theorem 28.** *For every $n, w \in \mathbb{N}$ and $\varepsilon > 0$ where $w \geq n \geq \log w$ and $2^{-n} > \varepsilon > 0$, there is a reversible catalytic machine \mathcal{P} that uses workspace $O(\log w/\varepsilon)$, catalytic space $O(\log(n) \log(nw))$, and time $\text{poly}(nw/\varepsilon)$. The machine takes input $s \in \{0, 1\}^{O(\log(w/\varepsilon))}$ and a substochastic matrix $M \in [0, 1]^{w \times w}$, where each entry of M is specified with $l = O(\log w/\varepsilon)$ bits of precision. Moreover:*

- *For every (s, M) , there is substochastic \tilde{M}_s (defined without reference to \mathbf{w}) with l bits of precision satisfying $\|\tilde{M}_s - M^n\|_1 \leq \varepsilon$ such that for every \mathbf{w} ,*

$$\mathcal{P}^{\mathbf{w}}(s, M) \in \{\perp, \tilde{M}_s\}.$$

- *For every M , $\Pr_s[\exists \mathbf{w}, \mathcal{P}^{\mathbf{w}}(s, M) = \perp] \leq 1/w^2$.*

Proof. Let $\bar{B} : [(w/\delta)^c] \times \{0, 1\}^{m=n \cdot O(\log(w/\delta))} \rightarrow [(w/\delta)^c]$ be the result of Lemma 15 applied to M with $n = n$ and $\varepsilon = \delta$ to be chosen later. We compose the output of this algorithm with Theorem 23, applied with $\bar{B} = \bar{B}$ and $w' = (w/\delta)^c$ and $n' = m = O(n^3)$ and $\varepsilon = \delta$ to be chosen later. Let $\vec{h} = (h_1, \dots, h_\ell)$ be the hash functions obtained from this call, which we have implicit access to via the current state of the catalytic tape \mathbf{w}' and V , and let $t = O(\log w/\delta)$ be the domain of the hash functions. Then enumerate over $x \in \{0, 1\}^t$ and for every $i, j \in [w]$ let

$$\tilde{M}_{i,j} = \Pr_{x \leftarrow U_t} [\bar{B}[i, \text{NIS}_{\vec{h}}(x)] = j].$$

Next, define $\tau = (s \cdot 2^{-2k}) \cdot J$ where $J = 1^{w \times w}$, interpreting $s \in [2^k]$. We next check if any entry of $\tilde{M} + \tau$ is within 2δ of a multiple of 2^{-l} , our rounding threshold. In this case, run $\mathcal{D}^{\mathbf{w}'}(\bar{B}, V)$ to reset the tape and return \perp . Otherwise, let

4:16 Derandomizing Logspace with a Small Shared Hard Drive

$$\widetilde{M}_s = \lfloor \widetilde{M} + \tau \rfloor_l$$

where $\lfloor \cdot \rfloor_l$ rounds each entry down to l bits of precision, and decreases the largest entry per row such that the final matrix is substochastic. Let this matrix be \widetilde{M}_s . Finally, run $\mathcal{D}^{\mathbf{w}'}(\widetilde{B}, V)$, and return \widetilde{M}_s .

Accuracy. By our choice of error in Theorem 23 and Lemma 15, we have that

$$\left\| \widetilde{M} - M^n \right\|_1 \leq 2w\delta$$

and moreover \widetilde{M} has each row sum at most 1. Furthermore, perturbing by τ and rounding down the largest entry causes an ℓ_1 error of at most $2w \cdot 2^{-k}$. Finally, rounding each entry down to a multiple of 2^{-l} causes a total error of at most $w \cdot 2^{-l}$, so

$$\left\| \widetilde{M}_s - M^n \right\| \leq 2w\delta + 2w \cdot 2^{-k} + w \cdot 2^{-l} \leq \varepsilon$$

Where the final inequality comes from choosing

$$l = O(\log(w/\varepsilon)), \quad k = 10 \cdot l, \quad \delta = 2^{-2k}.$$

Uniqueness. We claim that for every (s, M) , there is at most 1 possible non- \perp output over all choices of \mathbf{w} (which we denote \widetilde{M}_s in the theorem statement). Let $\widetilde{M}_{\mathbf{w}}, \widetilde{M}_{\mathbf{w}'}$ be the result of Theorem 23 on M initialized with catalytic tapes \mathbf{w}, \mathbf{w}' . By the accuracy guarantee of Theorem 23, for every i, j we have

$$\left| (\widetilde{M}_{\mathbf{w}} + \tau)_{i,j} - (\widetilde{M}_{\mathbf{w}'} + \tau)_{i,j} \right| \leq \left| (\widetilde{M}_{\mathbf{w}} + \tau)_{i,j} - (M + \tau)_{i,j} \right| + \left| (M + \tau)_{i,j} - (\widetilde{M}_{\mathbf{w}'} + \tau)_{i,j} \right| \leq 2\delta$$

Thus, if $(\widetilde{M}_{\mathbf{w}'} + \tau)_{i,j}$ is greater than 2δ from a multiple of 2^{-l} , we can be certain that no tape \mathbf{w}' will induce an estimate that falls on the other side of the threshold, and hence all non- \perp outputs will be rounded consistently.

Success Probability. Furthermore, we argue that for every M , with probability at least $1 - 1/w^2$ over s we return \widetilde{M}_s (not \perp) for every initial tape configuration. Fixing arbitrary s and $i, j \in [w]$, if $(M + \tau)_{i,j}$ is at least 3δ from every multiple of 2^{-l} , every \mathbf{w} will induce an estimate $(\widetilde{M} + \tau)_{i,j}$ that is at least 2δ from every multiple of 2^{-l} , and hence for every \mathbf{w} we will not produce \perp due to this entry. This occurs for every i, j simultaneously with probability at least

$$w^2 \cdot 2^l \cdot \frac{6\delta \cdot 2^k + 2}{2^k} \ll 1/w^2.$$

Time and Space. It is clear the algorithm runs in the claimed time and space bound, given Theorem 23.

Reversibility. As the only components of the algorithm that write to the catalytic tape are calls to Theorem 23, reversibility follows immediately from the equivalent result for that algorithm. \blacktriangleleft

We can then prove the main result.

► **Theorem 27.** *There is a catalytic machine that, given r_1, r_2 such that $r_1 r_2 = \log(n)$ and a stochastic matrix $M \in [0, 1]^{n \times n}$ where each entry is specified with $l = O(\log n)$ bits of precision, uses workspace $O(r_2 \cdot \log(n))$, catalytic space $O(r_1 \cdot \log(n))$, and time $n^{O(r_2)}$, and outputs \widetilde{M} such that $\|\widetilde{M} - M^n\| \leq 1/n$.*

Proof. Let $\vec{s} = (s_1, \dots, s_{r_2}) \in \{0, 1\}^{r_2 \cdot O(\log n)}$ be a vector of random shifts. Let $\widetilde{M}_0 = M$ and for $i \in [r_2]$ recursively define

$$\widetilde{M}_i = f\left(\widetilde{M}_{i-1}, s_i\right),$$

where f is the function defined by Theorem 28 with $\varepsilon = 1/n^3$ and $n = 2^{r_1}$. An easy inductive proof [19] establishes that, letting

$$\left\| \widetilde{M}_i - M^{2^{r_1 \cdot i}} \right\|_1 = \delta_i$$

we have $\delta_{i+1} \leq 1/n^3 + 2^{r_1} \cdot \delta_{i-1} \leq 2^{r_1+1} \cdot \delta_{i-1}$, and hence $\delta_{r_2} \leq 1/n$.

The final algorithm iterates over \vec{s} and computes \widetilde{M}_{r_2} by applying recursive composition of space-bounded machines Corollary 13 to the algorithm of Theorem 28 as defined above.⁸ The algorithm returns the first non- \perp output. The fact that the algorithm is catalytic follows from Corollary 13 and Theorem 28. Next, we claim there is some \vec{s} where the algorithm returns a value. Note that s_i is chosen obliviously to \widetilde{M}_{i-1} , and so with probability at least $1/n^2$ over s_i , on input $(\widetilde{M}_{i-1}, s_i)$ the algorithm returns \widetilde{M}_i (i.e. not \perp) when run with every possible catalytic tape. Thus, there is some \vec{s} where every level computes correctly.

Time and Space. Every application of Theorem 28 occurs with parameters $n = 2^{r_1}$ and $w = n$ and $\varepsilon = 1/n^3$, such that the algorithm uses workspace $O(r_1 + \log(n)) = O(\log n)$, catalytic space $O(r_1 \cdot (r_1 + \log(n))) = O(r_1 \cdot \log n)$, and time $\text{poly}(n)$, and moreover the shift s for each level is of length $O(\log n)$. Applying Corollary 13, we obtain that the composed algorithm uses workspace $O(r_2 \cdot \log(n) + |\vec{s}|) = O(r_2 \cdot \log n)$, catalytic space $O(r_1 \cdot \log n)$, and runs in time $n^{O(r_2)}$ as claimed. ◀

References

- 1 Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003. doi:10.1016/S0022-0000(03)00025-4.
- 2 Sagar Bisoyi, Krishnamoorthy Dinesh, and Jayalal Sarma. On pure space vs catalytic space. *Theor. Comput. Sci.*, 921:112–126, 2022. doi:10.1016/j.tcs.2022.04.005.
- 3 Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In *Symposium on Theory of Computing, STOC 2014*, pages 857–866. ACM, 2014.
- 4 Harry Buhrman, Michal Koucký, Bruno Loff, and Florian Speelman. Catalytic space: Non-determinism and hierarchy. *Theory Comput. Syst.*, 62(1):116–135, 2018. doi:10.1007/s00224-017-9784-7.
- 5 Jin-yi Cai, Venkatesan T. Chakaravarthy, and Dieter van Melkebeek. Time-space tradeoff in derandomizing probabilistic logspace. *Theory Comput. Syst.*, 39(1):189–208, 2006. doi:10.1007/S00224-005-1264-9.
- 6 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.

⁸ We can define the machine to take the entire shift vector and a pointer to the index it should use, such that we are recursively composing exactly the same machine, but we suppress this for clarity.

- 7 Gil Cohen, Dean Doron, Ori Sberlo, and Amnon Ta-Shma. Approximating iterated multiplication of stochastic matrices in small space. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023*, pages 35–45. ACM, 2023.
- 8 James Cook and Ian Mertz. Catalytic approaches to the tree evaluation problem. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020*, pages 752–760. ACM, 2020.
- 9 James Cook and Ian Mertz. Tree evaluation is in space $o(\log n \cdot \log \log n)$. *Electron. Colloquium Comput. Complex.*, TR23-174, 2023. [arXiv:TR23-174](#).
- 10 Samir Datta, Chetan Gupta, Rahul Jain, Vimal Raj Sharma, and Raghunath Tewari. Randomized and symmetric catalytic computation. In *Computer Science – Theory and Applications – 15th International Computer Science Symposium in Russia, CSR 2020*, pages 211–223. Springer, 2020.
- 11 Dean Doron, Edward Pyne, and Roei Tell. Opening up the distinguisher: A hardness to randomness approach for $BPL = L$ that uses properties of BPL. *Electron. Colloquium Comput. Complex.*, TR23-208, 2023. [arXiv:TR23-208](#).
- 12 Chetan Gupta, Rahul Jain, Vimal Raj Sharma, and Raghunath Tewari. Unambiguous catalytic computation. In *39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2019*, volume 150 of *LIPICs*, pages 16:1–16:13, 2019.
- 13 Daniel M. Kane, Raghu Meka, and Jelani Nelson. Almost optimal explicit johnson-lindenstrauss families. 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 628–639. Springer, 2011. [doi:10.1007/978-3-642-22935-0_53](#).
- 14 Daniel M. Kane and Jelani Nelson. Sparser johnson-lindenstrauss transforms. *J. ACM*, 61(1):4:1–4:23, 2014. [doi:10.1145/2559902](#).
- 15 Ian Mertz. Reusing space: Techniques and open problems. *Bulletin of EATCS*, 141(3), 2023.
- 16 Noam Nisan. Pseudorandom generators for space-bounded computation. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 204–212. ACM, 1990. [doi:10.1145/100216.100242](#).
- 17 Noam Nisan. On read-once vs. multiple access to randomness in logspace. *Theor. Comput. Sci.*, 107(1):135–144, 1993. [doi:10.1016/0304-3975\(93\)90258-U](#).
- 18 Noam Nisan. $RL \leq SC$. *Comput. Complex.*, 4:1–11, 1994. [doi:10.1007/BF01205052](#).
- 19 Aaron (Louie) Putterman and Edward Pyne. Near-optimal derandomization of medium-width branching programs. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023*, pages 23–34, 2023.
- 20 Edward Pyne, Ran Raz, and Wei Zhan. Certified hardness vs. randomness for log-space. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 989–1007. IEEE, 2023. [doi:10.1109/FOCS57990.2023.00061](#).
- 21 Michael E. Saks and Shiyu Zhou. $BP_HSpace(S) \subseteq DSPACE(S^{3/2})$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999. [doi:10.1006/jcss.1998.1616](#).
- 22 D. Sivakumar. Algorithmic derandomization via complexity theory. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 619–626. ACM, 2002. [doi:10.1145/509907.509996](#).

A Proofs of Lemmas

We first prove that a hash function drawn from a pairwise independent hash family is good for a function with high probability. To do this, we recall the hash mixing lemma:

► **Lemma 29** ([16]). *Let $A, A' \subseteq \{0, 1\}^t$ be arbitrary subsets of density $\rho = |A|/2^t$ and $\rho' = |A'|/2^t$. Then for every $\delta > 0$,*

$$\Pr_{h \leftarrow \mathcal{H}} \left[\left| \Pr_{x \leftarrow U_t} [x \in A, h(x) \in A'] - \rho\rho' \right| \geq \delta \right] \leq (1/\delta^2)/2^t.$$

Proof of Lemma 21. For every $i, k \in [w]$, let $A_{i,k} = \{x \in \{0, 1\}^t : f[i, x] = k\}$ and let $\rho_{i,k} = |A_{i,k}|/2^t$. Note that for every i, j ,

$$\Pr_{x, x' \leftarrow U_t} [f[f[i, x], x'] = j] = \sum_{k \in [w]} \rho_{i,k} \rho_{k,j}.$$

Thus, for every h such that for every i, k, j , $\Pr_{x \leftarrow U_t} [x \in A_{i,k}, h(x) \in A_{k,j}] - \rho_{i,k} \rho_{k,j} \leq \delta/w$, we have that h is δ -good for f . By Lemma 29 this event occurs with probability $1 - (w/\delta)^2/2^t$ over $h \leftarrow \mathcal{H}$ for each pair of sets, and thus with probability $1 - w^3(w/\delta)^2/2^t = 1 - w^5(1/\delta)^2/2^t$ for every tuple (i, j, k) . ◀

We recall there is a logspace algorithm which tests if a hash function is good, given oracle access to the function we wish to fool. We remark that there is work [17, 6, 20] on testing if an entire PRG is good for a branching program, but we need a much weaker claim.

► **Lemma 30** ([18]). *There is a space $O(t + \log(w/\delta))$ algorithm that, given oracle access to $f : [w] \times \{0, 1\}^t \rightarrow [w]$ and $h \in \mathcal{H}_t$ and $\delta > 0$, tests if h is δ -good for f .*

Proof. The algorithm enumerates over $i, j \in [w]$. For every i, j , the algorithm computes $p_{i,j} = \mathbb{E}_{x, x' \leftarrow U_t} [f[f[i, x], x']]$ (i.e. the correct probability) by enumeration over x, x' in space $O(t + \log w)$. Then it computes $\tilde{p}_{i,j} = \mathbb{E}_{x \leftarrow U_t} [f[f[i, x], h(x)]]$ and rejects if the estimate is greater than δ from the true value. Correctness and total space consumption are immediate. ◀

We can then prove that we can test if a hash function is good, given \overline{B} and pointers to preceding hash functions.

► **Proposition 25.** *There is a space $O(t + \log(w/\delta))$ algorithm that, given $n, w, t \in \mathbb{N}$ with $w \geq n$ and $\tilde{h} \in \{0, 1\}^{2t}$ and $\overline{B} : [w] \times \{0, 1\}^n \rightarrow [w]$ and (read only) \mathbf{w} and pointers p_1, \dots, p_r such that $\mathbf{w}_{[p_i, \dots, p_i+2t]} = h_i$ represents a hash function on t bits, returns if \tilde{h} is δ -good for $B_{t, (h_1, \dots, h_r)}$.*

Proof. By Lemma 30, it suffices to show that given $i \in [w]$ and $x \in \{0, 1\}^t$, we can compute $B_{t, (h_1, \dots, h_r)}[i, x]$. To do this, the algorithm maintains $v \in [w]$ as its current position in the branching program (initialized to $v = i$) and $i \in [n]$ to track the current layer. To determine the next position, it suffices to determine the i th block of the output of $\text{NIS}_{(h_1, \dots, h_r)}(x)$. It is well known that this can be computed in space $O(t + \log nw)$ given read-only access to the set of hash functions (by walking down the binary expansion of i , denoted $\langle i \rangle$, and applying h_j if $\langle i \rangle_j = 1$), which we have via the pointers. ◀

4:20 Derandomizing Logspace with a Small Shared Hard Drive

Finally, we provide a translation of our quantization statement. We first recall a strict specialization of the statement of [19]:

► **Lemma 31.** *There exists a **canonicalizer** algorithm \mathcal{C}_t that, given $n, w \in \mathbb{N}$ with $w \geq n$, takes in $\varepsilon > 0$ and a sub-stochastic matrix $M \in \mathbb{R}^{w \times w}$ with each entry represented by at most $O(\log w/\varepsilon)$ bits, runs in space $O(\log w/\varepsilon)$, and returns a branching program \overline{B} of length n and width $w + 1$ with alphabet $\{0, 1\}^t$ for $t = O(\log(w/\varepsilon))$. Moreover, letting $\widetilde{M} \in [0, 1]^{w \times w}$ be the matrix where for $i, j \in [w]$ we have*

$$\widetilde{M}_{i,j} = \Pr_{x \leftarrow U_{\{0,1\}^t}{}^n} [\overline{B}[i, x] = j]$$

then

$$\left\| \widetilde{M} - M^n \right\|_1 \leq \varepsilon.$$

We reduce the alphabet (and slightly increase the length) as follows. We transform \overline{B} into a branching program of width $(w + 1) \cdot 2^t = \text{poly}(w/\varepsilon)$ and length $n \cdot t = n \cdot O(\log w/\varepsilon)$, where each set of t layers reads t bits, interprets these bits as $\sigma \in \{0, 1\}^t$, and takes the transition labeled with σ in \overline{B} .

Explicit Time and Space Efficient Encoders Exist Only with Random Access

Joshua Cook  

Department of Computer Science, University of Texas at Austin, TX, USA

Dana Moshkovitz  

Department of Computer Science, University of Texas at Austin, TX, USA

Abstract

We give the first *explicit* constant rate, constant relative distance, linear codes with an encoder that runs in time $n^{1+o(1)}$ and space $\text{polylog}(n)$ provided random access to the message. Prior to this work, the only such codes were *non-explicit*, for instance repeat accumulate codes [19] and the codes described in [26]. To construct our codes, we also give explicit, efficiently invertible, lossless condensers with constant entropy gap and polylogarithmic seed length.

In contrast to encoders with random access to the message, we show that encoders with sequential access to the message can not run in almost linear time and polylogarithmic space. Our notion of sequential access is much stronger than streaming access.

2012 ACM Subject Classification Theory of computation \rightarrow Error-correcting codes; Theory of computation \rightarrow Expander graphs and randomness extractors; Theory of computation \rightarrow Streaming models; Theory of computation \rightarrow Lower bounds and information complexity

Keywords and phrases Time-Space Trade Offs, Error Correcting Codes, Encoders, Explicit Constructions, Streaming Lower Bounds, Sequential Access, Time-Space Lower Bounds, Lossless Condensers, Invertible Condensers, Condensers

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.5

Related Version *Full Version:* <https://ecc.weizmann.ac.il/report/2024/032/>

Funding This material is based upon work supported by the National Science Foundation under grant number 2200956.

Acknowledgements Thanks to Ryan Williams for questions that eventually led to this result. Thanks to David Zuckerman, Justin Oh, and Jesse Goodman for some advice about which extractors might be used in our condenser construction. Thanks to Niels Kornerup for conversations about time space lower bounds.

1 Introduction

In this paper, we study the time and space efficiency of encoders for error correcting codes. An error correcting code is a function that maps any two distinct messages to codewords that are very far apart in hamming distance. Error correcting codes [42, 27] have numerous practical and theoretical applications. Because of these applications, both codes with efficient encoders and lower bounds for encoders are useful. Codes can also be used as hard functions in lower bounds.

The efficiency of encoding codes has been extensively studied. Some notions of encoding complexity are the size and depth a circuit requires to encode the code. Spielman gave explicit codes that could be encoded by linear sized circuits with logarithmic depth and fan in 2 [49]. For unbounded fan in circuits, Gál, Hansen, Koucký, Pudlák, and Viola gave tight bounds on the size and depth required to encode codes [26]. For depth 2 parity circuits, Gál et al. showed that the minimum circuit size required to encode a constant relative distance code was $\Theta(n(\log(n)/\log(\log(n)))^2)$.



© Joshua Cook and Dana Moshkovitz;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 5; pp. 5:1–5:54

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



In this paper, the notion of encoding complexity we focus on is the time and space required to encode. We investigate the time and space with two different models of how the message is accessed. One is the RAM model where any bit of the message can be accessed in one time step. The other is a sequential model, where the message can only be accessed through heads that can only move one space at a time.

Efficient Encoders with Random Access

Branching programs are the nonuniform version of RAM algorithms. Many problems, such as sorting and finding unique elements, have time space lower bounds for branching programs, e.g. [11, 52, 5, 1, 7, 6].

There are time and space lower bounds for encoding codes too. Bazzi and Mitter proved that for any code with constant relative distance, any branching programs encoding them in linear time require linear space [4], but they don't give any lower bounds on space if the encoder time is $\Omega(n \log(n))$. Self dual codes require branching programs that run in time T and space S to have $ST = \Omega(n^2)$ [45]. Some of the best lower bounds for nondeterministic branching programs are for recognizing codewords. For any good enough code, L , we have that any nondeterministic branching program recognising L in linear time requires linear space [32].

Many codes can be computed either in almost linear time, or polylogarithmic space, but not both simultaneously. For example, Reed-Solomon codes have encoders that run in almost linear time with almost linear space, and encoders that run in polynomial time and polylogarithmic space, but Reed-Solomon codes do not have known encoders that run in almost linear time and polylogarithmic space. In fact, every constant rate linear code has both a quadratic time, log space branching program and a linear time, linear space branching program. However, not all linear codes have branching programs that are polylogarithmic space and almost linear time.

Repeat-Accumulate (RA) codes [19] and the depth 2 circuits of [26] are both *non*-explicit codes that have branching programs that run in quasilinear time and logarithmic space. The best explicit RA codes only have relative distance $O(\frac{\log(N)}{N})$ [24]. The authors of [26] could only partially derandomize their construction. Prior to this work, no *explicit* codes were known that can be encoded simultaneously in almost linear time and polylogarithmic space.

Efficient Encoders with Sequential Access

Branching programs are a model with *random* access to the input. But some hardware accesses data sequentially, and some algorithms output data sequentially. Our algorithms with sequential access to the message will have a restricted number of heads, h , to access the message with. These heads can only be moved one space per time step, or jumped to the location of any other head. Even though the algorithm only has sequential access to the input, it has random access to its smaller working space.

Sequential access arises naturally when composing bounded space algorithms. The standard way to run one algorithm, A , on the output of another algorithm, B , space efficiently is by running A until it wants an input bit, then running B until it outputs that bit. This is a very sequential way to access the output of B . The obvious way to make this faster without storing the whole output of B is to keep intermediate states of B and only simulate B starting from the most recent intermediate state. This is like storing multiple heads to the output of B . Copying one intermediate state over another is like jumping one head to another.

We emphasize that sequential access is much stronger than streaming access as the program can choose which head moves and can read the same message many times. It is also stronger than standard Turing Machine access as there are multiple heads and there are head to head jumps. Other researchers have also studied models of computation with head to head jumps [46, 36, 40].

1.1 Main Results

We give the first explicit code with constant relative distance, constant rate and an almost linear time, polylogarithmic space RAM algorithm encoder.

► **Theorem 1** (Explicit Almost Linear Time, Polylog Space Encodable Codes). *For any $\epsilon > 0$, and N , there exists a linear code*

$$C : \{0, 1\}^N \rightarrow \Sigma^M$$

that has relative distance $1 - \epsilon$, output length $M = O(N)$ and alphabet $\Sigma = \{0, 1\}^{\text{poly}(1/\epsilon)}$. Further C is computable in time $N \text{poly}(2^{\log(\log(N))^3}/\epsilon) + 2^{\text{poly}(1/\epsilon)}$ and space $O(\log(N)^2 + \log(N) \text{poly}(1/\epsilon))$ with random access to the message.

For constant ϵ , we have constant alphabet size, $\Sigma = \{0, 1\}^{O(1)}$, and further C is computable in time $N^{1+o(1)}$ and space $O(\log(N)^2)$.

While one might want an actually linear time, log space encoder, Bazzi and Mitter [4] established that any linear time encoder requires linear space. So if one requires polylogarithmic space encoders, they can't run in linear time.

The distance of a linear code is the Hamming weight of its smallest non-zero codeword. Hence, the idea of our code is to take any non-zero message and condense its weight in a codeword whose length is proportional to the weight. To do that, we use a function called a *lossless condenser*, so we call our codes “condenser codes”. A lossless condenser is a function that takes an input source with entropy and a uniform seed and outputs a source that is smaller than the input source, but still has the same entropy as the input source plus the seed. The idea of condenser codes is to treat the input bits that are 1 as a source of entropy. If this source can be losslessly condensed to an output with constant entropy gap, then this can be used to give a linear function whose output has constant Hamming weight. See Section 2.3 for details.

A condenser can be thought of as a bipartite expander graph. Expander graphs have been used in constructing codes before, for instance for distance amplification [2] or Spielmann codes [49]. But both of these construct codes iteratively in a way that makes it difficult to encode both time and space efficiently. In contrast, condensers give something closer to a one-shot construction that makes it easier to encode time and space efficiently. Expanders have also been used to define codes using local constraints, such as with LDPC codes [20, 38, 47], or even the c^3 LTC codes [18], but these codes do not have known efficient encoders [17].

Crucially, the encoder given in Theorem 1 assumes random access to the message it encodes. In contrast, we show that if the encoder only has sequential access to the message, it cannot encode in almost linear time and sub-linear space:

► **Theorem 2** (Lower Bounds For Encoders With Sequential Access). *Suppose C is a code with relative distance δ encoding N bits. Suppose A is an algorithm computing C running in time T space S and using h sequential heads to access the message. Further assume $S > h \log(N)$. Then*

$$hST = \Omega(\delta N^2).$$

Our lower bounds on the encoders with sequential access to the message are much stronger than those by Bazzi and Mitter [4] on encoders with random access to the message. We give lower bounds on the space of an encoder with sequential access to the message, as long as it runs in time $\ll N^2$. Since in the branching model of computation there are codes with time $O(N \log(N))$ and space $\log(N)$ encoders, this shows that random access to the message is important for time and space efficiently computing a code.

A work by Bangalore, Bhaduria, Hazay and Venkatasubramanian [3] gives a time space lower bound for streaming algorithms encoding codes, but their result is much more restrictive. Their result is in the streaming setting, which only allows one head at a time, and only allows that head to move forward. In contrast, our lower bounds holds for many simultaneous heads on the input where heads can move backward or even jump to other heads.

The sequential model of computation arises naturally when composing low space algorithms. In this scenario, $h = O(S)$, which gives a lower bound of $S^2 T = \Omega(N^2)$.

Finally we show that our lower bound for encoding codes using sequential access to the message, Theorem 2, is tight up to low order factors by giving an explicit code that nearly achieves the lower bound. This code is based on a tensor code using the code in Theorem 1.

► **Theorem 3** (Encoders With Sequential Access Meeting the Lower Bounds). *For any number of heads $h \geq 2$, time $T \geq N$, space $S = \Omega(h \log(N))$, relative distance $\delta > 0$ with $hST = \Omega(\delta N^2)$, there exists a code with constant rate and relative distance $\Omega(\delta)$ encoded by a time $TN^{o(1)}$, space $S \text{ polylog}(N)$ algorithm using h sequential heads to access the message.*

► **Remark 4** (Uniformity of Results For Sequential Access). We note that our lower bounds on sequential access hold for arbitrary operations on the working memory. That is, Theorem 2 even holds for nonuniform algorithms. But our encoders matching the lower bounds are uniform and only needs random access to working memory. That is, Theorem 3, provides a uniform algorithm.

1.2 Invertible Condensers

To construct the explicit codes for encoders with random access to the input, we use condensers. Our condensers need to be efficiently invertible. That is, given an output of the condenser, we need to efficiently iterate through all inputs that map to that output efficiently. To simplify this, we ask that our condenser output 2 outputs, the condensed output and a buffer so that the resulting function is invertible as a function. Additionally, our condenser needs to be good in several other ways. It needs to be lossless, have constant entropy gap, and have polylogarithmic seed size. See Section 2.2.1 for an informal definition of lossless condensers or Section 5 for a formal definition.

► **Theorem 5** (Good Invertible Condensers Exist). *For every n, k and ϵ such that $\epsilon > 2^{3 - n/\log^*(n)^{\log^*(n)}}$, there is a time $\text{poly}(n)$, space $O(n^2)$ invertible, lossless $(n, k) \rightarrow_\epsilon (m, k+d)$ condenser*

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with $d = O(\log(n/\epsilon)^3)$ and $m = k + d + O(\log(1/\epsilon))$.

The efficiency of inverting lossless condensers is not commonly studied. However, efficiently invertible extractors have been used in wiretap protocols [14] and non-malleable extractors [15, 13, 37]. Lossless condensers and extractors are closely related: they are both special cases of condensers.

Some prior lossless condensers are invertible, such as multiplicity code based condensers [33], but they don't have constant entropy gap and small seed. Prior constant entropy loss extractors [50, 25] can be used to build lossless condensers with small seed and small entropy gap, but are not known to be invertible.

1.3 Barrier To Time-Space Efficient PCP for Delegated Computation

In delegation [21, 10, 8, 44, 30] a computation should be carried out so it can be *verified* in short time. Specifically, the paper [8] poses the following challenge. Given:

- A time- T and space- S computation, specified by a transition function that maps time- t states to time $t + 1$ states for every $1 \leq t < T$;
- An input of size $n \leq S \leq T$;

produce in time $T \text{polylog } T$ and space $S \text{polylog } T$ both the outcome of the computation and a certificate that allows probabilistic verification in time $n \text{polylog } T$. The paper [8] calls such a transformation a “complexity-preserving PCPs”.

Theorem 2 gives a barrier towards the construction of complexity-preserving PCPs. It shows that it cannot simultaneously be that:

1. The computation is only accessed by simulating the transition function sequentially.
2. The certificate is an encoding of the computation by a good error correcting code.

Recall that known PCPs are (or can easily be modified to) encodings of the witnesses via (highly specialized) error correcting codes. Moreover, known delegation schemes only use blackbox access to the transition function. Therefore, Theorem 2 implies that a complexity-preserving PCP, if possible, would require significantly different delegation protocols than those proposed so far.

1.4 Time and Space Efficient Decoding?!

Other common notions of a code's complexity are the time required to encode and the time required to decode. Spielman codes [49] not only have linear time encoders, but also have linear time decoders. But both the encoder and decoder require linear space.

Repeat accumulate codes, the codes of [26], and ours all make non-adaptive, also known as input oblivious, queries. That is, on any message, they always query the same message bits in the same order at the same time. While one can time and space efficiently deterministically *encode* codes with non-adaptive queries, one can not efficiently *decode* with deterministic, non-adaptive decoders, even with non-uniform branching programs. Gronemeier [22] proved that any decoders which are deterministic and non-adaptive which run in time $O(N^{1+\alpha})$ must use space $\Omega(N^{1-\alpha})$. Thus one can not hope for decoders to be as efficient as encoders without being adaptive, or randomized.

There are randomized decoders that run in $N^{o(1)}$ space and almost linear time with random access to the message. Specifically, good locally decodable codes have this property [35, 34]. A follow up work by the same authors [16] shows that all locally correctable codes also have *adaptive* time and space efficient decoders. However, these codes have no known time and space efficient encoders.

1.5 On Sequential Access to The Input

The model of computation with sequential access to the input is less studied than branching programs. So we will formally define sequential access and briefly discuss some of its properties.

► **Definition 6** (Sequential Oracle). *Let x be some specific string of length n and h be an integer. Then a sequential oracle to x with h heads is a machine that has as state h integers within the range $[n]$. It has an input tape that can take up to two integers, $u_1, u_2 \in [h]$, indicating up to two of the h heads, and an operation which can be one of the following:*

1. *Move forward. This increments the u_1 th head by one, if it is less than n .*
 2. *Move backward. This decrements the u_1 th head by one, if it is more than 1.*
 3. *Read. Let i be the value of the u_1 th head. Then this returns x_i .*
 4. *Jump to. This sets the u_1 th head to be the same value of the u_2 th head.*
- Any one of these operations can be done in one time step.*

We call the oracle a non-reversible, sequential oracle if it can not use the “move backward” operation.

We call the oracle a non-jumping, sequential oracle if it can not use the “jump to” operation.

We say that an algorithm has sequential access to an input if the only way that input can be accessed is through a sequential oracle.

We assume that all sequential oracles start with all heads at position 1.

There are two potentially contentious operations of this sequential head model: the “move backward” operation, and the “jump to” operation. For simulating algorithms to access their output, the move backward operation may not be possible if the algorithm is not invertible. For hardware, often moving backward may be fine, but jumping heads is not. Our lower bounds, Theorem 2, holds even if heads can both jump and move backward. Our upper bounds, Theorem 3, needs heads that can jump, but not ones that move backward.¹

The “jump to” operation is very powerful. Even with only head to head jumps, non-reversible heads can simulate reversible heads with only logarithmic overhead. Thus the resources needed when given reversible, sequential access is within a log factor of what is needed for non-reversible, sequential access to the input.

► **Lemma 7** (Reversibility Can Be Efficiently Simulated With Jumping). *A single sequential head to a length N input can be simulated with $O(\log(N))$ non-reversible sequential heads to the same input with an expected time of $O(\log(N))$ for each head movement, and $O(\log(N))$ space.*

More generally, k sequential heads to a length N input can be simulated with $O(k \log(N))$ non-reversible sequential heads to that same input with an expected time of $O(\log(N))$ for each head movement, and $O(k \log(N))$ space.

Our sequential access to the input with reversible heads seems very similar to a Turing machine, but the addition of multiple heads makes it much more powerful. For instance, the classic example of a hard problem for a 1 tape Turing machine, the palindrome, takes quadratic time on a 1 tape Turing machine [29]. Just two heads make palindromes easy to solve in linear time. Even if one is only given $O(\log(N))$ non-reversible heads and $O(\log(N))$ space, palindrome can be solved in $O(N \log(N))$ time with non-reversible, sequential access to the input.

There has been research on multi-head Turing Machines. Savitch and Vitányi [46] studied multi-head Turing Machines with heads that can jump to the location of other heads, like our heads do. This model was compared to and contrasted with multi-tape Turing machines in several works [46, 36, 40]. Prior to this work, the only time space lower bounds for sequential access are those implied by the lower bounds for branching programs.

¹ Our codes can also be encoded in the same time and space bound with non-jumping sequential heads if one allows a preprocessing step to move all the heads into an initial position before starting.

For completeness, one might ask if non-reversible sequential access is stronger than non-jumping sequential access and if non-jumping sequential access is stronger than non-jumping, non-reversible access. We show that this is indeed the case in the time and space bounded setting. Informally, we prove that

- Random Access
- > Sequential Access
- \simeq Non-Reversible Sequential Access
- > Non-Jumping Sequential Access
- > Non-Jumping, Non-reversible Sequential Access

That random access is more powerful than sequential access is a direct consequence of our explicit codes in the RAM model, Theorem 1, and our code lower bounds for sequential access to the input, Theorem 2. The other two inequalities come from our code lower bounds for sequential access, Theorem 2, and our explicit codes for sequential access, Theorem 3. See Section 9.3 for details.

2 Technique

For our results, we use the convention that capital letters are an exponential factor larger than their lower case counterparts. For example, $N = 2^n$, $K = 2^k$ and $M = 2^m$. In particular, our codes with efficient encoders with random access to the input are constructed using condensers that act on the indexes of the bits in the code. So our codes are functions on N bits, and our condensers are functions on n bits.

We start with proving our lower and upper bounds for encoders with sequential access to the input. Then we show how to build explicit codes with efficient encoders using random access to the input.

2.1 Sequential Access To The Message

If one only has a bounded number of heads to access the message and they can only move one space (or jump to other heads) in one time step, can we still time and space efficiently encode any code? We show that no codes with good distance can be time and space efficiently computed with only sequential access to the message.

We show that given space S , time T , relative distance δ and max number of heads h such that $hST = o(\delta N^2)$ that any algorithm running in time T and space S limited to at most h sequential heads can not compute a code with relative distance δ . Further when $hST = \Omega(\delta N^2)$, we show that an algorithm with time close to T , space close to S , and h heads computes a code with distance close to δ .

We start by showing the lower bound for non-adaptive sequential access to the input as a warm up. Then we give the lower bound for even adaptive sequential access to the input. Finally we show how to use time and space efficient codes that use random access to their message in order to construct codes with encoders with sequential access to the input that match our lower bounds.

2.1.1 Non-Adaptive Lower Bound

The idea is to group the message bits into intervals larger than S . So for any interval, we think of a two person game, where one player has access to an interval when a head is in it, and gets to communicate about the contents of that interval to a second player whenever all

heads leave that interval. We want to show that the second player can't learn the contents of the interval, thus must do the same thing for two different settings of that interval. The main ideas are that:

1. When a head leaves an interval, it can only communicate S bits. Thus the second player gets very little information about that interval every time it is visited.
2. If every head is far from an interval, it takes a lot of time to move a head back to that interval. So for most intervals, the second player only gets information about that interval few times.
3. If there are few heads, most intervals must have all heads very far from them at any given time. So for most intervals, the second player has to write most output bits.

Thus for most intervals, if the second player does not get as much information about an interval as the interval contains, then for two different messages, the second player must get the same information for both messages. Then the second player must output the same codeword bits for both. If the second player also writes most of the codeword, then most of the codeword will be the same for those two messages. Thus these two messages will have close codewords, so the code will not have good distance.

So we want that most intervals both have most output bits written when no head was near them, and that most intervals only went from having a head in them to having all heads far away a small number of times. Thus all the bits written when all heads were far away from an interval must have the same output for two separate messages.

In particular, we set the number of bits in each interval to be around $I = \frac{\delta N}{8h}$ so that most intervals have at least one interval between them and any head at any point in time. Since it takes at least I time steps to move a head into an interval that is distance I from every head, the number of times an interval transitions from being far from any heads to having a head on it is at most $\frac{T}{I}$.

For non-adaptive encoders, the argument follows fairly directly. An interval must have been visited at least $\frac{I}{S}$ times for player 2 to know enough about that interval to write different things for every possible contents of that interval. So only $\frac{ST}{I^2}$ many intervals can be visited enough to have any distance when a head isn't near them. Only $\frac{3h}{\delta}$ of the intervals can have a head near it when more than δ fraction of output bits are written. Thus a constant fraction of intervals, $\frac{N}{I} - \frac{3h}{\delta} = \frac{5h}{\delta} = \frac{5N}{8I}$, has at least a δ fraction of bits written when no head is near them.

Thus to have distance δ , we need the number of intervals visited often enough, $\frac{ST}{I^2}$, to be at least the number of intervals that often don't have a head near them, $\frac{5N}{8I}$. Otherwise, for one of the rarely visited intervals, for two different messages, the encoder must write the same thing when all heads are far from that interval. Thus the code wouldn't have good distance. Thus

$$\begin{aligned} \frac{ST}{I^2} &\geq \frac{5N}{8I} \\ ST &\geq \frac{5}{8}NI \\ ST &\geq \frac{5}{8}N\frac{\delta N}{8h} \\ hST &\geq \frac{5}{64}\delta N^2 \end{aligned}$$

2.1.2 Adaptive Lower Bound

Adaptive lower bounds are trickier since the queries can change for different messages. As an example, our prior lower bound even works for messages where the entire message is zero, except for one interval. A non-adaptive algorithm can not even encode this extremely restricted message into codewords with large distance. But a non-adaptive algorithm that knows we will use this kind of counterexample will search for such an interval, then only encode that interval. This cuts down the number of bits the adaptive algorithm will have to encode by a factor of $\frac{\delta}{sh}$ which allows our adaptive encoder to outperform a non-adaptive encoder on these particular kinds of counterexamples.

The issue in this counterexample is that the encoder gets to know everything outside a target interval for free. So instead of always choosing the message outside an interval to be zero, our counterexample will fix it in some way that will depend on the encoder, so that way the algorithm can not know it ahead of time. So the idea essentially is to choose a random interval and a random restriction to everything outside the interval and then try to use a similar argument.

More specifically, we use a proof by contradiction. We say restriction of the message is good if it restricts everything but one interval, and for most assignments to variables in that interval, most bits are written when no head is near the interval, and heads don't enter the interval many times. If this holds for most assignments to the interval, the same prior arguments work. But if no good restrictions exist, it must be because on average the intervals are visited too many times for the number of heads and the time of the algorithm. But since we have a bound on the time and number of heads of the algorithm, this can not happen.

2.1.3 Upper Bound

The lower bounds from the previous section is tight since we can construct codes that match them, up to small factors. We construct these codes using a tensor code product of any code that has a space efficient, non-adaptive encoder using random access to the message and any other time efficient code. There is a straightforward way to time and space efficiently compute the tensor code of any space efficient code with another code. As a reminder, a tensor code arranges the message into a table, then one code encodes each row to construct an intermediate table. Then the other code encodes each column of the intermediate table to get the final result.

Specifically, for target space S , we arrange the code into S columns, each of size about $\frac{N}{S}$. Then each row is encoded in any arbitrary time efficient linear code. And each column is encoded in our time and space efficient code that uses random access to the message.

The time efficient way to encode the tensor code is to literally construct the intermediate table by encoding every row, then encode every column of the intermediate table. But storing this intermediate table is not space efficient. If the code encoding the columns is space efficient and non-adaptive, there is an alternate, space efficient way to do this. This is to simulate the encoder for all the columns in parallel, and every time it needs to query one of the rows, we encode this row and then give the result to the space efficient code.

So our encoder places each of the h heads uniformly across the rows of the message so that any row is within $O(\frac{N}{h})$ of a head. As long as there are fewer heads than rows, since the row encoder is time efficient, moving to the row will take more time than encoding it. Then every query to a row during the column encoding only takes time $O(\frac{N}{h})$. And since the column encoder is time efficient, we only need to query the rows around $\frac{N}{S}$ times. So the total time is around $T \simeq \frac{N^2}{hS}$, thus $hST \simeq N^2$. The space is similar to S since the column encoder is space efficient and only needs to store the state of S columns at once.

Finally, to get the tradeoff with distance, we use the same tensor code. Then we only bother to encode the message in size $\Omega(\delta N)$ intervals and encode each interval independently. If each of these smaller codes have constant relative distance, then these codes together have relative distance $\Omega(\delta)$. Using the prior time space efficient algorithms, for space S and heads h , you can compute a code with constant distance on a size (δN) message in time about $\frac{(\delta N)^2}{hS}$. Doing that $1/\delta$ times only requires time about $\frac{\delta N^2}{hS}$, which is what we want.

2.2 Random Access To The Message

First, we recall some relevant definitions. For any alphabet Σ with a special zero character, and any $x \in \Sigma^N$, we say the weight of x , denoted $weight(x)$, is the number of non-zero symbols in x . Similarly the relative weight of x is $\frac{weight(x)}{N}$. A linear code is any code whose encoding function is a linear function. Then the distance of a linear code is the weight of its smallest non-zero codeword. See Section 5 for more details.

So the goal is to construct some linear function which is fast and space efficient to compute, but has high weight for any non-zero message. Our basic strategy is to construct a different linear function for every possible (order of magnitude of) the message weight. Then we combine these linear functions into one linear function, such that if any linear function outputs a high weight output, the combined linear function will too.

This high level approach is nearly identical to the codes with depth 2 circuits of Gál, Hansen, Koucký, Pudlák, and Viola [26]. In fact, our code can also be viewed as an almost linear sized, depth 2 circuit. But our codes are explicit, and our results require several new ideas. For a detailed comparison, see Section 3.1.

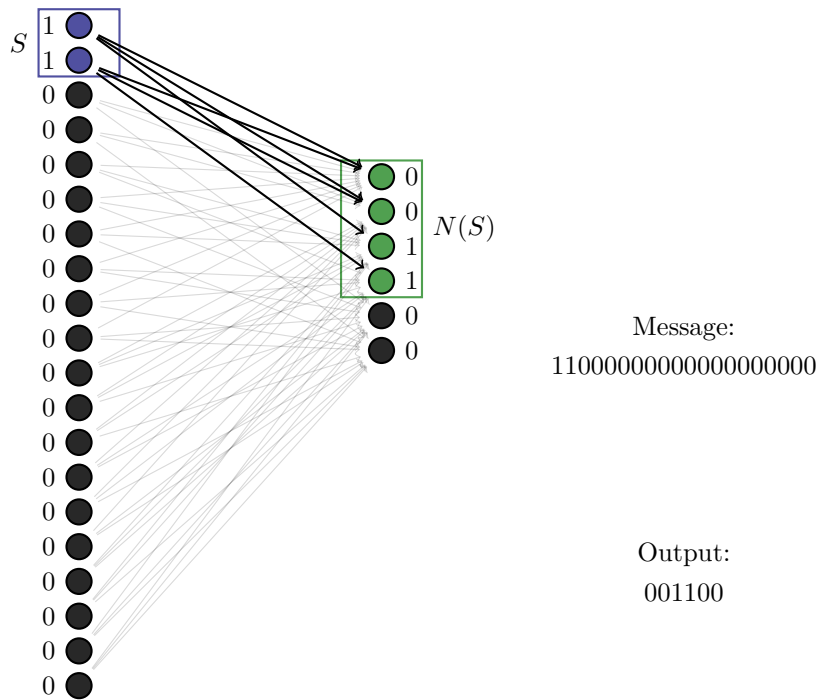
2.2.1 Weight Fixers From Condensers

For any input range $R \subseteq [N]$ and relative weight $\delta > 0$, we call a function $F : \{0, 1\}^N \rightarrow \Sigma^M$ an R to δ weight fixer if for every message x with weight within R , we have that $F(x)$ has relative weight at least δ . We note that R is usually an interval. So our first goal is just to construct $[2^{i-1/2}, 2^{i+1/2}]$ to δ weight fixers for some $\delta > 0$ for every $i \in [\log(N)] = [n]$. For this we use lossless condensers.

One way of looking at lossless condensers is as bipartite graphs [43, 50]. A lossless condenser is a regular graph with left vertices L , right vertices R , with left degree $D_L = 2^d$. We call d the seed length. We call the graph a lossless condenser for min-entropy k if for every $S \subseteq L$ with $|S| = 2^k$, we have that the neighbor set of S , denoted $N(S)$, has size $|N(S)| \geq |S|D_L(1 - \epsilon)$. We say the condenser has constant entropy gap if $|N(S)| = \Omega(|R|)$, that is, a constant fraction of R has a neighbor in S .

If ϵ is small, then most of elements of $N(S)$ have one unique neighbor in S . Here, we view L as the input bits ($|L| = N$), and R as the output bits ($|R| = M$). If the message, x , has weight 2^k , we can let S be the one bits of the message. If the condenser has constant entropy gap, then a constant fraction of the output bits have a neighboring one bit. In fact a constant fraction of the output bits have exactly one neighbor that is a one bit.

Our weight fixer will just xor all an output bit's neighbors to compute its value. Then all of the output bits with a unique one bit neighbor will output one. Since for a weight 2^k message this is a constant fraction of the output bits, this weight fixer will give a constant relative weight output on a weight 2^k input. See Figure 1 for an example of a condenser and its corresponding weight fixer.



■ **Figure 1** Lossless Condenser And $\{1, 2\}$ to $\frac{1}{3}$ Weight Fixer Example.

The lossless condensers we construct gives us, for any i , a $[2^{i-1/2}, 2^{i+1/2}]$ to δ weight fixer $F_i : \{0, 1\}^N \rightarrow \{0, 1\}^M$ where $M = 2^i 2^{O(\log(\log(N))^3)}$. This $2^{O(\log(\log(N))^3)}$ factor is related to the seed length of the condenser. Each bit in the output of F_i can be computed in time about $N/2^i$ and space $O(\log(N)^2)$, so the total time to compute the entire output of an F_i is $N 2^{O(\log(\log(N))^3)} = N^{1+o(1)}$. See Section 2.3 for more details.

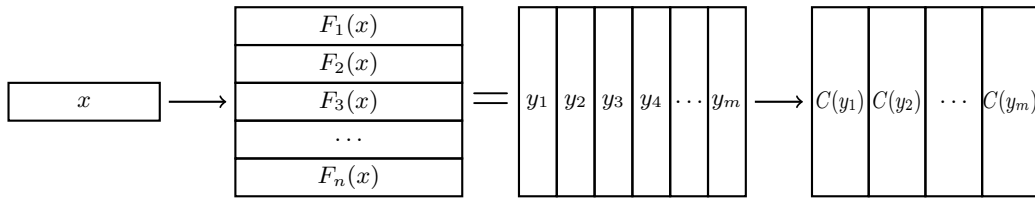
2.2.2 Mixing Weight Fixers

Now we can perform a tensor like operation to mix all these weight fixers into one code. The idea is that a code always maps non-zero inputs into a string with high weight. So if we can partition the output of our weight fixers into small clusters so that most clusters are non zero, applying the code to each cluster gives a large weight output.

For any input weight message, some weight fixer F_i will have large weight. So all we need to do is make sure each cluster contains a good sample of the output of each F_i . This is easy, just have each cluster contain a single output symbol from each F_i in any way as long as each F_i has each of its outputs in the same number of clusters. To do this time and space efficiently, we just use the first symbol from F_i as many times as we need to, than the second symbol and so on.

We can visualize this mixing technique with a table. First put the output of each F_i into a row in a table. Repeat the symbols in each row until every row is the same length. Then encode the columns of that table with any asymptotically good code. The resulting weight will be at least the minimum weight of any row times the distance of the code. See Figure 2 for a diagram.

To run this mixer time and space efficiently, we encode one column at a time, and store the current symbol for each of the $n = \log(N)$ weight fixers. Then to compute the next column, we only need to compute weight fixers who have changed their value since the last



■ **Figure 2** Mixing Weight Fixers F_1, F_2, \dots, F_n with code C .

column. For most columns steps, most weight fixers won't use a new symbol because most weight fixers have their symbols repeated many times. This only requires $O(\log(N))$ space for the current output bit of each F_i , plus the space to compute any single F_i . It only requires the time to compute each F_i and to run C .

One issue with the approach we have described thus far is that the output length will be at least the output length of the largest F_i , which is $N2^{O(\log(\log(N))^3)}$. Thus the output length of the code we have just described is $N2^{O(\log(\log(N))^3)}$, but we need an output length of $O(N)$ to get constant rate. The weight fixers for larger message weights are the issue. So instead of using our condenser based weight fixers for every message weight, we only use them for small weight messages. We use a different weight fixer for messages with large weights and mix the two results.

2.2.3 Weight Fixers For Large Weight Messages

For large weight messages, we use a code extremely similar to Spielman codes [49]. Spielman codes use a recursive approach, where every level of recursion is on a smaller input and increases the distance. We use the same codes, except that instead of recursing all the way down to a code on a constant sized message, we stop early with a weight fixer that is just a bit smaller than N . We call these Spielman style weight fixers.

Spielman style weight fixers always have constant rate, but the weight of messages they can fix increases with each level of recursion. One can compute the output of Spielman style weight fixers space efficiently, but the time increases exponentially in the number of recursions. Thus by choosing an appropriate recursion depth, we can balance the time used by the Spielman style weight fixers with the rate of the condenser style weight fixers.

The Spielman style weight fixer uses a lossless condenser to get a function, $A : \{0, 1\}^N \rightarrow \{0, 1\}^{N/2}$, such that for any input with weight less than some constant α , A gives an output with the same weight as the input. For the lossless condensers in the Spielman style weight fixers, we use the condensers from [12]. Then the idea is to repeatedly apply A in several levels until you have constant relative weight, then mix all of these levels.

If one stops the recursion early, we will not have had enough rounds to concentrate the small weight messages to be a constant fraction of the bits at any level. But each round will still let you fix weights a constant fraction smaller. The output bits at any given level are just some particular xor of message bits, but the number of bits you xor increases exponentially with the depth, which is why this algorithm has time exponential in its depth.

Now choosing appropriate number of recursions for the Spielman style weight fixers gives fixers for the very heavy messages. Light messages are handled by our weight fixers based on condensers. But this only gives us codes with *some* constant distance, and we would like codes that have arbitrarily large constant distance.

2.2.4 Distance Amplification

A first try would be to use an off the shelf distance amplification procedure, like that of Alon, Bruck, Naor, Naor and Roth (ABNNR)[2]. While we use the ideas of ABNNR, we can not use it directly as it does not preserve the space and time complexity of the encoder. First we explain ABNNR, and its limitations. Then we explain how to modify it to work for us.

The idea of ABNNR is to take a base code and a bipartite expander graph, and identify the left hand side with the message bits, and the right hand side with the output bits. Then the output of a right hand side vertex is just the concatenation of all its adjacent message bits. This increases the alphabet, and if the graph is an expander, increases the weight.

Now if we just apply ABNNR directly to our final code, it is unclear how to encode the resulting code efficiently. There is a straightforward, time efficient way to encode the code: just compute the whole left side code, than concatenate the symbols on the right hand side. But if one wants to do this space efficiently, then we do the following instead: for each right hand side vertex, we compute all the left hand side bits incident to it and append them together. The issue is that each of the left hand vertices are a function of a constant fraction of the message bits, thus take time $O(n)$ to compute. Spending $O(n)$ time to compute each of the $O(n)$ output vertices takes quadratic time, which is too much.

Instead, we need to run distance amplification on each of the $O(\log(N))$ weight fixers before we combine them. Then when the bits of the weight fixer are expensive to compute, there are fewer output bits, so the distance amplification only increases encoder time by a constant factor. Then our mixer preserves this distance, as long as the code in the mixer also has good distance. This allows us to get arbitrary good weight.

2.3 Invertible Condensers

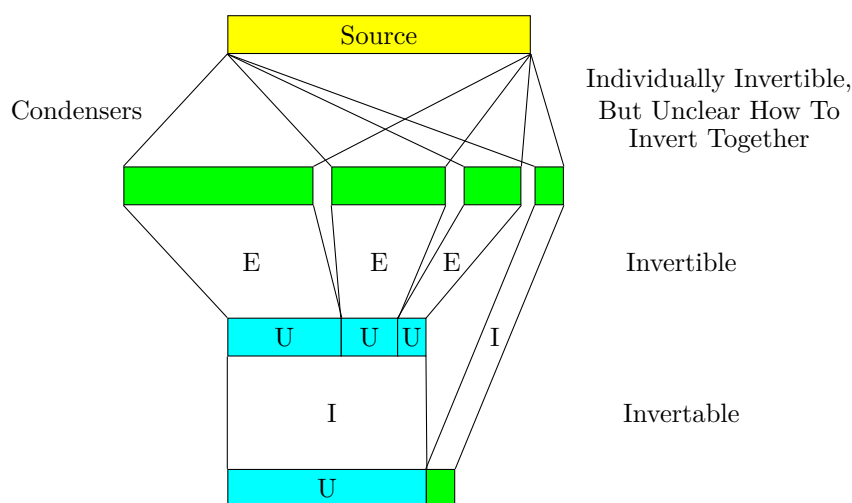
Now we discuss how to modify existing condenser and extractor constructions to get invertible condensers. To understand these condenser constructions, we need to explain an alternative, equivalent definition of condensers. Instead of thinking of condensers as bipartite graphs, one can also think of them as functions that take low entropy sources over long bit strings to sources with a similar entropy over short bit strings.

Explicitly, a k entropy lossless condenser is a function $C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ such that for any random variable $x \in \{0, 1\}^n$ with min entropy k and $U_d \in \{0, 1\}^d$ an independent, uniform distribution, we have that $C(x, U_d)$ is close to a distribution with min entropy $k + d$. We call $m - (k + d)$ the entropy gap since it is the difference between the amount of entropy that could be in an output with m bits and how much entropy is in that output. The entropy gap of the condenser is related to the weight of the related weight fixer. We want constant entropy gap.

The state of the art condensers based on Pavarash-Vardy codes and Multiplicity codes [25, 33] are lossless, but require large seeds to get small entropy gaps. One can think of extractors as a kind of condenser with no entropy gap at all. We know explicit extractors with small seeds [51]. Unfortunately extractors must have entropy loss, and we need lossless condensers.

One might hope for a way to combine lossless condensers and extractors to get a the benefits of both: small entropy loss for extractors, and small entropy gap for condensers. This is possible using the condense and extract framework [43, 50]. Next we describe the condense and extract framework.

In the condense and extract framework, one first condenses the message to concentrate the entropy, then extracts much of it. Then there is still some remaining entropy in the message, conditioned on the output of the extractor, so we condense it again to a much



■ **Figure 3** Standard Condense And Extract. Source is condensed from many times in parallel, and the results are extracted in parallel. Legend: E, Extractor. I, identity map. U, Uniform Bits.

smaller message, which we can then more efficiently extract from. Then we repeat until almost all the entropy has been extracted. Then to convert the final result to a lossless condenser, we condense the remaining entropy one final time without extracting it.

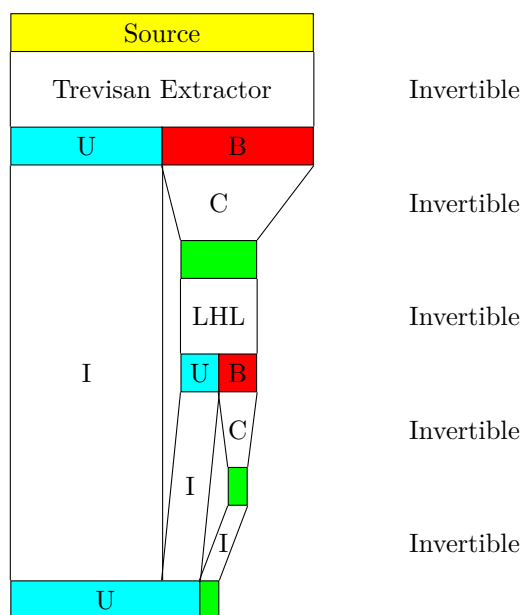
Things get a bit more complicated when one requires invertibility. First, we require our component condensers and extractors to be invertible, which is doable. But then we run into an issue of extracting and condensing from the message many times in parallel. Now even if we can invert each of the condensers and extractors by themselves efficiently, it remains difficult to space and time efficiently determine which messages can give the expected output of each extractor and condenser simultaneously.

This is solved by changing our extractors into buffered extractors, like those of [12], and then condensing from that buffer. A buffered extractor is just an extractor with a second output, called a buffer, which contains all the entropy the first output missed. With this change, inversion is straightforward. For the same reason, the extractors and condensers of [12] are also efficiently invertible.

Now it only remains to choose appropriate invertible extractors and condensers to compose to get our final condenser. For condenser we choose the multiplicity code based condensers as these are both efficient and are easy to invert. For extractors, we use the Trevisan extractor [51, 41] to extract most of the entropy, and then a left-over hash lemma based extractor to get the rest. So the final condenser runs the Trevisan extractor, condenses, runs the left-over hash lemma extractor, and then condenses again.

The Trevisan extractor is efficiently invertible because it is a linear function conditioned on the seed, thus can be inverted by Gaussian elimination. This requires quadratic space and polynomial time (in $n = \log(N)$). One subtle issue is that we define invertibility of an extractor as a literal function inversion of an extractor along with a buffer. But the Trevisan extractor with a fixed seed is not always full rank, thus is not an invertible function. To handle this, our extractor detects such bad seeds (which don't extract well anyway) and just use any arbitrary invertible function with them.

This Trevisan extractor is the main limitation in our encoders time and space. Getting an extractor with shorter seed length will improve the time of the encoder, and a more space efficient inversion process would improve the space of the encoder.



■ **Figure 4** Our Condenser. It has two extractors, a Trevisan extractor and a Left-over Hash Lemma (LHL) based extractor. Both extractors output some uniform bits and a buffer with the left over entropy. Before the buffer can be extracted from efficiently, it needs to be condensed first. Legend: C, Condense. I, identity map. U, Uniform Bits. B, Buffer.

3 Comparison With Other Codes

3.1 Comparisons With Codes For Shallow Circuits

While we investigate time and space efficiency of encoding, other works have investigated the circuit complexity of encoding codes. For instance, while Spielman codes [49] are often cited as linear time encodable, they are also encodable by a uniform, fan-in 2, log depth circuit with linear size. A later work by Gál, Hansen, Koucký, Pudlák, and Viola [26] considered unbounded fan-in circuits with arbitrary gates. For any depth, Gál et al. gave tight bounds on the size of a circuit required to encode a code with constant relative distance.

Gál et al. show that encoding any asymptotically good code with depth 2 circuits requires $\Omega\left(n\left(\frac{\log(n)}{\log(\log(n))}\right)^2\right)$ wires and depth 3 circuits requires $\Omega(n \log(\log(n)))$ wires. As depth increases further, the number of wires required sharply decreases, with linear sized circuits at depth $\log^*(n)$. We emphasize that [26] give both lower bounds and matching upper bounds. However, their codes are non-explicit.

Our code constructions and the depth 2 circuits of [26] are, conceptually, extremely similar. Their circuit constructions use what they call “range detectors” which are equivalent to weight fixers. Its depth 2 circuits do exactly what we do: make weight fixers for each order of magnitude, then mix them. Our encoders could also be stated as uniform, almost linear sized, depth 2 circuits.

Derandomization of the codes of [26] was left as an open problem. They could only achieve partial derandomization. Their partial derandomization is a variation of our mixer (compare Theorem 19 with [26, Claim 37]), but they had no explicit constructions for weight fixers. Our weight fixers can actually be expressed as parity gates, and our encoders can be described as explicit depth 2 circuits of almost linear size. Thus we solve the open problem in [26] of finding explicit codes with depth 2 circuits of almost linear size.

Gál et al. never analyzed the time and space required to encode their codes. It is not obvious that the codes of [26] should be encodable in almost linear time and logarithmic space. While it is true that any constant depth, almost linear sized circuit can be evaluated in either almost linear time and almost linear space, or logarithmic space and polynomial time, they can't always be computed in almost linear time and logarithmic space simultaneously. The fully randomized codes with depth 2 encoders from [26] do not seem to have almost linear time and log space encoders, only their partially derandomized codes do.

The main differences between their construction and ours come from the fact that ours are explicit. We use condensers to make our weight fixers explicit, and the best known condensers cannot make weight fixers that are as good as the randomized construction of Gál et al. So we need several new ideas to get our codes. A few difficulties and solutions include:

1. For weight K inputs, our lossless condensers give weight fixers with output length $\Omega(K^{2^{\text{poly}(\log(\log(N)))})$. When K is close to N , this is larger than N . If we used these weight fixers for large weight messages, the output would have super linear size, so our code would not be constant rate. So we need to construct weight fixers for large input weight messages in a different way (through Spielman style weight fixers).

The weight fixers in [26] for weight K inputs have output length $O(\log(\frac{N}{K}))$. So their weight fixers always have less than linear output length.

2. Condensers cannot give us distance $1 - \epsilon$ for small constant ϵ . To improve our distance, we need to use the distance amplification of ABNNR [2], but in a special way. ABNNR does not seem to simultaneously preserve time and space efficiency of encoding: it can do one or the other. So we have to apply it to our weight fixers *before* they are mixed.

Gál et al. only gives codes with relative distance δ for some constant $\delta > 0$, it does not try to give large relative distance. However, if their randomized construction is modified to give weight fixers with multiple output bits, then the same approach gives codes with distance $1 - \epsilon$, codeword length $O(\frac{N}{\epsilon^2})$, alphabet $\{0, 1\}^{O(\log(1/\epsilon))}$ and encoders running in time $N \text{poly}(\log(N)/\epsilon)$ and space $O(\log(N) \log(1/\epsilon))$.

Thus while the high level code construction of [26] is similar to ours, we need several new ideas to make it explicit, keep the rate constant, and the distance close to one.

3.2 Why Spielman Codes Aren't Enough

Spielman codes [49] are well known codes that can be encoded in linear time, so it is natural to ask whether they can also be encoded in small space. From Bazzi and Mitter [4], we know that it's linear time encoder cannot be sublinear space. We know of an alternate way to encode Spielman codes in logarithmic space, but this approach requires time $n^{1+\beta}$ for some $\beta > 0$. Standard Spielman codes have $\beta > 1.5$, but even optimizing their parameters one cannot get β approaching zero without the relative distance of the code also approaching zero. Subsequent improvements, like those of Guruswami and Indyk [23], still contain Spielman codes within them, and thus suffer from the same problems with encoding efficiency.

Now we give a brief explanation of this space efficient evaluation of Spielman codes, and why it can't be time efficient. One can view Spielman codes [49] as a parity circuit. Each layer in the parity circuit is given by some family of regular bipartite expanders A_i for $i \in [O(\log(N))]$. For simplicity, you can think of the circuit as identifying the gates of layer i of the circuit with the left vertices of A_i , and the gates in layer $i + 1$ of the circuit with the right vertices of A_i . Then the edges in A_i denote the inputs to the parity gates.

The obvious space efficient way to evaluate such a circuit is to recursively evaluate a gate's value by iterating over each input to that gate. For a depth depth circuit with fan in f with final layer size L , this only takes space $O(\text{depth} \log(f))$, but requires time $f^{\text{depth}} \cdot L$. Then

fan-in, f , is the degree of the expanders, and depth, depth , decreases with the expansion of the expanders. Improving one hurts the other. One can improve the tradeoff by making the expanders more imbalanced, but that hurts the distance of Spielman codes. Our one shot approach allows us to use very imbalanced expanders without hurting our distance.

4 Open Problems

There are many open problems related to time and space efficiency of encoding.

1. Get better condensers to construct codes with more efficient encoders using random access to the input. Our codes need, for every $k \in [n]$, a lossless $(n, k) \rightarrow_\alpha (m, k + d)$ condenser, $C_k : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^{m_k}$, with constant entropy gap ($m_k = k + d + b$ for $b = O(1)$) and approximation error, α , less than one half. Our codes have messages of length $N = 2^n$.
 - Improve the space of our encoder.
If each C_k is invertible in space S , then our code can be encoded in space $S + O(n)$. Our condenser is constructed with a Trevisan style extractor [51, 41], which we only know how to invert by exploiting its linearity, which takes space n^2 . If one uses a more efficiently invertible condenser, that will improve the space required by the encoder. We suspect that other condenser designs, like [25], could be made space $O(n)$ and $\text{poly}(n)$ time invertible, but have not checked.
 - Improve the time of our encoder.
If each C_k is invertible in time T and has seed length d , then the time of our encoder is $O(N \log(N) 2^{dT})$. So if one can get efficiently invertible, lossless condensers which condense all (except $O_\alpha(1)$) bits of entropy with seed length $O(\log(n))$, then there is a code with an encoder that runs in time $N \text{polylog}(N)$.
We note that if one can give an extractor with seed length $O(\log(n))$ which extracts all the entropy (except $O_\alpha(1)$ bits), then we have a lossless condenser with a similar seed length and an encoder that runs in time $N \text{polylog}(N)$. The best known explicit extractors [25] require seed length $O(\log(n)^2)$ to extract all (but $O_\alpha(1)$ bits) of the entropy of a source. This seed length is not short enough to get a quasilinear time, polylog space encoder.
 - Improve the dependence on ϵ .
A simple version of our codes only achieve constant relative distance. To get relative distance $1 - \epsilon$ for any constant ϵ , we use extractors. For the entropy gap $b = O(1)$, for every k , we need an $(m_k - b, \epsilon)$ extractor $E : \{0, 1\}^{m_k} \times \{0, 1\}^{d'} \rightarrow \{0, 1\}^{m'_k}$ to get a code with distance $1 - \epsilon$. If E is time T and space S invertible, then this distance amplification increases the encoding time by a factor of $O(2^{d'} T)$ and increases space by $O(S)$.
For our distance amplification, we use the extractors from [12]. These use some small, non explicit conductors of size $\text{poly}(1/\epsilon)$. Since we only find these using brute force, this adds an extra time of $2^{\text{poly}(1/\epsilon)}$ and an extra space of $\text{poly}(1/\epsilon)$. Using a different extractor that does not require a brute force search could give a better dependence on ϵ .
2. Give a code with an encoder *and* decoder that run in small time and space with random access to the input, or show they can not exist. We know from this work that there exist codes with a time and space efficient encoder, and from follow up work [16] that there are codes with a time and space efficient decoder. Can a single code be both time and space efficient to encode and decode?

3. There are other interesting properties of codes, like

- Local Testability.
- Local Decodability.
- Relaxed Local Decodability.

In particular, let X be the interesting property (such as local test ability or local decodability), then for the different kinds of access, we ask

a. In the sequential access setting, is there a code with X and an encoder running in time T and space S with $h = O(S)$ sequential heads to access the input such that $ST \ll N^2$?

We proved in Theorem 3 that we can do better than this with sequential access to the message for some codes. In particular, if $S = h = \sqrt{N}$, then there is a code that can be encoded in almost linear time. So if a code cannot be encoded time and space efficiently, it must be because of X not just because we require constant relative distance.

b. In the random access setting, is there a code with X that can be encoded in almost linear time and polylogarithmic space?

We know codes with time and space efficient encoders can't have some interesting properties. For example, self-dual codes require encoders running in time T and space S to have $ST = \Omega(N^2)$ [45].

4. Derandomize the Repeat Accumulate codes (RA codes).

Repeat accumulate codes have a simple description: first, take an input, x , and repeat it k times (think of $k = O(\log(N))$) to get y . Then, for some fixed random permutation π (this is why the code is not explicit), permute y by π to get z . Finally, for every $i \in [kN]$, the i th output bit is the xor of bits in z before index i : if the resulting codeword is C , then $C_i = \bigoplus_{j < i} z_j$.

Non-explicit RA codes have faster encoders than condenser codes and are simpler to describe. RA codes run in time $O(N \log(N))$ and use space $O(\log(N))$. Even with optimal condensers, condenser codes *cannot* be made to run in $O(N \log(N))$ time. This is because condenser codes can also be described as depth 2 circuits, and Gal et al [26] proved depth 2 circuits encoding a code require size $\Omega(N \log(N)^{1.999})$. So there may be a simpler, more efficient, explicit code based on RA codes.

The best known derandomization of RA codes [24] only have distance $O(\log(N))$, i.e. relative distance $O(\frac{\log(N)}{N})$. This low distance is inherent to the technique: the distance is the girth of a three regular graph, and all three regular graphs have girth $O(\log(N))$.

5 Preliminaries

In this paper, it will be convenient to use linear codes as it has a simple way to characterize its distance. But we also want larger alphabet sizes as achieving high distance is easier with larger alphabet. So we will define all of our functions as if they are over a binary alphabet, but for distance we will use a larger alphabet. This doesn't change any of the actual codes, but simplifies some of the analysis.

A code is just a function whose outputs differ in most locations. Here is a formal definition of an error correcting code.

► **Definition 8 (Code, Distance, and Rate).** *Let Σ_1 and Σ_2 be any alphabets over binary bits: $\Sigma_1 = \{0, 1\}^a$ and $\Sigma_2 = \{0, 1\}^b$ for some integers a and b . Then for any function $C : \Sigma_1^N \rightarrow \Sigma_2^M$, we say C is a code with relative distance δ if for any two $x, y \in \Sigma_1^N$ we have that $C(x)$ and $C(y)$ differ on at least δ fraction of indexes.*

We say that C has rate $\frac{N}{M}$. We say that an element $u \in \Sigma_2^M$ is a codeword of C if for some $x \in \Sigma_1^N$ we have that $C(x) = u$.

All of our codes will be linear, so we need to define a linear function.

► **Definition 9** (Linear Function). *For any function $L : \{0, 1\}^N \rightarrow \{0, 1\}^M$, we say L is linear if every output bit of L is just a parity of some specific set of input bits.*

Let Σ_1 and Σ_2 be any alphabets over binary bits: $\Sigma_1 = \{0, 1\}^a$ and $\Sigma_2 = \{0, 1\}^b$ for some integers a and b . Then we say that any function $L' : \Sigma_1^N \rightarrow \Sigma_2^M$ is linear if L' viewed as a function on individual bits is linear.

Linear codes, that is codes who are themselves linear functions, have many nice properties. One nice property is that the distance of the code is equal to the weight of its smallest, nonzero output. The weight of a vector is just its number of nonzero elements. Here is a formal definition of the weight of a string.

► **Definition 10** (Weight of a String). *Let Σ be any alphabet over binary bits: $\Sigma = \{0, 1\}^a$ for some integer a . Then for any $x \in \Sigma^N$ for some integer a , we define $weight(x)$ to be the number of symbols in x that are not the all 0 symbol: 0^a .*

The relative weight of x is $\frac{weight(x)}{N}$.

Now we can show a useful characterization of the distance of linear function.

► **Lemma 11** (Distance of a Linear Code). *Let Σ_1 and Σ_2 be any alphabets over binary bits: $\Sigma_1 = \{0, 1\}^a$ and $\Sigma_2 = \{0, 1\}^b$ for some integers a and b . Let $C : \Sigma_1^N \rightarrow \Sigma_2^M$ be a linear function. Then C is a code whose relative distance, δ , is the weight of its smallest non-zero output:*

$$\delta = \min_{x \in \Sigma_1^N, (0^a)^N \neq x} \frac{weight(C(x))}{M}.$$

Proof. See that for any two elements $u, v \in \Sigma_2^M$, for any index $i \in [m]$ we only have $(u - v)_i = 0^b$ if u and v are equal on index i . Thus the distance between two outputs of C is just the weight of their difference. Thus the distance of the C can be written as

$$\delta = \min_{x, y \in \Sigma_1^N, x \neq y} \frac{weight(C(x) - C(y))}{M}.$$

But since C is linear, we can just simplify $C(x) - C(y)$ as $C(x - y)$. Thus the distance between $C(x)$ and $C(y)$ is just the weight of $C(x - y)$. So by letting $z = x - y$, we can write

$$\delta = \min_{z \in \Sigma_1^N, z \neq (0^a)^N} \frac{weight(C(z))}{M}. \quad \blacktriangleleft$$

To construct our explicit codes, we need to use pseudorandom objects called extractors and condensers. The goal of these objects is to take inputs with some randomness and a lot of correlations, and give a shorter output with a similar amount of randomness. Extractors want the shorter output to look almost uniform, but often are so short they lose some randomness. Condensers want the shorter output to contain almost all the randomness, but may not be short enough to be close to uniform.

To formally define extractors and condensers, we need to define min entropy: H_∞ . Intuitively, the min entropy is the number of bits of information one always gets from a single output of a distribution. This is in contrast to the standard notion of entropy, which is like the average amount of information a single output gives. Min entropy is a more convenient notion of entropy for us.

► **Definition 12 (Min Entropy).** For any distribution, X , over any alphabet, Σ , we define the min entropy, H_∞ , of X as

$$H_\infty(X) = \max_{\sigma \in \Sigma} -\log(\Pr[X = \sigma]).$$

If X is a uniform distribution over $K = 2^k$ different elements, we call X a flat k source, and $H_\infty(X) = k = \log(K)$.

Now we often don't actually have a low min entropy source. Often there may be one or two inputs that have a large chance of appearing, and we still need to work with these. So we relax our requirements on distributions to not necessarily be high min entropy themselves, but to be close to something with high min entropy. So let us define the distance of two distributions.

► **Definition 13 (Statistical Distance).** For any distributions, X, Y over some alphabet Σ , the distance of X to Y is

$$\Delta(X, Y) = \frac{1}{2} \sum_{\sigma \in \Sigma} |\Pr[X = \sigma] - \Pr[Y = \sigma]|.$$

If $\Delta(X, Y) \leq \epsilon$, we say X is an ϵ approximation of Y or that X is ϵ close to Y .

Distances in this sense compose neatly with functions in the sense that if X is an ϵ approximation of Y , then for any function f , we have that $f(X)$ is also an ϵ approximation of $f(Y)$.

One subtlety of extractors and condensers is that there is no general function that is able to take any input with high entropy and be able to give a smaller output without losing just as much entropy. To do this, we need some extra structure on the input entropy. Here, we provide that structure by giving our extractors and condensers a second, very small, uniform input called a "seed". In this work, we assume all condensers and extractors are seeded.

► **Definition 14 (Seeded Extractor Definition).** Let $E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ be any function. Let U_d be the uniform distribution over $\{0, 1\}^d$.

We say E is a (k, ϵ) extractor if for any distribution, X , over $\{0, 1\}^n$, with min entropy k , we have that $E(X, U_d)$ is ϵ close to the uniform distribution over $\{0, 1\}^m$.

Condensers are defined similarly, except that we don't enforce the output to be close to uniform, but just some high entropy distribution. If the distribution the output is close to has all the entropy of the original distribution plus the entropy of the seed, we call the condenser lossless since it didn't lose any of the input entropy.

► **Definition 15 (Seeded Condenser Definition).** Let $C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ be any function. Let U_d be the uniform distribution over $\{0, 1\}^d$.

We say C is a $(n, k) \rightarrow_\epsilon (m, k')$ condenser if for any distribution, X , over $\{0, 1\}^n$ with min entropy k , we have that $C(X, U_d)$ is ϵ close to some distribution over $\{0, 1\}^m$ with min entropy k' .

If $k' = k + d$, we say that C is a lossless condenser.

To prove our lower bounds for sequential access to the input, we will use a tool called random restrictions. The idea of a random restriction is to fix some of the inputs to a function, and not others.

► **Definition 16** (Restriction). *A restriction of length n is just a string $x^* \in \{0, 1, *\}^n$. Any index where x^* is either 0 or 1 is fixed, and any index where x^* is not fixed, we say it is free.*

If x^ is a restriction with k free indexes, and there is a binary string $y \in \{0, 1\}^k$, then we define the string $x_y = y \circ x^*$ to be the string that agrees with x^* on indexes where x^* is fixed, and on the j th index that x^* is free agrees with y_j .*

6 From Condensers To Codes

In this section, we will show how to construct codes using condensers. We will later show how to construct the condensers we need.

Our codes are constructed by mixing weight fixers, so we will start by defining weight fixers.

6.1 Weight Fixers

A weight fixer is a linear function that outputs strings with a constant relative weight, as long as the weight of the message is within a specified range.

► **Definition 17** (Weight Fixer). *For any two alphabets, Σ_1 and Σ_2 , any subset $R \subseteq [N]$ and constant $\delta \in (0, \frac{1}{2})$, a linear function $F : \Sigma_1^N \rightarrow \Sigma_2^M$ is said to be an R to δ weight fixer if, given any x with $\text{weight}(x) \in R$ then $\text{weight}(F(x)) \geq \delta M$.*

We say that R is the input weight range, δ is the relative output weight, N is the input length, and M is the output length.

A straightforward corollary of this definition is that any $[N]$ to δ weight fixer is a code with relative distance δ , since the distance of a linear code is the weight of its smallest non zero codeword. Thus our strategy will be to combine several weight fixers that cover different parts of the weight range into a single weight fixer that covers the entire weight range.

For large weight messages, we can perform a repeated weight amplification type procedure to get the appropriate weight. The exact way we perform weight amplification is similar to Spielman's codes and described in Section 8.

► **Theorem 18** (Weight Fixer For Heavy messages). *For some constant $\alpha > 0$, and any integers N and i , there is a $[[N/2^i], N]$ to $\alpha/4$ weight fixer $F : \{0, 1\}^N \rightarrow \{0, 1\}^{4N}$. Further, for some constant, c , any bit in the output of F can be computed in time $c^i \text{polylog}(N)$ and space $O(i + \log(N))$.*

Now that we have defined weight fixers, we will show how to efficiently mix them to get better weight fixers, and eventually codes.

6.2 Weight Fixer Mixer

Now we show how to combine many weight fixers with different input ranges, and combine them to get a weight fixer whose input range is their union. The idea is to arrange the output of each weight fixer as rows in a table, where each weight fixer is repeated until they all have the same length, then encode the columns with any arbitrary code.

While this is a geometrically easy way to think about the combination, and what we do in the following theorem, we note it is not optimal if the weight fixers have very different outputs. In particular for our weight fixers. Once can improve the result by splitting very large weight fixers into multiple rows, so that all the small weight fixers don't need to be copied many, many times. This is what is done in [26, Claim 37], but for simplicity, we do not do it here.

5:22 Explicit Time and Space Efficient Encoders Exist Only with Random Access

► **Theorem 19** (Fixer Mixer). *Suppose for some ℓ , for each $i \in [\ell]$ there is an R_i to δ weight fixer $F_i : \Sigma_1^N \rightarrow \Sigma_2^{M_i}$. Let $C : \Sigma_2^\ell \rightarrow \Sigma_3^c$ be any linear code with relative distance δ' computable in time T' and space S' .*

Suppose for some R we have $R \subseteq \bigcup_{i \in [\ell]} R_i$ and for some length M for each $i \in [\ell]$ we have $M_i | M$. Then there is an R to $\delta\delta'$ weight fixer $F : \Sigma_1^N \rightarrow \Sigma_3^{cM}$.

Further, if for each $i \in [\ell]$ any individual output element of F_i is computable in time T_i and space S_i , then the full output of F is computable in space

$$O(\ell \log(|\Sigma_2|) + \log(M)) + \max\{S', \max_{i \in [\ell]} S_i\}$$

and time

$$(T' + O(1))M + \sum_{i \in [\ell]} M_i T_i.$$

Alternatively, if for each $i \in [\ell]$ the full output of F_i is computable in time T_i and space S_i , then the full output of F is computable in space

$$O(\ell \log(|\Sigma_2|) + \log(M)) + S' + \sum_{i \in [\ell]} S_i$$

and time

$$(T' + O(1))M + \sum_{i \in [\ell]} T_i.$$

Proof. The idea of the code is simple. First, we lengthen the output of each weight fixer so that it has length M . Then we apply C element wise to the output of each weight fixer. More specifically, for $a \in [M]$ we define $y_a \in \Sigma_1^\ell$ to be the a th column in the table. That is, for $i \in [\ell]$ we have

$$(y_a)_i = F_i(x)_{\lceil aM_i/M \rceil}.$$

Then F just applies C to each column, y_a , and concatenates them. That is for any $a \in [M]$ and $b \in [c]$ column a row b of the output is just the b th output of $C(y_a)$. That is,

$$F(x)_{(a-1)c+b} = C(y_a)_b.$$

Suppose x has weight $w \in R$. Then for some i , we know $w \in R_i$. Thus $F_i(x)$ has relative weight δ . Thus for at least δ fraction of a , we have $y_a \neq 0$. For each such a , by the distance of C , for δ' fraction of b , we have $C(y_a)_b \neq 0$. Therefore, for at least $\delta\delta'$ fraction of (a, b) pairs we have $F(x)_{(a-1)c+b} \neq 0$. Thus F has relative weight at least $\delta\delta'$.

To encode F , we compute the code for each F_i in parallel and apply C in a straightforward way.

If the individual bits of each weight fixer is efficient to compute, then the space can be reused between the different weight fixers, keeping only the current bit of each F_i and some indexes in memory. This takes space $O(\log(M) + \ell \log(|\Sigma_2|) + \max\{S', \max_{i \in [\ell]} S_i\})$. Similarly the time is just the sum of the time to encode with C for M times, plus some book keeping, plus time to compute every bit of every weight fixer. This is time $(T' + O(1))M + \sum_{i \in [\ell]} M_i T_i$.

If the full output of each L_i is efficient to compute, the encoding algorithms is essentially the same, only now we cannot reuse space between the different weight fixers. This is because we need to pause each weight fixer after it outputs a bit and resume it when we need its next one. This requires space $O(\ell \log(|\Sigma_2|) + \log(M)) + S' + \sum_{i \in [\ell]} S_i$ and time $(T' + O(1))M + \sum_{i \in [\ell]} T_i$, noting that in this case T_i is the time to output all bits, not just a single one. ◀

Now if we had weight fixers covering every input weight, then we could mix them to get codes. Now we show how to get weight fixers from lossless, invertible condensers.

6.3 Weight Fixers From Invertible Condensers

Most of our weight fixers will be constructed through condensers. One additional, non standard property our condensers will need is invertibility. This is because the final weight fixer will enumerate through all the message bits whose index map to an output bits index and xor them together to get that output bit. So it needs to be efficient to, given any output bit index, enumerate through all the message indexes that map to that output. We call a condeser invertible if this can be done efficiently.

To simplify our definitions and proofs slightly, we restrict ourselves to condensers that output an index function along with its condensed output such that the two together is an efficiently invertible function. That is, not only can we enumerate through the inputs that give an output, but given the index of the input that gives an output, compute that specific input efficiently.

► **Definition 20** (Invertible Condeser). *Suppose $C' : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ is an $(n, k) \rightarrow_\epsilon (m, k')$ condeser. Suppose there is a function $I : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^{n+d-m}$, and define $C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$ by*

$$C(x, s) = (C'(x, s), I(x, s)).$$

Suppose C is a bijection with $C^{-1} : \{0, 1\}^m \times \{0, 1\}^{n+d-m} \rightarrow \{0, 1\}^n \times \{0, 1\}^d$ its inverse. Then we say C is an invertible $(n, k) \rightarrow_\epsilon (m, k)$ condeser.

We say C is time T and space S invertible if C^{-1} can be computed in time T and space S . We call C^{-1} the inverse of C , we call C' the condeser part of C , and I the index function of C . We still call d the seed length and m the output length.

To use condensers to create weight fixers, we need the following properties. To get a short output, we need small seed length. Since we have Theorem 18, to handle very heavy messages, seed length $\text{polylog}(n)$ suffices for our work. To get high weight, we need a lossless condeser with only constant entropy gap. That is, the condeser needs to output all the entropy and the number of bits in the output needs to be at most a constant number many more bits than the amount of entropy. Finally, they need to be invertible in polynomial time. Such good condensers exist.

Now we show that lossless condensers, when used as described above, give weight fixers.

► **Theorem 21** (Lossless Condensers give Weight Fixers). *Suppose you have an invertible $(n, k) \rightarrow_\epsilon (k + d + b, k)$ lossless condeser*

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^{k+d+b} \times \{0, 1\}^{n-k-b}$$

with seed length d that is time T and space S invertible. Let $m = k + d + b$.

Then there is a $[2^{k-1/2}, 2^{k+1/2}]$ to $(\frac{1}{2} - 2\epsilon) 2^{-b}$ weight fixer, $F : \{0, 1\}^{2^n} \rightarrow \{0, 1\}^{2^m}$, with input length $N = 2^n$ and output length $M = 2^m$ whose individual output bits can be computed in time $(T + O(1))2^{n-k-b}$ and space $S + O(d + b + n)$.

Proof. Our linear function F identifies every message bit with an n bit index, i , and every output bit with an m bit index, j . Then the output bit at index j is the parity of all message bits x_i where for some seed s we have $C(i, s)_1 = j$. More formally:

$$F(x)_j = \bigoplus_{i, s: C(i, s)_1 = j} x_i = \bigoplus_{i \in \{0, 1\}^{n-k-b}} x_{C^{-1}(j, i)}.$$

By inspection, one can see that the output length is $2^m = M$.

To compute $F(x)_j$, we simply have to invert C to find all 2^{n-k-b} message bits that map to j and xor the corresponding bits together. Since C is time T and space S invertible, this only takes time $T2^{n-k-b}$ to perform each inversion plus $O(2^{n-k-b})$ for book keeping. Similarly for space we just need to keep track of which bit we are outputting, which neighbor of that bit we are at, and the space for the inversion. This is space $O(m) + O(n - k - b) + S$.

To see that F is weight fixing, choose any x with weight $w \in [2^{k-1/2}, 2^{k+1/2}]$. Now we show that for a large fraction of the $j \in \{0, 1\}^m$, there is only one index ℓ and seed s with $x_\ell \neq 0$ such that $C(\ell, s) = j$. This would imply that for such j that

$$F(x)_j = \bigoplus_{i,s:C(i,s)=j} x_i = x_\ell \neq 0.$$

If $w \geq 2^k$, then we will show that any specific set of 2^k ones of x approximately map to unique outputs and the extra ones can't cancel out too much.

Take any $X \subseteq \{0, 1\}^n$ such that $|X| = 2^k$ and for all $i \in X : x_i = 1$. Then there must be at least $(1 - \epsilon)2^{k+d}$ distinct indexes j such that for some index i and seed s we have $C(i, s) = j$. Otherwise, we have a k entropy flat source whose output in expectation over the seed differs from any $k + d$ source by more than ϵ . Then at most $\epsilon 2^{k+d}$ of the index i seed s pairs map to a j for a second time. Thus at least $(1 - 2\epsilon)2^{k+d}$ output indexes j have a unique index $i \in X$ seed s that maps to i and with $x_i = 1$. The rest of the at most $(\sqrt{2} - 1)2^k$ ones in x and 2^d seeds can only hit $(\sqrt{2} - 1)2^{k+d}$ of these.

So at least

$$(1 - 2\epsilon - \sqrt{2} + 1)2^{k+d} > (1/2 - 2\epsilon)2^{k+d}$$

of the output indexes j have a distinct i and s such that $C(i, s) = j$. Thus the output has weight at least $(1/2 - 2\epsilon)2^{k+d}$. This is relative weight $(1/2 - 2\epsilon)2^{-b}$.

If $w \leq 2^k$, then we will show that any specific super set of 2^k ones containing those of x approximately map to unique outputs and the missing ones can't be too many of these.

Take any $X \subseteq \{0, 1\}^n$ such that $|X| = 2^k$ and for all $i \in X : x_i = 1 : x \in X$. Then, as in the last case, at least $(1 - 2\epsilon)2^{k+d}$ output indexes j have a unique $i \in X$ with x_i and seed s that maps to them. Now x is only missing $(1 - \frac{1}{\sqrt{n}})2^k$ of the ones in X . These missing indexes only contribute $(1 - \frac{1}{\sqrt{n}})2^{k+d}$ ones to these output pairs.

So at least

$$(1 - 2\epsilon - 1 + 1/\sqrt{2})2^{k+d} > (1/2 - 2\epsilon)2^{k+d}$$

of the output indexes j have a distinct i and s such that $C(i, s) = j$. Thus the output has weight at least $(1/2 - 2\epsilon)2^{k+d}$. This is relative weight $(1/2 - 2\epsilon)2^{-b}$. ◀

Now since good invertible condensers exist, and good invertible condensers give good weight fixers, good weight fixers exist.

► **Lemma 22 (Weight Fixers For Small Weight Messages).** *For some constant $\beta > 0$, for every $N = 2^n$ and $K = 2^k$, there is a $[2^{k-1/2}, 2^{k+1/2}]$ to β weight fixer $F : \{0, 1\}^N \rightarrow \{0, 1\}^M$ where $M = O(K2^{O(\log(\log(N))^3)})$. Further, any bit of F can be computed in time $O(\text{polylog}(N)\frac{N}{K})$ and space $O(\log(N)^2)$.*

Proof. This is a direct application of Theorem 21 to Theorem 5. So for $\epsilon = 1/10$, we have from Theorem 5 a time $\text{poly}(n)$, space $O(n^2)$ invertible, lossless $(n, k) \rightarrow_\epsilon (m, k + d)$ condenser

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with $d = O(\log(n/\epsilon^3)) = O(\log(n)^3)$ and $m = k + d + O(\log(1/\epsilon))$. Let $b = m - k - d = O(\log(1/\epsilon)) = O(1)$

Then from Theorem 21 there is a $[2^{k-1/2}, 2^{k+1/2}]$ to $(\frac{1}{2} - 2\epsilon) 2^{-b}$ weight fixer, $F : \{0, 1\}^{2^n} \rightarrow \{0, 1\}^{2^m}$, with message length 2^n and output length $2^m = O(K 2^{\log(n)^3})$ whose individual output bits can be computed in time $\text{poly}(n) 2^{n-k-b} = O(\text{polylog}(N) \frac{N}{K})$ and space $O(n^2 + d + b + n) = O(\log(N)^2)$. See that the output weight $(\frac{1}{2} - 2\epsilon) 2^{-b}$ is a positive constant since b is constant and $2\epsilon < \frac{1}{2}$. ◀

Now if we assume we have condensers, we have weight fixers. But not for arbitrarily large constant weight. We handle that next.

6.4 Distance Amplification

So now we have weight fixers that are time and space efficient to compute for every order of magnitude, but these weight fixers only have some constant output weight. It could be very small. We want weight close to 1. So we apply a final weight fixer to them that takes constant relative weight inputs and amplifies them to outputs with weight close to 1.

We note that we have to do this amplification on the individual weight fixers before we combine them, rather than afterward. This is because our distance amplification weight fixer queries it's input in a random order, not sequentially. So the time to compute the amplified weight fixer is proportional to the length of it's output, and the cost to compute a random symbol of it's input. If applied after mixing all of the weight fixers, this time per input symbol will be close to N with close to N outputs, which will take N^2 time. But when applied to an individual weight fixer, it will only increase the time it takes to output a symbol by a constant factor.

Our weight fixer for very heavy messages uses an extractor to group message bits such that all but ϵ fraction of groups has a one in it. Then we just output all the bits in a group as a symbol in the alphabet. While we could output a code of all the bits in a group, we don't need to for our results and doing so would be unnecessarily complicated.

For this to work, we need to define an invertible extractor, similar to an invertible condenser.

▶ **Definition 23** (Invertible Extractor). *Suppose $E' : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ is (k, ϵ) extractor. Suppose there is a buffer function $B : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^{n+d-m}$, and define $E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n-m}$ by*

$$E(x, s) = (E'(x, s), B(x, s)).$$

Suppose E is invertible with inverse $E^{-1} : \{0, 1\}^m \times \{0, 1\}^{n-m} \times \{0, 1\}^d \rightarrow \{0, 1\}^n$. Then we say E is an invertible extractor.

We say E is time T and space S invertible if E^{-1} can be computed in time T and space S . We call E^{-1} the inverse of E , we call E' the extractor part of E , and B the buffer, or index function of E .

Now we show that using our extractors as described before gives a weight fixer.

▶ **Theorem 24** (Extractors give Weight Fixers). *Suppose you have an invertible (k, ϵ) extractor*

$$E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with seed length d that is time T and space S invertible.

$$\text{Let } \Sigma = \{0, 1\}^{2^{n+d-m}}.$$

Then there is a $[2^k, 2^n]$ to $1 - \epsilon$ weight fixer, $F : \{0, 1\}^{2^n} \rightarrow \Sigma^{2^m}$, with message length 2^n and output length 2^m whose individual output bits can be computed in time $(T + O(1)) 2^{n+d-m}$ and space $S + O(n + d + 2^{n+d-m})$ with just 2^{n+d-m} queries to the message.

5:26 Explicit Time and Space Efficient Encoders Exist Only with Random Access

Proof. This is the most crude form of ABNNR [2]. For every output bit $i \in \{0, 1\}^m$, we let $F(x)_i$ be the concatenation of every bit x_j where for some $s \in \{0, 1\}^d$ we have $E(j, s)_1 = i$. This can be easily computed by inverting E for 2^{n+d-m} many times.

For any message with at least 2^k ones, by the extractor property of E , must hit at all but ϵ fraction of outputs, otherwise the corresponding distribution could not be ϵ close to uniform. \blacktriangleleft

The following extractor is a specific instantiation of [12, Theorem 7.2], which is efficiently invertible. See the discussion in Section 7.1.

► **Lemma 25** (Invertible, Very High Entropy Extractors Exist). *For every n, k , and $\epsilon > 0$, there exists a time $\text{poly}(n \log(1/\epsilon))$ space $O(n) + \text{poly}(2^{n-k}/\epsilon)$ invertible (k, ϵ) extractor*

$$E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^k \times \{0, 1\}^{n-k+d}$$

with seed length $d = O(\log(n - k) + \log(1/\epsilon))$.

This extractor requires $2^{\text{poly}(2^{n-k}/\epsilon)}$ preprocessing time.

Proof. This just instantiates [12, Theorem 7.2] with $t = n - k$. \blacktriangleleft

This implies weight fixers for arbitrarily large output weights, starting from any constant weight. We will always use the following lemma where K is within a constant factor of N .

► **Corollary 26** (Weight Fixers with Large Output Weight). *For every $N = 2^n, K = 2^k$, and $\epsilon > 0$, there exists a $[K, N]$ to $1 - \epsilon$ weight fixer, $F : \{0, 1\}^N \rightarrow \Sigma^K$, with message length N and output length K whose individual output bits can be computed in time $\text{poly}(\log(N) \frac{N}{\epsilon K})$ and space $O(\log(N)) + \text{poly}(\frac{N}{\epsilon K})$ using $\text{poly}(\frac{N}{\epsilon K})$ queries to the message.*

Here, $\Sigma = \{0, 1\}^{\text{poly}(\frac{N}{\epsilon K})}$. There is also an additional $2^{\text{poly}(\frac{N}{\epsilon K})}$ preprocessing time.

Proof. First, we apply Lemma 25 to get a time $\text{poly}(n, \log(1/\epsilon)) = \text{poly}(\log(N) \log(1/\epsilon))$ space $O(n) + \text{poly}(2^{n-k}/\epsilon) = O(\log(N)) + \text{poly}(\frac{N}{\epsilon K})$ invertible (k, ϵ) extractor

$$E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^k \times \{0, 1\}^{n-k+d}$$

with seed length $d = O(\log(n - k) + \log(1/\epsilon))$.

See that

$$\begin{aligned} 2^{n+d-k} &= \frac{N}{K} \text{poly}\left(\frac{\log(N/K)}{\epsilon}\right) \\ &= \text{poly}\left(\frac{N}{K\epsilon}\right). \end{aligned}$$

Then we can apply Theorem 24 to get a $[N = 2^n, K = 2^k]$ to $1 - \epsilon$ weight fixer, $F : \{0, 1\}^{N=2^n} \rightarrow \Sigma^{K=2^k}$, with message length $N = 2^n$ and output length $K = 2^k$ whose individual output bits can be computed in time

$$\text{poly}(\log(N) \log(1/\epsilon)) 2^{n+d-k} = \text{poly}(\log(N) \frac{N}{\epsilon K})$$

and space

$$\begin{aligned} S + O(n + d + 2^{n+d-k}) &= S + O(\log(N/K\epsilon) + \frac{N}{K} \text{poly}\left(\frac{\log(N/K)}{\epsilon}\right)) \\ &= S + \text{poly}\left(\frac{N}{K\epsilon}\right) \end{aligned}$$

with just $2^{n+d-k} = \text{poly}(\frac{N}{K\epsilon})$ queries to the message.

Here

$$\begin{aligned}\Sigma &= \{0, 1\}^{2^{n+d-k}} \\ &= \{0, 1\}^{\text{poly}(\frac{N}{K\epsilon})}.\end{aligned}$$

There is also an additional $2^{\text{poly}(\frac{N}{K\epsilon})}$ preprocessing time. \blacktriangleleft

Now we can apply this to both our Spielman style weight fixer and our condenser based weight fixer to get weight fixers for any order of magnitude message, and any constant weight output. This corollary just applies Corollary 26 to the output of Theorem 18.

► **Corollary 27** (Heavy Message, Very Heavy Output Fixers). *Take any integer $K = 2^k$, and any integer $N = 2^n$ such that $N \geq K$. Then for any $\epsilon > 0$, there is a $[K, N]$ to $1 - \epsilon$ weight fixer $F : \{0, 1\}^N \rightarrow \Sigma^M$ where $\Sigma = \{0, 1\}^{\text{poly}(1/\epsilon)}$ and $M = O(N)$. Further any symbol in the output of F can be computed in time $\text{poly}(\frac{N \log(N)}{K\epsilon})$ and space $O(\log(N) + \text{poly}(1/\epsilon))$.*

There is also an additional $2^{\text{poly}(\frac{1}{\epsilon})}$ preprocessing time.

In the same way, this corollary just applies Corollary 26 to the output of Lemma 22.

► **Corollary 28** (Small Message Weight, Large Output Weight Fixers). *For any constant $\epsilon > 0$, for every $N = 2^n$ and $K = 2^k$, there is a $[2^{k-1/2}, 2^{k+1/2}]$ to $1 - \epsilon$ weight fixer $F : \{0, 1\}^N \rightarrow \Sigma^M$ where $M = 2^m = O(K 2^{O(\log(\log(N))^3)})$ and $\Sigma = \{0, 1\}^{\text{poly}(1/\epsilon)}$. Further, any symbol of F can be computed in time $O(\frac{N}{K} \text{poly}(\frac{\log(N)}{\epsilon}))$ and space $O(\log(N)^2 + \text{poly}(1/\epsilon))$.*

There is also an additional $2^{\text{poly}(1/\epsilon)}$ preprocessing time.

6.5 Putting it all together

Now that we have weight fixers for every input range, and they are good, all that is left is to assemble our final code.

Now we can combine all the condenser based weight fixers to get a single weight fixer that works on all small messages. We recall that because of our long seed, we do not get linear length output for all weight ranges. Thus we only combine up to some message weight that is about $\frac{N}{2^d}$ where d is the seed length of the condenser. That is, we invoke the following theorem with $K \frac{N}{2^d}$.

We also don't mix the condenser based weight fixers with the Spielman style ones at this step either, since our weight fixer mixer adds a small overhead to the output length. As noted before, we can fix this by giving a better weight fixer mixer. But we instead mix our small weight fixers first, then mix our large weight fixer with the result.

► **Lemma 29** (Mixing Small Weight Fixers To Get One Weight Fixer). *For any $\epsilon > 0$, for any $N = 2^n$ and $K = 2^k$, there is a $[K]$ to $1 - \epsilon$ weight fixer $F : \{0, 1\}^N \rightarrow \Sigma^M$ with $M = O(K \text{poly}(1/\epsilon) 2^{O(\log(\log(N))^3)})$ and $\Sigma = \{0, 1\}^{\text{poly}(1/\epsilon)}$. Further, the full output of F can be computed in space*

$$O(\log(N)^2 + \log(N) \text{poly}(1/\epsilon))$$

and time

$$N 2^{O(\log(\log(N))^3)} \text{poly}(1/\epsilon) + 2^{\text{poly}(1/\epsilon)}.$$

Proof. The proof works by invoking Corollary 28 for every $i \leq K$ and combining them with Theorem 19.

So specifically, for every $i \leq k$, Corollary 28 gives a $[2^{i-1/2}, 2^{i+1/2}]$ to $(1-\epsilon/2)$ weight fixer $F_i : \{0, 1\}^N \rightarrow \Sigma_1^{M_i}$ where $M_i = O(2^i 2^{O(\log(\log(N))^3)})$. Further, any bit of F_i can be computed in time $O(\frac{N}{2^i} \text{poly}(\frac{\log(N)}{\epsilon}))$ and space $O(\log(N)^2 + \text{poly}(1/\epsilon))$. Here $\Sigma_1 = \{0, 1\}^{\text{poly}(1/\epsilon)}$.

There is also an additional $2^{\text{poly}(1/\epsilon)}$ preprocessing time. This preprocessing is the same for each i .

Finally, to use Theorem 19, we need the existence of some efficient code, linear, code from $k \text{poly}(1/\epsilon)$ bits to $k \text{poly}(1/\epsilon)$ symbols of $O(\text{poly}(1/\epsilon))$ bits with distance $1 - \epsilon/2$. Since this is a code on only $k \text{poly}(1/\epsilon)$ bits, we can afford to use a less efficient code. So we can for instance use a Spielman code [48] with [2] (this is the same code as Corollary 27 with $K = 1$, evaluated in a more time, less space efficient way) to get such a code, call it $C : \Sigma_1^k \rightarrow \Sigma_2^{k'}$ where $\Sigma_2 = \{0, 1\}^{\text{poly}(1/\epsilon)}$ and $k' = O(k \text{poly}(1/\epsilon))$.

Note that since each M_i is a power of 2, we can upper bound the least common multiple of the M_i s by some $M' = K 2^{O(\log(\log(N))^3)}$.

Now we can apply Theorem 19 to get a $[K]$ to $(1 - \epsilon/2)^2 > (1 - \epsilon)$ weight fixer $L : \{0, 1\}^N \rightarrow \Sigma_2^{O(M'k')}$. Let

$$\begin{aligned} M &= M'k' \\ &= O(K \log(K) \text{poly}(1/\epsilon) 2^{O(\log(\log(N))^3)}) \\ &= O(K \text{poly}(1/\epsilon) 2^{O(\log(\log(N))^3)}). \end{aligned}$$

Further the full output of L is computable in space

$$O(k \log(|\Sigma_2|) + \log(M) + \log(N)^2 + \text{poly}(1/\epsilon)) = O(\log(N) \text{poly}(1/\epsilon) + \log(N)^2)$$

and time

$$O(M \text{poly} \log(N) + \sum_{i \leq k} 2^i 2^{O(\log(\log(N))^3)} \frac{N}{2^i} \text{poly}(\frac{\log(N)}{\epsilon})) = O(N 2^{O(\log(\log(N))^3)} \text{poly}(1/\epsilon)).$$

Adding in the $2^{\text{poly}(1/\epsilon)}$ preprocessing time, this gives a total time of

$$N 2^{O(\log(\log(N))^3)} \text{poly}(1/\epsilon) + 2^{\text{poly}(1/\epsilon)}. \quad \blacktriangleleft$$

Now that we have a weight fixer for light messages and a weight fixer for heavy messages, we can combine them to get a weight fixer for every message weight, which must be a code since weight fixers are linear. We now prove Theorem 1.

► **Theorem 1** (Explicit Almost Linear Time, Polylog Space Encodable Codes). *For any $\epsilon > 0$, and N , there exists a linear code*

$$C : \{0, 1\}^N \rightarrow \Sigma^M$$

that has relative distance $1 - \epsilon$, output length $M = O(N)$ and alphabet $\Sigma = \{0, 1\}^{\text{poly}(1/\epsilon)}$. Further C is computable in time $N \text{poly}(2^{\log(\log(N))^3} / \epsilon) + 2^{\text{poly}(1/\epsilon)}$ and space $O(\log(N)^2 + \log(N) \text{poly}(1/\epsilon))$ with random access to the message.

For constant ϵ , we have constant alphabet size, $\Sigma = \{0, 1\}^{O(1)}$, and further C is computable in time $N^{1+o(1)}$ and space $O(\log(N)^2)$.

Proof. The basic idea is to combine Lemma 29 with Corollary 27 setting

$$K = \frac{N}{\log(1/\epsilon) 2^{O(\log(\log(N))^3)}}.$$

This is the setting at which Lemma 29 stops having linear length, and Corollary 27 still has almost linear time. In particular, choose c to be the constant so that the output length of Lemma 29 is $M \leq cK(1/\epsilon)^c 2^{c \log(\log(N))^3}$, and set

$$K = \frac{N}{c(1/\epsilon)^c 2^{c \log(\log(N))^3}}$$

so that

$$\begin{aligned} M &\leq cK(1/\epsilon)^c 2^{c \log(\log(N))^3} \\ &= N. \end{aligned}$$

Then by Lemma 29 there is a $[K]$ to $1 - \epsilon$ weight fixer $F_1 : \{0, 1\}^N \rightarrow \Sigma'^M$ with $\Sigma' = \{0, 1\}^{\text{poly}(1/\epsilon)}$ and $M = O(K \text{poly}(1/\epsilon) 2^{O(\log(\log(N))^3)})$. Further, the full output of F_1 can be computed in space

$$O(\log(N)^2 + \log(N) \text{poly}(1/\epsilon))$$

and time

$$N 2^{O(\log(\log(N))^3)} \text{poly}(1/\epsilon) + 2^{\text{poly}(1/\epsilon)}.$$

Using the same K with Corollary 27 gives a $[K, N]$ to $1 - \epsilon$ weight fixer $F_2 : \{0, 1\}^N \rightarrow \Sigma'^{M'}$ where $\Sigma' = \{0, 1\}^{O(\text{poly}(1/\epsilon))}$ and $M' = O(N)$. Further any bit in the output of F can be computed in time $\text{poly}(\frac{N \log(N)}{K\epsilon}) = \text{poly}(2^{\log(\log(N))^3} / \epsilon)$ and space $O(\log(N) + \text{poly}(1/\epsilon))$. So one can say the entire output of F_2 can be computed in time $O(N \text{poly}(2^{\log(\log(N))^3} / \epsilon))$ and space $O(\log(N) + \text{poly}(1/\epsilon))$.

There is also an additional $2^{\text{poly}(1/\epsilon)}$ preprocessing time.

Now for the code needed by Theorem 19, we use the trivial code that just outputs one symbol containing the entire message. This has distance 1, and has output alphabet $\Sigma = \Sigma'^2 = \{0, 1\}^{\text{poly}(1/\epsilon)}$.

Then by Theorem 19 there is an $[N]$ to $1 - \epsilon > 1$ weight fixer $L : \{0, 1\}^N \rightarrow \Sigma^{2M' = M = O(N)}$. That is, L is a linear code with distance $1 - \epsilon$.

Further, the full output of L is computable in space

$$O(\log(N)^2 + \log(N) \text{poly}(1/\epsilon))$$

and time

$$N \text{poly}(2^{\log(\log(N))^3} / \epsilon) + 2^{\text{poly}(1/\epsilon)}. \quad \blacktriangleleft$$

7 Constructing Invertible Condensers

Our condensers need to have constant entropy gap, polylogarithmic seed length, and be efficiently invertible. All the condenser constructions we know of do not achieve all three.

The condensers of Capalbo, Reingold, Vadhan, and Wigderson [12] require non-explicit gadgets which take too long to find and too much space to store if the starting entropy gap is too large. The condensers of Guruswami, Umans, and Vadhan [25] or Kalev and Ta-Shma [33] do not have small seed length while having constant entropy gap. Another approach would be to use very good extractors, like those of Ta-Shma, Umans, and Zuckerman [50], then apply a condenser to concentrate the remaining entropy to get a lossless condenser with small entropy gap. Unfortunately, the extractors of [50] or [25] don't appear to be invertible.

$$(C', I')(x, s_1 \circ s_2 \circ s_3 \circ s_4) = (y_1 \circ y_2 \circ y_3, w_1 \circ w_2).$$

$(y_1, z_1) = (E_1, B_1)(x, s_1)$	Trevisan Extractor
$(z_2, w_1) = (C_1, I_1)(z_1, s_2)$	Multiplicity Condenser
$(y_2, z_3) = (E_2, B_2)(z_2, s_3)$	Left-over Hash Extractor
$(y_3, w_2) = (C_2, I_2)(z_3, s_4)$	Iterated Multiplicity Condenser.

■ **Figure 5** Our Condenser Diagram.

The issue with the standard condense and extract framework is that they run many condensers on the same message in parallel. Thus even if individually each condenser is efficiently invertible, it is unclear how to invert them all together efficiently. To see what we mean, consider the condenser which takes a message, x , applies an extractor E to x to output a $0.9k$ bits of entropy, then applies a condenser, C , to x to output a length $0.11k$ bit output containing the remaining $0.1k$ bits of entropy. Then the final result, $C'(x) = (E(x), C(x))$ is indeed a lossless condenser with output length $1.01k$, so has an entropy gap of $0.01k$.

It is not clear how to both time and space efficiently invert C' . For any (y_1, y_2) , there are about $2^{n-0.9k}$ values of x such that $E(x) = y_1$, and about $2^{n-0.11k}$ values of x such that $C(x) = y_2$. It is not space efficient to hold all such x in memory, so for every x such that $E(x) = y_1$, we need to check it against every value of x such that $C(x) = y_2$. This would take time around $2^{2n-1.01k}$ to enumerate through every input that condenses to a given output. When $n = \log(N)$, this is around quadratic time, which is too much for our application.

This issue is fixed if C does not condense from x directly, but instead condenses from some index function of E , or a buffer as [12] would call it. Then as long as E and C are invertible, we can invert C' in a sequential way. With this change, condensers made using a condense then extract framework can be made efficiently invertible, as long as the component condensers and extractors are.

Since we are not too concerned about the seed length, our construction will use the extractors of Trevisan [51, 41]. To conserve space in the algorithm we use the explicit, log space weak combinatorial designs of Hartman and Raz [28]. To make sure that Trevisan's extractor is invertible, we need to restrict how large k can be, and handle certain “bad” seeds.

We will also need to use a variant of the condensers of [33], and the left-over hash lemma based extractors [31, 39] to finish extracting the rest of the bits in the input, and turn it into a condenser. So the final condenser runs the Trevisan extractor, E_1 , to extract all but $O(\log(n/\epsilon)^3)$ bits of entropy. Then we run the multiplicity condenser, C_1 , followed by the hash based extractor, E_2 , to extract all but $O(\log(1/\epsilon))$ bits of entropy. Finally, we run an iterated version of the multiplicity code condenser, C_2 , to condense the remaining entropy into $O(\log(1/\epsilon))$ bits: $y_3 = C_2(z_3, s_4)$. See Figure 5 for more explicit equations.

Before we start proving the provided condensers exist, we will give our composition theorems.

7.1 Composition Theorems

To construct our condenser, we will compose invertible condensers and extractors together. So we first need to define invertible extractors. Invertible extractors are the same as permutation extractors of [12], which are themselves a special case of buffered extractors, with the extra

condition that the buffered extractor is efficient to invert. Equivalent composition theorems are found in [12], our only change is including inversion time and space as a consideration. Thus all the condensers of [12] are also efficiently invertible.

For our extractor composition to work, we need the following observation of distributions. This seems to be well known folklore, so we will not prove it here.

► **Lemma 30** (Approximate Uniform Marginals Keep Entropy). *Suppose for some joint distribution, X, Y , with min entropy k , if X, Y is ϵ close to some joint distribution U, V where U is uniform, then X, Y is also ϵ close to some distribution U, W with min entropy k for the same, uniform U . Additionally, for any $x \in \{0, 1\}^{|U|}$, we have $W|U = x$ has entropy $k - |U|$.*

Our next lemma just says if you apply an invertible condenser to the buffer of the invertible extractor, you get a better invertible condenser.

► **Lemma 31** (Invertible Condensers Compose With Extractors). *Suppose there is a time T_1 and space S_1 invertible, (k, ϵ_1) , invertible extractor $E : \{0, 1\}^n \times \{0, 1\}^{d_1} \rightarrow \{0, 1\}^{m_1} \times \{0, 1\}^{n-m_1}$ and a time T_2 , space S_2 invertible, $(n - m_1, k')$ \rightarrow_{ϵ_2} $(m_2, k - m_1)$ invertible condenser $C : \{0, 1\}^{n-m_1} \times \{0, 1\}^{d_2} \rightarrow \{0, 1\}^{m_2} \times \{0, 1\}^{n-m_1-m_2}$.*

Then there is a time $T_1 + T_2 + O(1)$, space $O(n) + \max\{S_1, S_2\}$ invertible, $(n, k) \rightarrow_{\epsilon_1 + \epsilon_2}$ $(m_1 + m_2, m_1 + k')$ condenser $C' : \{0, 1\}^n \times \{0, 1\}^{d_1 + d_2} \rightarrow \{0, 1\}^{m_1 + m_2} \times \{0, 1\}^{n + d_1 + d_2 - m_1 - m_2}$.

Proof. For any $x \in \{0, 1\}^n$, $s_1 \in \{0, 1\}^{d_1}$, and $s_2 \in \{0, 1\}^{d_2}$, define

$$\begin{aligned} y_1 &= E_{s_1}^E(x) \\ u &= E_{s_1}^I(x) \\ y_2 &= C_{s_2}^C(u) \\ z &= C_{s_2}^I(u) \\ C'_{(s_1, s_2)}{}^C(x) &= (y_1, y_2) \\ C'_{(s_1, s_2)}{}^I(x) &= z. \end{aligned}$$

Then observe that one can compute $C'_{(s_1, s_2)}{}^{-1}((y_1, y_2), z)$ by

$$\begin{aligned} u &= C_{s_2}^{-1}(y_2, z) \\ x &= E_{s_1}^{-1}(y_1, u). \end{aligned}$$

The space here is just the space to store u plus the max of the space to invert C and E . Similarly the time is the time to invert C and the time to invert E .

Now to show that C'^C is a condenser, first see that given any X with entropy k , since E^E is a strong extractor, we have that for uniform s_1 , distribution s_1, y_1 is ϵ close to uniform. Further, since E is invertible, the distribution s_1, y_1, u has the same entropy as X , thus has entropy $k + d$.

By Lemma 30 we have that s_1, y_1, u is ϵ close to some distribution s_1, U, u' where s_1 and U are uniform, and u' has entropy $k - m_1$ conditioned on s_1 and U . Let $y'_2 = C_{s_2}^C(u')$. Then since C is a condenser, we have that $s_2 y'_2$ is ϵ_2 close to a distribution that has entropy $d_2 + k'$ for every correlated value of s_1 and U . Thus $s_1 U s_2 y'_2$ is ϵ_2 close to a $d_1 + m_1 + d_2 + k'$ source. Thus $s_1, y_1, s_2 y_2$ is $\epsilon_1 + \epsilon_2$ close to being a $d_1 + m_1 + d_2 + k'$ source. Therefore, C^C is a strong $(n, k) \rightarrow_{\epsilon_1 + \epsilon_2} (m_1 + m_2, k')$, invertible extractor. ◀

The following theorem just says that if you apply an invertible condenser to the output of a condenser you get a better condenser.

► **Theorem 32** (Invertible Condensers Compose With Condensers). *Given a time T_1 and space S_1 invertible $(n, k) \rightarrow_{\epsilon_1} (m_1, k_1)$ invertible condenser $A : \{0, 1\}^n \times \{0, 1\}^{d_1} \rightarrow \{0, 1\}^{m_1} \times \{0, 1\}^{n-m_1}$ and a time T_2 space S_2 invertible, $(m_1, k_1) \rightarrow_{\epsilon_2} (m_2, k_2)$ condenser $B : \{0, 1\}^{m_1} \times \{0, 1\}^{d_2} \rightarrow \{0, 1\}^{m_2} \times \{0, 1\}^{m_1-m_2}$.*

Then there is a time $T_1 + T_2 + O(1)$ space $O(n) + \max\{S_1, S_2\}$ invertible, $(n, k) \rightarrow_{\epsilon_1+\epsilon_2} (m_2, k_2)$ invertible condenser

$$C : \{0, 1\}^n \times \{0, 1\}^{d_1+d_2} \rightarrow \{0, 1\}^{m_2} \times \{0, 1\}^{n+d_1+d_2-m_2}.$$

Proof. For any $x \in \{0, 1\}^n$, $s_1 \in \{0, 1\}^{d_1}$, and $s_2 \in \{0, 1\}^{d_2}$, define

$$\begin{aligned} u &= A_{s_1}^C(x) \\ z_1 &= A_{s_1}^I(x) \\ y &= B_{s_2}^C(u) \\ z_2 &= B_{s_2}^I(u) \\ C_{(s_1, s_2)}^C &= y \\ C_{(s_1, s_2)}^I &= (z_1, z_2). \end{aligned}$$

Then observe that one can compute $C_{(s_1, s_2)}^{-1}(y, (z_1, z_2))$ by

$$\begin{aligned} u &= B_{s_2}^{-1}(y, z_2) \\ x &= A_{s_1}^{-1}(u, z_1). \end{aligned}$$

This only takes space that is the max of the space to invert B and the space to invert A plus space to hold u . It also only takes the time to invert B and A plus some book keeping.

Since A^c is a strong condenser, s_1, u is an ϵ_1 approximation of some $d_1 + k_1$ source. Then since B^c is a strong condenser, we have that s_1, s_2, y is an $\epsilon_1 + \epsilon_2$ approximation of a $d_1 + d_2 + k_2$ source. ◀

7.2 Our Base Condenser

For our base condensers that we compose with extractors to make our final condenser, we use the condenser of Kalev and Ta-Shma [33]. These condensers are based on multiplicity codes, which we find easier to understand, and thus invert, then the condensers based on Pavarash-Vardy codes of [25]. While these condensers are great at condensing inputs with k bits of entropy into $O(k)$ bits of output, they can not efficiently condense inputs with k bits of entropy to $k + O(\log(1/\epsilon))$ bits, which is what we need. This is why we need to perform composition to get our final condenser.

► **Lemma 33** (Invertible Lossless Expander). *For every field \mathbb{F}_q and integers $\ell, s \in \mathbb{N}$ with $15 \leq s \leq \ell \leq \text{char}(\mathbb{F}_q)$, there exists an explicit graph $\Gamma : \mathbb{F}_q^\ell \times \mathbb{F}_q \rightarrow \mathbb{F}_q^s$ which is a (K, A) expander for every $K > 0$ with*

$$A = q - \frac{\ell s}{2} (qK)^{\frac{1}{s}}.$$

Further, Γ is invertible in time $O((\ell \log(q))^2)$ and space $O(\ell \log(q))$.

Proof. This expander is from [33, Theorem 1.3], all we need to show is that Γ is efficiently invertible. Examining Γ , we see it is actually a very straightforward construction based on multiplicity codes. It views its first input as a degree at most ℓ polynomial, p , and its second input as an evaluation point, x . Then it outputs the first s Hasse derivatives of p evaluated at y .

The expander Γ can be made invertible by also evaluating the remaining $\ell - s$ potentially non zero Hasse derivatives at y to create the index function. Then one can reconstruct the original polynomial using a Taylor expansion at y . This only takes time $O((n \log(q))^2)$ and space $O(n \log(q))$. ◀

Kalev and Ta-Shma [33] carefully choose parameters to get very good output length, at the cost of severe restrictions on α, k, n , and ϵ . The proof is in [33].

► **Lemma 34** (Kalev and Ta-Shma Condensers). *For every set of integers $k \leq k_{max} \leq n$ and $\epsilon > 0$ with $\frac{16 \log(\frac{n}{\epsilon})}{\sqrt{k}} \leq \alpha \leq 1$, there is a time $O(n^2)$ space $O(n)$ invertible $(n, k) \rightarrow_{\epsilon} (m, k + d)$ lossless, invertible condenser $C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n-m}$ with $d = (1 + 1/\alpha) \log(nk_{max}/\epsilon) + O(1)$ and $m \leq (1 + \alpha)k_{max}$.*

These limitations on the parameters of α, k, n and ϵ are inconvenient, so we give an alternate choice of parameters that gives a worse condenser, but are easier for us to work with.

► **Lemma 35** (Our Basic Condenser). *For every set of integers $k \leq k_{max} \leq n$ and $1 > \epsilon > 0$ with $26 \log(2n/\epsilon) \leq k_{max} \leq \frac{n}{2}$, there is a time $O(n^2)$ space $O(n)$ invertible $(n, k) \rightarrow_{\epsilon} (m, k + d)$ lossless, invertible condenser*

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n-m}$$

with $d = O(\log(nk_{max}/\epsilon))$ and $m \leq 2k_{max} + O(\log(n/\epsilon))$.

Proof. Let $q' = \left(\frac{2nk_{max}}{\epsilon}\right)^2$, and q be any prime between q' and $2q'$. Let $K = 2^{k_{max}}$. Then we choose s to be the smallest integer so that $q^{s-2} \geq K^2$. Finally we set $\ell = \lceil \frac{n}{\log(q)} \rceil + 2$.

Now we show that we can apply Lemma 33. For this, we need to show $15 \leq s \leq \ell \leq q$. To show that $15 \leq s$, we just need to show that $q^{13} < K^2$. This can be shown by

$$\begin{aligned} \log(q^{13}) &= 13 \log(q) \\ &< 13 \log(2q') \\ &= 13 \log\left(2 \left(\frac{2nk_{max}}{\epsilon}\right)^2\right) \\ &\leq 13 \log\left(2 \left(\frac{n^2}{\epsilon}\right)^2\right) \\ &< 13 \log\left(\left(2\frac{n}{\epsilon}\right)^4\right) \\ &< 52 \log(2n/\epsilon) \\ &< 2k_{max}. \end{aligned}$$

Finally exponentiating both sides gives $q^{15-2} < K^2$, thus s must be at least 15.

To see that $s \leq \ell$, it suffices to show that $q^{\ell-2} \geq K^2$. But this must be true since $q^{\ell-2} \geq 2^n \geq 2^{2k_{max}} = K^2$. Finally it is clear that $q \geq n$ since $q \geq q' > n$. Thus we can apply Lemma 33.

Due to Lemma 33, there is an explicit graph $\Gamma : \mathbb{F}_q^{\ell} \times \mathbb{F}_q \rightarrow \mathbb{F}_q^s$ which is a (K, A) expander for every $K > 0$ with

$$A = q - \frac{\ell s}{2} (qK)^{\frac{1}{s}} = q \left(1 - \frac{1}{q} \frac{\ell s}{2} (qK)^{\frac{1}{s}}\right)$$

Further, Γ is invertible in time $O((\ell \log(q))^2) = O(n^2)$ and space $O(\ell \log(q)) = O(n)$.

Now we want to show that A , the expansion rate of size K sets, is very close to q , the degree, to get a lossless condenser. To do this, we show that $\frac{1}{q} \frac{\ell s}{2} (qK)^{\frac{1}{s}}$ is at most epsilon. See that since $q^{s-2} \geq K^2$, we also have $K \leq q^{s/2-1}$. We also have that $q^{s-3} < K^2$, thus $s < \frac{2k_{max}}{\log(q)} + 3 < 2k_{max}$ since $k_{max}, q > 16$. We also have that $\ell < n$. Thus

$$\begin{aligned} \frac{1}{q} \frac{\ell s}{2} (qK)^{\frac{1}{s}} &\leq \frac{1}{q} \frac{\ell s}{2} (qq^{s/2-1})^{\frac{1}{s}} \\ &\leq \frac{\ell s}{2\sqrt{q}} \\ &< \frac{n2k_{max}}{2\sqrt{q}} \\ &= \epsilon/2. \end{aligned}$$

Thus we can bound A by

$$A \geq q(1 - \epsilon/2).$$

Also see that $q^\ell > n$. Then our final condenser just interprets the input as an element of \mathbb{F}_q^ℓ , and, its seed as an element of \mathbb{F}_q , and outputs an element of \mathbb{F}_q^s . On a technical note, q is not a power of 2, and in fact may be far from a power of 2. While we could work with this, to get our stated result, we will need to sample the same element of \mathbb{F}_q for multiple seeds so that our distribution of seeds is approximately uniform. We can $\epsilon/2$ approximate a uniform distribution over \mathbb{F}_q with $\log(q) + O(\log(1/\epsilon))$ extra bits, which we do. But to avoid collisions, we need to include the extra $O(\log(1/\epsilon))$ bits of seed in the output.

We claim this is a $(n, k) \rightarrow_\epsilon (m, k + d)$ condenser with seed length is $d = \log(q) + O(\log(1/\epsilon)) = O(\log(n/\epsilon))$, and output bit length $s \log(q) + O(\log(1/\epsilon)) = 2k_{max} + O(\log(q) + \log(1/\epsilon)) = 2k_{max} + O(\log(n/\epsilon))$. The seed length and output bit is given directly from Γ and the extra padding needed to work over bits.

To see that it is a condenser, we simply observe that any flat k source is a uniform distribution over an element of \mathbb{F}_q^ℓ , and if our seed was uniform over \mathbb{F}_q , then the output could only have at most $\epsilon/2$ fraction of collisions, thus is an $\epsilon/2$ approximation of a $k + \log(q)$ source. Since our seed is an $\epsilon/2$ approximation of an element of \mathbb{F}_q and the extra $d - \log(q)$ entropy is copied directly to the output, our output is an ϵ approximation of a $k + d$ source. \blacktriangleleft

We want to use our condenser both where k is around $O(\log(n/\epsilon))$ and when $k = O(\log(1/\epsilon))$. Lemma 35 is already good enough when $k = O(\log(n))$, but the restriction of $k_{max} = \Omega(\log(n/\epsilon))$ makes our condenser output length too long when $k = O(\log(1/\epsilon))$. By composing this condenser with itself $\log^*(n)$ many times we get a condenser that handles even constant entropy k . Since each successive instance is so much smaller than the first, this gives the same asymptotic time, space, and seed length.

► Corollary 36 (Iterated Basic Condenser). *For every set of integers $k \leq k_{max} \leq \frac{n}{2}$ and $\frac{1}{2} > \epsilon > 0$ with $100 \log(1/\epsilon) \leq k_{max} \leq n$, there is a time $O(n^2)$ space $O(n)$ invertible, $(n, k) \rightarrow_\epsilon (m, k + d)$ lossless, invertible condenser $C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n-m}$ with $d = O(\log(nk_{max}/\epsilon))$ and $m \leq 2k_{max} + d + O(\log(1/\epsilon))$.*

7.3 Our Condenser for Polylogarithmic Entropy

While the proceeding condensers are quite good, they cannot by themselves get a constant entropy gap. That is, they can't give number of output bits with $m = k + O(\log(1/\epsilon))$ unless $k = O(\log(1/\epsilon))$. Here, we show we can give a lossless condenser, albeit with very long seed length, that outputs all $k + d$ bits of entropy into a length $k + d + O(\log(1/\epsilon))$ bit output.

This will just use the left-over hash lemma based extractor [31] composed with our iterated condenser, Corollary 36 to get a condenser with an $O(\log(1/\epsilon))$ entropy gap. So first, we will state the extractor given by the left-over hash lemma.

► **Lemma 37** (Base Case Extractor). *For any $n > k > 0$ and ϵ , there is a time $O(n^2)$, space $O(n)$ invertible (k, ϵ) extractor*

$$E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n-m}$$

with seed length $d = O(n)$, and $m = k + d - O(\log(1/\epsilon))$.

Proof. This is an extractor based on a pairwise independent hash function. The soundness is based on the well known leftover hash lemma. For invertibility, we just use the hash function that views input x as an element of \mathbb{F}_{2^n} , and the seed as two elements $a, b \in \mathbb{F}_{2^n}$ and outputs the m least significant bits of

$$ax + b$$

and the seed. And the index function are the $n - m$ most significant bits of $ax + b$. Then inversion is straightforward. ◀

Now we can use our extractor with our multiplicity based condenser to get a new condenser.

► **Corollary 38** (Base Case Condenser). *For any $n > k > 0$ and ϵ , there is a time $O(n^2)$, space $O(n)$ invertible, $(n, k) \rightarrow_\epsilon (m, k + d)$ condenser*

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with seed length $d = O(n)$, and $m = k + O(\log(1/\epsilon))$.

Proof. By Lemma 37, there is an extractor a time $O(n^2)$, space $O(n)$ invertible $(k, \epsilon/2)$ extractor

$$E : \{0, 1\}^n \times \{0, 1\}^{d_1} \rightarrow \{0, 1\}^{m_1} \times \{0, 1\}^{n-m_1}$$

with seed length $d_1 = O(n)$, and $m_1 = k + d_1 - O(\log(1/\epsilon))$. Let $k_1 = k + d_1 - m_1 = O(\log(1/\epsilon))$

By Corollary 36, there is a time $O(n^2)$ space $O(n)$ invertible, strong $(n - m_1, k_1) \rightarrow_{\epsilon/2} (d_2 + O(k_1), d_2 + k_1)$ invertible condenser

$$C_1 : \{0, 1\}^n \times \{0, 1\}^{d_2} \rightarrow \{0, 1\}^{m_2} \times \{0, 1\}^{n-m_2}$$

with $d_2 = O(\log(nk_{max}/\epsilon))$ and $m_2 \leq 2k_1 + d_2 + O(\log(1/\epsilon))$.

By Lemma 31 the final result is a time $O(n^2)$, space $O(n)$ invertible $(n, k) \rightarrow_\epsilon (m_1 + m_2 = k + d_1 + d_2 + O(\log(1/\epsilon)), m_1 + d_2 + k_1 = k + d_1 + d_2)$ condenser

$$C : \{0, 1\}^n \times \{0, 1\}^{d_1+d_2} \rightarrow \{0, 1\}^{m_1+m_2} \times \{0, 1\}^{n-m_1-m_2}.$$

Finally see that the seed length of C is $d_1 + d_2 = O(n)$. ◀

7.4 Final Condenser

Now to construct our final condenser, we first start with an invertible extractor that can extract most of the entropy (all but $\text{polylog}(n/\epsilon)$). We start with the time and space efficient variation of Trevisan's extractor [51, 41] by Hartman and Raz [28]. This extractor is invertible because it is linear, a fact commonly used by non-malleable extractors [14, 37]. We note that this choice of extractor (and the techniques used to invert it) are the main bottleneck to getting log space and quasilinear time encoders. Better seed length and linear space inversion of our condenser would give us better encoders.

The Trevisan extractor algorithm depends on two constructions, a code with good distance, and a weak combinatorial design. We use the same code and weak design as [28].

► **Lemma 39 (Invertible Trevisan Extractor).** *For every n, k and ϵ such that $k \leq n$ and $\epsilon > 2^{1-n/\log^*(n)^{\log^*(n)}}$, there is a time $\text{poly}(n)$, space $O(n^2)$ invertible (k, ϵ) extractor*

$$E : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with seed length $d = \Theta(\log(n/\epsilon)^3)$ and $m = k$

Proof. We start with the extractor in [28, Theorem 10] using $\epsilon/2$ in place of ϵ .

This immediately gives us a $(k, \epsilon/2)$ extractor

$$E' : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$$

with seed length $d = O(\log(n/\epsilon)^3)$ and $m = k$. Unfortunately, E' is probably not invertible. To convert it into an invertible extractor, we need to first explain how E' is computed.

The extractor E' is built using a code, $C : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$, where n' is a power of two and a weak combinatorial design, $S : [m] \rightarrow [d]^{\log(n')}$. Given an input x and a seed s , the extractor E' outputs

$$y = (C(x)|_{s(S(1))}, C(x)|_{s(S(2))}, \dots, C(x)|_{s(S(m))})$$

where $C(x)|_{s(S(i))}$ means to first concatenate the bits in s indicated by $S(i)$, and then use the resulting string to index into C .

Importantly, S can be computed in $\log(n)$ space and time $\text{poly}(n)$. And C is a Reed-Solomon code concatenated with a Hadamard code. In particular, for some a with $\log(n/\epsilon) < a < 4 \log(n/\epsilon)$, code C is the Reed Solomon codes over \mathbb{F}_{2^a} with degree at most n/a composed with the Hadamard code. The important thing about this extractor is that after a given seed is chosen, the extractor is a linear function whose generator matrix can be found in polynomial time and space $O(n^2)$.

So if for a given seed s , all the output bits are linearly independent, then one can create a full rank matrix where the first k rows output the extractor by a greedy search in $\text{poly}(n)$ time and $O(n^2)$ space. Applying this matrix, and then appending the seed, gives the buffered extractor. And by Gaussian elimination, one can invert this matrix again in $\text{poly}(n)$ time and $O(n^2)$. Thus for these “good” seeds, one where the output bits are linearly independent, one can invert this extractor efficiently.

When the seed is “bad”, that is the output bits are not linearly independent, one can also detect this in time $\text{poly}(n)$ and space $O(n^2)$, again using Gaussian elimination. For these bad seeds, the extractor already fails, so we just output the entire input. This is trivially invertible.

So in our final invertible extractor, we take an input x and a seed s and first check if it is bad. If it is, we give up and output the input and the seed. Otherwise, we output for our extractor part $E'(x, s)$ and for our buffer part s along with the rest of the information need

to invert E' . The buffer here only needs to output a d bit seed, plus the information about x missing from the extractor, which is just $n - k$ bits, exactly what we need: an $n + d - m$ bit buffer.

To see the result is an extractor, all we need to note is that the seed is bad rarely. This is because E' outputs an $\epsilon/2$ approximation of the uniform distribution and when a seed is bad, since the extractor is linear, it E' can only hit at most half the space of outputs. So when the seed is bad, that seed has distance $1/2$ from uniform. So E only doubles the error on bad seeds and maintains the same error on all other seeds. Thus the error of E is at most ϵ . ◀

Now we can construct our final condenser by composing our invertible Trevisan Extractor, Lemma 39 with our multiplicity condenser Lemma 35 and our base case condenser Corollary 38. We now prove Theorem 5.

► **Theorem 5** (Good Invertible Condensers Exist). *For every n, k and ϵ such that $\epsilon > 2^{3-n/\log^*(n)^{\log^*(n)}}$, there is a time $\text{poly}(n)$, space $O(n^2)$ invertible, lossless $(n, k) \rightarrow_\epsilon (m, k+d)$ condenser*

$$C : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m \times \{0, 1\}^{n+d-m}$$

with $d = O(\log(n/\epsilon)^3)$ and $m = k + d + O(\log(1/\epsilon))$.

Proof. For small enough constant n , we can just use the probabilistic method to find the condenser. So take n to be a sufficiently large value. If $k > n/2$, we can just increase n to $2k$ by padding inputs. The resulting condenser will work equally well on length n inputs.

We start by stating our extractor and two condensers we will compose along with the parameters we use. Then we compose them from smallest to largest to get the final condenser.

By Lemma 39, we have a time $\text{poly}(n)$, space $O(n^2)$ invertible $(k, \epsilon/4)$ extractor

$$E : \{0, 1\}^n \times \{0, 1\}^{d_1} \rightarrow \{0, 1\}^{m_1} \times \{0, 1\}^{n+d_1-m_1}$$

with seed length $d_1 = \Theta(\log(n/\epsilon)^3)$ and $m_1 = k$. Let $n_1 = n + d_1 - m_1$ and $k_1 = d_1$.

Now to use Lemma 35, we need $26 \log(64n_1/\epsilon) \leq k_1 \leq \frac{n_1}{2}$. Well since $m_1 = k \leq n/2$ and, for large enough n , we have $k_1 = d_1 = \Theta(\log(n/\epsilon)^3) < n/2$ we have that

$$\begin{aligned} n_1 &= n + d_1 - m_1 \\ &\geq n/2 + d_1 \\ &> 2k_1. \end{aligned}$$

Similarly for large enough n , we know that $k_1 = d_1 = \Theta(\log(n/\epsilon)^3) > 26 \log(64n_1/\epsilon)$.

So by Lemma 35, there is a time $O(n^2)$ space $O(n)$ invertible $(n_1, k_1) \rightarrow_{\epsilon/4} (m_2, k_1 + d_2)$ invertible condenser

$$C_1 : \{0, 1\}^{n_1} \times \{0, 1\}^{d_2} \rightarrow \{0, 1\}^{m_2} \times \{0, 1\}^{n_1-m_2}$$

with $d_2 = O(\log(n_1 k_1 / \epsilon))$ and $m_2 \leq 2k_1 + O(\log(n_1 / \epsilon))$.

And lastly, by Corollary 38 there is a time $O(m_2^2) = O(n^2)$ space $O(m_2) = O(n)$ invertible $(m_2, k_1 + d_2) \rightarrow_{\epsilon/2} (m_3, k_1 + d_2 + d_3)$ condenser

$$C_2 : \{0, 1\}^{m_2} \times \{0, 1\}^{d_3} \rightarrow \{0, 1\}^{m_3} \times \{0, 1\}^{m_2+d_2-m_3}$$

with seed length $d = O(m_2) = O(\log(n/\epsilon)^3)$ and output length $m_3 = k_1 + d_2 + d_3 + O(\log(1/\epsilon))$.

Now to compose these condensers. First, we compose C_1 and C_2 using Theorem 32 to get a time $O(n^2)$ space $O(n)$ invertible $(n_1, k_1) \rightarrow_{\frac{3\epsilon}{4}} (m_3, k_1 + d_2 + d_3)$ invertible condenser

$$C_3 : \{0, 1\}^{n_1} \times \{0, 1\}^{d_2+d_3} \rightarrow \{0, 1\}^{m_3} \times \{0, 1\}^{n_1+d_2+d_3-m_3}.$$

Now we compose C_3 with E using Lemma 31 to get a time $\text{poly}(n)$, space $O(n^2)$ invertible $(n, k) \rightarrow_{\epsilon} (m_1 + m_3, m_1 + k_1 + d_2 + d_3)$ condenser

$$C' : \{0, 1\}^n \times \{0, 1\}^{d_1+d_2+d_3} \rightarrow \{0, 1\}^{m_1+m_3} \times \{0, 1\}^{n+d_1+d_2+d_3-m_1-m_3}.$$

Finally, see that since $k_1 = d_1$, that for $d = d_1 + d_2 + d_3$ we have that $m_1 + m_3 = k + d + O(\log(1/\epsilon))$ and the output entropy is $m_1 + d = k + d$. ◀

8 Spielman Style Weight Fixers

If one does not consider decoding, one can naturally characterize Spielman codes in terms of weight amplifiers. This perspective is useful for us, as we cannot afford to use full Spielman codes. Instead, we only use them to amplify the weight of already heavy messages and leave lighter messages to be fixed by our condenser based weight fixers.

The idea is to amplify the weight while reducing the output size. If you do this a few times, it will amplify the weight until you have a heavy string at some stage. You need to reduce the size each time so that you end up with a linear length string. Now that you know some stage of the repeated condensing has large weight, now you apply the same weight amplifier again on each stage and the outputs of the smaller stages, starting from the bottom, to pull that weight back up. This is the same thing Spielman's code does, but our analysis can be much simpler since we are not trying to decode.

Our basic tool is a weight amplifier, which is only promised to increase the weight of any light enough inputs by a constant factor. Notably, it could output zero weight for inputs that are already very large. This is necessary since we also want outputs that shrink the input.

► **Definition 40 (Weight Amplifier).** *For any $R < N$, any alphabets Σ_1 and Σ_2 composed of binary bits, and constant $\delta > 1$, a linear function $A : \Sigma_1^N \rightarrow \Sigma_2^M$ is said to be an R to δ weight amplifier if, given any x with x with $\text{weight}(x) \leq R$ then $\text{weight}(A(x)) \geq \delta \text{weight}(x)$.*

We say that R is the input weight range, δ is the relative output weight, N is the input length, and M is the output length.

The main component of our weight amplifiers are lossless expanders. The following lossless expanders are from [12, Theorem 7.1].

► **Lemma 41 (Constant Degree, Lossless Expanders).** *For some constant c , for every $N, T \leq N$ and $\epsilon > 0$, there is a D to DT regular bipartite graph $G : [N] \times [D] \rightarrow [N/T]$ that is a $(\frac{cN}{DT\epsilon}, D(1-\epsilon))$ expander where $D = \text{poly}(T/\epsilon)$.*

Further, G is invertible in time $\text{poly}(\log(N), 1/\epsilon, T)$ and space $O(\log(N) + \text{poly}(T/\epsilon))$. By invertible, we mean that for some $G' : [N] \times [D] \rightarrow [N/T] \times [DT]$, where G' restricted to its first component is G , the function G' is invertible in this time and space.

There is also an additional $2^{\text{poly}(1/\epsilon)}$ preprocessing time.

Proof. The same theorem is given in [12, Theorem 7.1], except that they use the language of conductors instead of expanders. These are equivalent just by taking the log or exponent of their parameters appropriately. All we note here is that all of the component conductors in [12] are efficiently invertible, and thus so are their compositions. See Section 7.1 for more details. ◀

Now using the lossless expanders above, we can give weight amplifiers. Our weight amplifiers, as all of our weight fixers, just output for every vertex on the right the xor of all its adjacent message bits on the left. This structure is important to know for doing fine grain analysis of the space used by a weight fixer that applies many weight amplifiers as subroutines.

► **Lemma 42** (Shrinking Weight Amplifiers Exist). *For some constant $\alpha > 0$ and constant c , for every even N , there exists an αN to 2 weight amplifier $A : \{0, 1\}^N \rightarrow \{0, 1\}^{N/2}$.*

Further, given any output bit, $i \in [N/2]$, the i th output bit of A is just the xor of at most c many input bits to A , and any of those input indices can be computed in $\text{polylog}(n)$ time and $O(\log(n))$ space.

Proof. This is given by first using Lemma 41 by setting with $\epsilon = \frac{1}{4}$ and $T = 2$ to get a function $G : [N] \times [D] \rightarrow [N/2]$ for some constant D that is an $(\alpha N, \frac{3}{4}D)$ expander for some constant $\alpha > 0$. Then A just maps every bit according to G , and takes the parity of every bit mapped to a right hand vertex. Computing this parity is efficient because G is efficiently invertible and only requires constantly many queries since D is constant.

To see that it is a weight amplifier, we first observe that since it is a $\frac{3}{4}D$ expander, we must have $D \geq 4$. Thus for any set, S , with less than αN left vertices, has at least $D/2 \geq 2$ of its neighbors with a unique neighbor in S . Therefore, the parities in these bits must be 1, and thus the output of L have at least $2|S|$ ones in it.

We note the preprocessing only takes constant time since ϵ is constant. ◀

Now one can use this shrinking weight amplifier recursively to make a constant rate, constant distance code. This is what Spielman does. But such a code is not simultaneously space and time efficient. Alternatively, we can apply this recursion a bounded number of times to get a constant rate, constant weight, weight fixer that fixes already high weight messages. Using more levels of recursion increases the range of weights that can be fixed.

► **Lemma 43** (Single Step In Spielman Style Recursion). *For the constants $\alpha > 0$ and c from Lemma 42, suppose for some N and $M \leq N$ there is an $[M, N]$ to $\alpha/4$ weight fixer $F : \{0, 1\}^N \rightarrow \{0, 1\}^{4N}$. Further, suppose that any output bit of F is an xor of at most ℓ different input bits to F , and any of these input bit indexes can be computed in time T and space S .*

Then there is an $[M/2, 2N]$ to $\alpha/4$ weight fixer $F' : \{0, 1\}^{2N} \rightarrow \{0, 1\}^{8N}$. Further any output bit of F' is an xor of at most $c^2\ell$ input bits, and any of these input bit indexes can be computed in time $T + O(\text{polylog}(N))$ and space $O(1) + \max\{S, O(\log(n))\}$.

Proof. Our weight fixer is the same as Spielman's: we take an input x , and first apply the weight amplifier of Lemma 42 to get an output y . Then we apply the weight fixer from the lemma assumption to y to get an output, z . Finally, we apply Lemma 42 to z to get the output w . Then the final output of our weight fixer is (x, z, w) . See that $|x| = 2N$, $|y| = N$, $|z| = 4N$ and $|w| = 2N$.

To see that this works, we break our problem into cases.

1. $\text{weight}(x) > 2\alpha N$. Then since x is included in the output, the relative weight of our new fixer is at least $\alpha/4$.
2. $M/2 \leq |x| \leq 2\alpha N$, then we have that $y \geq M$. Thus by the weight fixing property of F , we have that z has weight at least $\text{weight}(z) \geq \frac{\alpha}{4}|z| = \alpha N$. Then we break this into two more cases.
 - a. $\text{weight}(z) \geq 4\alpha N$. Then the relative weight of the output is at least $\alpha/2 > \alpha/4$.
 - b. $\alpha N \leq \text{weight}(z) \leq 4\alpha N$. Then $\text{weight}(w) \geq 2 \text{weight}(z) \geq \alpha 2N$. Thus the relative weight of the output is at least $\alpha/4$.

5:40 Explicit Time and Space Efficient Encoders Exist Only with Random Access

Now we show time and space needed to compute F' . Output bits from x are just input bits and can be given directly. Output bits from z are just an xor of at most ℓ different elements from y , who are themselves an xor of at most c elements of x . Thus a bit of z just an xor of $c\ell$ elements of x . Any individual bit of which can be looked up in time $T + \text{polylog}(n)$ due to the lookup time of F and Lemma 42. The space is just to either compute the neighbor in Lemma 42, or from F , which can be reused. Similarly, bits in w are just the xor of c bits of z , which are xor of $c\ell$ bits of x . So bits in w are xors of $c^2\ell$ bits of x , whose indexes be efficiently computed for the same reason. ◀

Now applying this recursion a bounded number of times, we can get a weight fixer which is time and space efficient, but only fixes the weights of messages that are already heavy. We now prove Theorem 18.

► **Theorem 18** (Weight Fixer For Heavy messages). *For some constant $\alpha > 0$, and any integers N and i , there is a $[[N/2^i], N]$ to $\alpha/4$ weight fixer $F : \{0, 1\}^N \rightarrow \{0, 1\}^{4N}$. Further, for some constant, c , any bit in the output of F can be computed in time $c^i \text{polylog}(N)$ and space $O(i + \log(N))$.*

Proof. This comes from applying Lemma 43 for i many times to the trivial weight fixer that just repeats a $K = \lceil N/2^i \rceil$ bit input 4 times. So we start with the trivial weight fixer that maps K bits to themselves that is a $[K, K]$ to 1 weight fixer.

We can see that each application of Lemma 43 increases the number of bits in the input by a factor of 2, and decreases the lower bound of the fixer range by a factor of 2, giving the claimed fixer input range and output weight.

To see the performance, see that the outputs of F must be the xor of at most c^{2i} input bits, any of which can be computed in time $i' \text{polylog}(n)$ and space $O(i + \log(n))$ by recursion. Then one need only iterate through each of the c^{2i} bits and xor them, which only takes time c^{2i} times the time to compute a single bits index, and space $O(2i)$ to store which bit we are on, plus the space to compute a single bit index. ◀

9 Encoders With Sequential Access To The Message

Now we discuss the resources needed for encoding codes in a model of computation where the program only has sequential like access to the message. This model of computation is useful for low space, black box composition of two low space algorithms. First we will show lower bounds in this model. Then we will give codes that can be encoded in nearly the same time and space as those lower bounds, proving they are tight. Finally, we will give some basic relationships between different kinds of sequential access using these upper and lower bounds.

9.1 Lower Bounds

Before we start our lower bounds, let us first clarify what we mean by space and time of an algorithm.

► **Remark 44** (Space And State Of An Algorithm). To simplify our proof here, we will refer to the space of an algorithm as the size needed to hold its entire state, including:

- Its work tape.
- All head locations.
- Number of bits written.

We assume that an algorithm always prints its output bits in order, so the number of bits written is enough to know which bit will be output next. Space does *not* include the bits printed so far, or the input bits.

If one interprets the space, S , to just be the size of the work tape, since we assumed $S \geq h \log(N)$, using our alternative definition of space only increases S by a constant factor, so our final results hold. So for simplicity, we assume S is the space needed to store its entire state.

► **Remark 45 (On Time And Uniformity).** In this specific section on sequential lower bounds, we will consider a non-uniform model of computation. We allow the algorithm with sequential access to the input to use any function to define its state transitions and head movements. The only condition is that such transitions are only functions of the working tape and whatever bits are under the heads to the input.

In particular, the time in our algorithm lower bounds is actually the number of head movements.

Our lower bounds work by partitioning the message into intervals and showing that most intervals need to be visited many times for the code to have good distance. This is because our space is bounded, so not much can be remembered about an interval when the heads leave it. So it must be visited many times for each of the different possible messages in that interval to have different things written to the code when no head is in that interval. Then the fact that our access to the message is sequential and we have few heads makes visiting an interval slow, requiring a long time to visit each interval enough times.

First, we need to formalize the idea that our algorithm can not have one of its heads enter new intervals in the message very often. But this is not true if we just count how many times the algorithm moves a head into an interval it was not in. As a counterexample, suppose a head is right next to the boundary of two intervals. Then the head can move in and out of it once every two time steps to visit it many times. In fact every interval may have been visited many times. So we need a more strict notion of visiting an interval.

So instead, we want to only count the number of times an interval transitions from having no head near it (so in one of its neighboring intervals) to having a head in it. In this setup, our algorithm really has to spend a full intervals length worth of time transitioning from having every head far from an interval to having one inside it. So to formally describe this, we introduce interval marking, where we mark an interval when a head enters it, and only unmark it when all heads are far. It requires a lot of time to mark an interval after it has been unmarked.

Before we define a marking, we need to define the distance of an interval to a head. This is defined in the obvious way.

► **Definition 46 (Distance).** For any set $S \subseteq [N]$ and any head locations $H \subseteq [N]$ we define our distance between S and H as

$$\Delta(H, S) = \min_{h \in H, s \in S} |h - s|.$$

Now we can define our interval markings for an algorithm.

► **Definition 47 (Interval Marking).** Let A be an algorithm running in time T and space S with sequential head access to the message and an interval length I .

For a length N message x , let $m = \lceil \frac{N}{I} \rceil$. Partition the message into m length I intervals: B_1, \dots, B_m where $B_i = (I(i-1), Ii]$ with $B_m = (I(m-1), n]$. A marking of the intervals is just a set $a \in \{0, 1\}^m$.

For an algorithm A on a message x , for $t \in [T]$, let H^t be the set of intervals that A running on message x has a head in at time t .

Then we inductively define a marking of A on message x . a^0 has nothing marked: $a^0 = 0^m$. At any subsequent time step t with head positions H , we define a^t by

$$a_i^t = \begin{cases} 1 & H^t \cap B_i \neq \emptyset \\ 0 & \Delta(H^t, B_i) \geq I \\ a_i^{t-1} & \text{otherwise.} \end{cases}$$

The sequence a^0, \dots, a^T is the I marking of A on message x .

We say interval B_i is covered at time t during algorithm A if $a_i^t = 1$. We say A marks interval B_i at time t if $a_i^t = 1$ but $a_i^{t-1} = 0$ and A unmarks interval B_i at time t if $a_i^t = 0$ but $a_i^{t-1} = 1$. We say there is a marking at time t if A marks any interval at time t , and there was an unmarking at time t if A unmarks any interval at time t .

We now emphasize, we use markings to refer to when an interval changes from uncovered to covered, and unmarking to when an interval changes from covered to uncovered. Marking always refers to this *change*, while cover always refers to how things *are*. For example, number of markings is how many times intervals *change* to be covered.

Now using our terminology, we can formalize our argument. First, its straightforward to observe that if there are few heads, most intervals are uncovered as no heads are near them.

► **Lemma 48** (Max Number Of Covered Intervals). *For any algorithm A running in time T , space S , and h heads, at any $t \in [T]$ the total number of intervals covered in an I marking of A on a length N message are at most $3h$.*

Proof. See that if for any i , interval B_i is only covered if $\Delta(H, B_i) < I$. For any head h A has at time t , there are only at most 3 intervals that can be within I of h . Specifically, the one that h is in and the two next to it. Thus each individual head only covers at most 3 intervals, and there are only h heads, so only $3h$ intervals can be covered. ◀

Now we show that it takes a long time to mark many intervals.

► **Lemma 49** (Max Number of Markings/Unmarkings). *For any algorithm A running in time T , space S , and h heads, any marking of A on a length N message makes at most $1 + T/I$ markings, and $1 + T/I$ unmarkings.*

Proof. The idea is that after the first step and the first marking, each interval will take time I to get through to mark the next one. Then we only unmark an interval once it will take time I to mark it again.

We only bound the number of markings, since the number of unmarkings is less than the number of markings.

We formally bound the number of markings by showing each time step can only “contribute” to marking one interval, and that every marking of an interval needs at least I “contributions” every time it is marked.

So we say any step t contributed to a marking of interval i if the head A moves at time t is in an interval adjacent to interval i and A moves that head toward interval i . We see that by this definition, A only contributes to one interval i per time step, since A can only move one head, and it either moves that head towards the interval above or below it. Let i_t be the interval that A contributes to at time t .

Suppose at any time t , the head positions of A are H^t . Then for any interval B_i see that if $\Delta(H^t, B_i) \leq I$ and $\Delta(H^{t+1}, B_i) < \Delta(H^t, B_i)$ then it must be because $i_t = i$. Otherwise the closest head to B_i must not have moved toward it. Thus for the distance to decrease beyond I , it must be due to contributions from A .

Further, if $\Delta(H^{t+1}, B_i) < \Delta(H^t, B_i)$, then $\Delta(H^{t+1}, B_i) = \Delta(H^t, B_i) - 1$ since heads can only move one position per time step. Thus to change $\Delta(H^t, B_i) \geq I$ to $\Delta(H^{t+t'}, B_i) = 0$ requires at least I contributions from A to B_i .

Finally, see that after the first marking of the first interval that the distance of every uncovered interval to a head starts out at least I . And that every interval that is unmarked also starts with distance I from any head.

Thus, after the first interval is marked, every new marking of an interval requires at least I contributions from A . And A can only contribute to one interval per time step. Thus A can only have at most $1 + T/I$ markings. Similarly, every unmarking must come from exactly one other marking, we also have at most $1 + T/I$ unmarkings. ◀

Now we can handle a special case: when the encoder is non-adaptive. A non-adaptive algorithm is an algorithm that always reads the same bits in the same order (for a given input size), regardless of the contents of those bits. As a warm up, we will prove our lower bounds for the non-adaptive case.

► **Theorem 50** (Lower Bounds For Encoders With Sequential Access). *Suppose C is a code with distance δ encoding N bits. Suppose A is a non-adaptive algorithm computing C running in time T space S and using h sequential heads to access the message. Further assume $S > h \log(N)$. Then*

$$hST = \Omega(N^2\delta).$$

Proof. The idea is that if any message interval is not marked often enough, than the state of the algorithm when that interval is not covered will look the same for two different contents of that interval. Thus we can find two messages that will give exactly the same output for any bits written while that interval is not covered. Thus if any interval is both

1. unmarked too few times and
2. uncovered when too many output bits are written

then two different messages will have encodings with little distance. We will show that most intervals must have both if $hST \ll N^2\delta$, so in particular some interval has both.

First we set the interval size to be $I = \frac{\delta N}{8h}$ so there are at least $\frac{N}{I} = \frac{8h}{\delta}$ intervals. Now we can show that most intervals are covered when at most a δ fraction of output bits are written. Since each head can only cover at most 3 intervals in a time step at any given time, at any time only at most $3h$ intervals are covered. So at most a $\frac{3\delta}{8}$ fraction of intervals are covered when any bit is written. Thus the number of intervals that are covered when at least a δ fraction of the time steps output bits are written is at most $\frac{3}{8}$.

Now we show that few intervals are unmarked frequently. Intuitively, each time an interval is unmarked, it reveals at most S bits about that interval, so we need an interval to be visited $\frac{I}{S}$ times for the entire interval to be revealed. So we want to bound the number of intervals that have been unmarked $\frac{I}{S}$ times.

We know from Lemma 49 that there are at most $1 + T/I \leq 2\frac{T}{I}$ unmarkings. Then the number of intervals that are unmarked at least $\frac{I}{S}$ times is at most $\frac{2ST}{I^2}$. Since there are at least $\frac{N}{I}$ intervals, at most $\frac{2ST}{NI} = \frac{16STh}{\delta N^2}$ fraction of intervals are unmarked more than $\frac{I}{S}$ times.

Now if we assume for contradiction that $hST < \frac{\delta N^2}{32}$, we have that at most half of the intervals are unmarked more than $\frac{I}{5}$ times. Since at most a $\frac{3}{8}$ fraction of the intervals have at least a δ fraction of output bits written when they are covered, that means that a $\frac{1}{8}$ fraction of intervals are both

1. unmarked less than $\frac{I}{5}$ times and
2. covered when less than a δ fraction of output bits are written.

Take one such interval (which is not the final interval, so has length I), call it interval B_i .

Now we will create two adversarial messages for the algorithm that will not have good distance for the code. We do this by showing that two messages look the same when interval B_i is not covered. Let x^* be the restriction that sets everything outside interval B_i to zero.

Now for any assignment, y , to interval B_i , define $R(y)$ to be the tuple of the states of the algorithm on input $x_y = y \circ x^*$ every time B_i is unmarked. Since B_i is unmarked less than $\frac{I}{5}$ times, the length of $R(y)$ is less than I . Thus for two different y , call them y_1 and y_2 , we have that $R(y_1) = R(y_2)$. Then for input $x_1 = y_1 \circ x^*$ and $x_2 = y_2 \circ x^*$, we have that A acts on x_1 and x_2 exactly the same when B_i is uncovered. Then since less than a δ fraction of output bits are written when B_i is covered and A has the same output for both when B_i is uncovered, $C(x_1)$ can only differ from $C(x_2)$ on at most a δ fraction of codeword bits.

Thus C has distance less than δ . Contradiction, so we must have $hST \geq \frac{\delta N^2}{32} = \Omega(\delta N^2)$. \blacktriangleleft

The argument becomes a bit more complex when we allow the algorithm to be adaptive. Specifically, which intervals are marked frequently may change depending on the message, as are the intervals that are uncovered frequently. Specifically, our adversarial inputs set everything outside of one interval to 0. If the algorithm knows that everything outside one interval will be zero, than it can detect which interval is non-zero and spend all its time on that interval.

To get around these issues, we choose messages randomly. For any random message, for a random interval, with high probability that interval is only covered when a small fraction of the output bits are written, and that interval is unmarked few times. Equivalently, we can select a random interval, randomly restrict everything outside that interval, than randomly assign that interval. These sample the same distribution, so we also have that for most random intervals chosen, B_i , and most restrictions outside B_i , it must be that for most assignments to B_i we have that B_i is unmarked few times and uncovered when most output bits are written.

So we define a good restriction to be such a restriction, and then show that good restrictions give distinct messages that have too close code words. Finally we show good restrictions exist when $hST \ll \delta N^2$.

► Definition 51 (A Good Restriction of the Message). *For any algorithm A with time T and space S , given distance δ and interval length I , we say a restriction x^* is good for interval i with distance δ on algorithm A if the following holds.*

1. x^* fixes every bit outside of the interval B_i , and leaves every bit inside B_i unfixed.
2. Let Y be the set of assignments for B_i such that for $x = y \circ x^*$ we have:
 - a. A on message x writes at most δ fraction of its bits when B_i is covered.
 - b. A on message x unmarks B_i at most $\frac{8T}{N}$ times.

We also require that $|Y| > 2^I/2$.

Now we show that a good restriction is enough to give our lower bounds.

► **Lemma 52** (Good Restrictions Give Lower Bounds). *Suppose A is an algorithm with time T , space S , and h sequential heads to the message. If x^* is good for interval i with interval size I and distance δ , then whatever code A outputs has distance at most δ if*

$$8ST < N(I - 1).$$

Proof. The idea is just to consider the Y from Definition 51. For the x that come from Y , we have that B_i is unmarked rarely, so rarely in fact that multiple x must have the exact same states every time B_i is unmarked. Since the state includes the positions of the heads, they must write the same thing when the interval is uncovered. Since the state includes the number of bits written, they must be written at the same location when B_i is uncovered. Thus the distance between these messages is at most what is written when B_i is covered, which is at most a δ fraction of bits.

More rigorously, for any $y \in Y$, let $x_y = y \circ x^*$. By definition, A on message x_y marks B_i at most $\frac{8T}{N}$ times. Then define $R(y)$ to be the tuple of the state of A at every step B_i is uncovered when running on message x_y . Then since the state only has S bits and B_i is uncovered at most $\frac{8T}{N}$ times, we have

$$|R(y)| \leq S \frac{8T}{N}.$$

Now see that since $|R(y)| \leq 8 \frac{ST}{N}$, then the total number of distinct values for $R(y)$ is at most $2^{8 \frac{ST}{N}}$. Since the restriction is good, the total number of distinct $y \in Y$ is at least $2^I/2$. Finally, by lemma premise, we have that $8ST < N(I - 1)$. Thus we can show that

$$\begin{aligned} 8ST &< N(I - 1) \\ 8 \frac{ST}{N} &< I - 1 \\ 2^{8S \frac{T}{N}} &< 2^I/2 \end{aligned}$$

Thus the number of distinct $R(y)$ is less than the number of distinct y , so by pigeonhole principle, there must be $y_1, y_2 \in Y$ such that $R(y_1) = R(y_2)$.

Now by definition of $R(y)$, when running A on message x_y we must have that everything written when interval B_i is uncovered is dependent only on $R(y)$ and x^* . In particular, $R(y_1)$ and $R(y_2)$ write the same bits when B_i is uncovered, at the same places.

Since $y_1, y_2 \in Y$, we have that at most δ fraction of the bits are written when B_i is covered, and these are the only bits where they can differ. So the distance between the outputs of A on x_{y_1} and x_{y_2} is at most δ . ◀

It remains to show that there must be some good restriction if I is chosen well. Specifically when $I < \frac{N\delta}{24h}$.

► **Lemma 53** (Good Restrictions Exist). *Suppose A is an algorithm with time T , space S , and h sequential heads to access the message.*

Then if $I < \frac{N\delta}{24h} < N < T$ then there is some restriction which is good for some interval i with distance δ on algorithm A .

Proof. Suppose by way of contradiction that every restriction is not good for any interval with distance δ on algorithm A . The idea is to show that, in expectation, either more than $3h$ of intervals are covered whenever a bit is written (contradicting Lemma 48) or more than $1 + T/I$ intervals are marked (contradicting Lemma 49).

5:46 Explicit Time and Space Efficient Encoders Exist Only with Random Access

So choose a random interval, i , a random restriction x^* for every variable outside B_i . We assumed that x^* is bad, so with probability at least $1/2$ for a random assignment, y , of the variables in B_i will $y \notin Y$ for the Y defined in Definition 51. That is, with probability at least $1/2$ will we have for $x_y = y \circ x^*$ that algorithm A on message x_y will unmark interval i more than $\frac{8T}{N}$ times or will be covered when at least δ fraction of the output bits written.

See that x_y is uniformly randomly distributed and so is i , and these are independent of each other. Thus in expectation over a randomly chosen interval, B_i , and a randomly chosen input, x , we have that with probability at least $1/2$ algorithm A on input x either unmarks interval B_i more than $\frac{8T}{N}$ times or interval B_i be covered when at least a δ fraction of the output bits written. Now we show that x cannot have enough intervals covered or unmarked to achieve this.

By Lemma 49, the total number of markings is at most $T/I + 1 < 2T/I$. So at most $\frac{N}{4I}$ intervals are marked more than $\frac{8T}{N}$ times. There are at least N/I intervals, so only at most $\frac{1}{4}$ fraction of the intervals can be marked more than $\frac{8T}{N}$ times.

By Lemma 48, at any given time, at most $3h$ intervals can be covered. So the probability that a random interval is covered is at most

$$\begin{aligned} \frac{3h}{N/I} &= \frac{3hI}{N} \\ &< \frac{3hN\delta}{N24h} \\ &= \frac{\delta}{8}. \end{aligned}$$

Then by a Markov inequality, the probability that a random interval is covered greater than a δ fraction of the times an output bit is written is at most $\frac{1}{8}$.

Thus by a union bound, the total of fraction of intervals that are either unmarked more than $\frac{8T}{N}$ times or covered for at least δ fraction of the times an output bit is written is $\frac{3}{8} < \frac{1}{2}$. But by choice of x , these must occur for at least half i . Contradiction. So some restriction and interval must be good with distance δ . ◀

Now that we know good restrictions exist, and good restrictions imply our lower bounds, we can prove Theorem 2.

► **Theorem 2** (Lower Bounds For Encoders With Sequential Access). *Suppose C is a code with relative distance δ encoding N bits. Suppose A is an algorithm computing C running in time T space S and using h sequential heads to access the message. Further assume $S > h \log(N)$. Then*

$$hST = \Omega(\delta N^2).$$

Proof. Let $I = \frac{N\delta}{50h} < \frac{N(\delta/2)}{24h}$. Then by Lemma 53, a good restriction with distance $\delta/2$ exists. Then by Lemma 52, since C has distance greater than $\delta/2$, it must be the case that $32ST \geq NI$. Thus

$$\begin{aligned} 8ST &\geq N(I - 1) \\ 16ST &\geq NI \\ &= N \frac{N\delta}{50h} \\ 800STh &\geq N^2\delta \\ hST &= \Omega(n^2\delta). \end{aligned}$$

◀

9.2 Upper Bounds

The codes that achieve our upper bounds are just a tensor code. There is a natural way to compute tensor codes in a space efficient way when one has a limited number of sequential heads to access the message. By choosing one of the codes to be a code with time and space efficient encoders using random access to its message, one can get a smooth trade off between the time and space required to encode a code.

► **Lemma 54** (Tensor Codes Are Efficient To Compute). *Suppose there is a code $C_1 : \{0, 1\}^{N_1} \rightarrow \Sigma_1^{M_1}$ with relative distance δ_1 computable in time T_1 and space S_1 where $N_1 \leq M_1$ and $\Sigma_1 = \{0, 1\}^{\ell_1}$. Suppose there is another code $C_2 : \{0, 1\}^{N_2} \rightarrow \Sigma_2^{M_2}$ with relative distance δ_2 computable in time T_2 and space S_2 with non-adaptive, random access to its message where $N_2 \leq M_2$ and $\Sigma_2 = \{0, 1\}^{\ell_2}$.*

Denote the tensor code of C_1 and C_2 as $C : \{0, 1\}^{N=N_1N_2} \rightarrow \Sigma^{M=M_1M_2}$, where $\Sigma = \{0, 1\}^{\ell_1\ell_2}$. That is, C , is a tensor code on binary symbols, but we then group into the output bits into ℓ_1 by ℓ_2 squares. Then C has relative distance $\delta_1\delta_2$.

Further, for any number of heads $h \geq 2$, there is an algorithm that computes C and runs in time $O(T_2(\frac{N}{h} + T_1))$ and space $O(S_1 + S_2M_1\ell_1)$ using at most h non-reversible sequential heads to access the message.

Proof. In a tensor code, the message is arranged as a table, table 1, with N_2 rows, each containing N_1 columns. The tensor code is the code defined by first encoding each of the rows of table 1 with C_1 to get a new table, table 2, with N_2 rows and $M_1\ell_1$ columns. Then encode the columns of table 2 with C_2 to get table 3 with $M_2\ell_2$ rows and $M_1\ell_1$ columns. Finally, we group table 3 into ℓ_1 by ℓ_2 cells to get the final symbols of our final codeword. We assume the message is arranged by row, with all the first N_1 bits being row 1, the next N_1 bits being row 2, and so on.

Then our encoder first distributes $h - 1$ heads evenly between the N_2 rows. Then we simulate C_2 in parallel for each of the $M_1\ell_1$ columns, and every time they need to query a bit from a row that has been encoded by C_1 , we just move the nearest head to that row, encode it by C_1 , and then give that row to each of the $M_1\ell_1$ parallel computations of C_2 .

The time needed for this is just the time to encode C_2 the $M_1\ell \leq T_1$ times, plus the number of rows that are queried (trivially bounded by T_1) times the time to move a head to that row (bounded by $\frac{N_2N_1}{h-1} = O(\frac{N}{h})$) and the time to encode that row T_1 . This takes time at most

$$T_2M_1\ell + T_2(\frac{N_2N_1}{h-1} + T_1) = O(T_2(T_1 + \frac{N}{h})).$$

The space is just the space to compute C_1 , plus the space to store the output of C_1 , plus the space to hold $M_1\ell_1$ copies of S_2 's algorithm. This is just space $O(S_1 + S_2M_1\ell_1)$. ◀

Now applying this tensor code with any good code and our explicit time space efficient codes shows that our lower bound on algorithms with sequential like access to the message is almost tight. We now prove Theorem 3.

► **Theorem 3** (Encoders With Sequential Access Meeting the Lower Bounds). *For any number of heads $h \geq 2$, time $T \geq N$, space $S = \Omega(h \log(N))$, relative distance $\delta > 0$ with $hST = \Omega(\delta N^2)$, there exists a code with constant rate and relative distance $\Omega(\delta)$ encoded by a time $TN^{o(1)}$, space $S \text{ polylog}(N)$ algorithm using h sequential heads to access the message.*

Proof. To make things simple, we assume that $4S$ divides δN and $N' = \delta N$ is an integer. If not, pad S and N and decrease δ until this is true. This can be done only changing constant factors, for instance by making each a power of 2.

Since we only need relative distance $\Omega(\delta)$, we just use an efficient, constant relative distance code on $1/\delta$ sets of δN message bits. Since all of our codes are linear, the min weight codeword must have weight $\Omega(\delta N)$, or relative weight $\Omega(\delta)$. Thus final code will have relative distance $\Omega(\delta)$.

Let our first code be the Spielman code $C_1 : \{0, 1\}^S \rightarrow \{0, 1\}^{4S}$, which has some constant relative distance, $\delta_1 > 0$, and is encodable in linear time and linear space. Then using Theorem 1, there exists a linear code

$$C_2 : \{0, 1\}^{N'/4S} \rightarrow \{0, 1\}^{M'}$$

that has some constant relative distance $\delta_2 > 0$, and output length $M' = O(N'/S) = O(\delta N/S)$. Note to get binary alphabet, we just output each bit of Σ individually. This hurts distance, but only by a constant factor. Also, C is computable in time $\frac{N'}{S} \text{poly}(2^{\log(\log(N'))^3}) = \frac{\delta N}{S} N^{o(1)}$ and space $O(\log(N)^2)$.

Now applying Lemma 54 with C_1 and C_2 , we get a tensor code

$$C : \{0, 1\}^N \rightarrow \{0, 1\}^{M'4S}$$

with constant relative distance $\delta_1 \delta_2 > 0$ and output length $M'4S = O(\delta N)$. Further, C can be computed in time

$$O\left(\frac{\delta N}{S} N^{o(1)} \left(\frac{\delta N}{h} + S\right)\right) = \frac{\delta^2 N^2}{Sh} N^{o(1)} + \delta N N^{o(1)}$$

and space

$$O(S + \log(N)^2 S) = S \text{polylog}(N)$$

using only h sequential heads to access the message.

Now to compute the final code, we need only repeat this encoding procedure $1/\delta$ times, which uses the same space and same number of heads, but takes $1/\delta$ times longer to get final time of

$$\frac{\delta N^2}{Sh} N^{o(1)} + N N^{o(1)} = O(TN^{o(1)}). \quad \blacktriangleleft$$

9.3 Basic Relations of Sequential Access

In this section, we establish some relationships between variations of sequential access. We show whether reversibility or jumping adds power to time and space bounded algorithms with sequential access to the input.

While reversibility, being able to move heads backward, may seem powerful, it can actually be simulated with non-reversible, jumping heads with only a logarithmic factor overhead. We now prove Lemma 7.

► **Lemma 7** (Reversibility Can Be Efficiently Simulated With Jumping). *A single sequential head to a length N input can be simulated with $O(\log(N))$ non-reversible sequential heads to the same input with an expected time of $O(\log(N))$ for each head movement, and $O(\log(N))$ space.*

More generally, k sequential heads to a length N input can be simulated with $O(k \log(N))$ non-reversible sequential heads to that same input with an expected time of $O(\log(N))$ for each head movement, and $O(k \log(N))$ space.

Proof. The idea is to store one to two non-reversible heads at every order of magnitude behind the reversible head being simulated. By being appropriately lazy with removing heads and adding them as the simulated head moves, this can be done with only $O(\log(N))$ heads and only $O(\log(N))$ time overhead. The extra $O(\log(N))$ bits of space are just to remember where the heads are.

More specifically, there are i levels and each responsible for its own order of magnitude. That order of magnitude only ever removes heads that are far from the simulated head, relative to its order of magnitude. So when a head is removed, the simulated head must use a lot of time to force it to be added again. This requires only storing at most two heads per order of magnitude.

Level i always stores heads that are 2^i distance apart, can either store 0, 1, or 2 heads, and these heads are always at the multiples of 2^i immediately behind the head being simulated. A new head is added to a level whenever the simulated head passes forward over a multiple of 2^i , and a head is removed if either the simulated head passes over it when moving backward, or when the level has more than 2 heads, in which case the first head is removed.

Now the trick is to move a simulated head left, instead of moving a head left (which is not allowed), we instead have the head jump to whichever level, i , has the nearest head behind it, and simulate it forward till it reaches one position before where it was before.

The space and number of heads is clear from the construction.

For time, we note that a left move can only trigger level i once every 2^{i-1} steps. To see this, first we claim that level i is only triggered when the simulated head is at a multiple of 2^{i-1} minus 1. This is because for every j , level j 's earliest head is earlier than or equal to all of level $j-1$'s heads. This is straightforward from the construction as level $j-1$ removes heads before level j . Thus the last head that was passed before triggering level i must have been in level $i-1$.

After level i was triggered, there is a head in level $i-1$ at least $2^{i-1}-1$ before the simulated head. And we note that moving forward can never decrease the distance to the furthest head in level $i-1$ to below 2^{i-1} . Thus to trigger level i after a move would require 2^{i-1} moves.

Each level, i , takes only time $O(2^i)$ when it is triggered, and it is triggered at most once every $\frac{1}{2^{i-1}}$ steps. Thus each level only costs an expected constant time per simulated head movement. There are only $\log(N)$ levels. Thus the expected time per head movement is $O(\log(N))$.

If there are multiple heads being simulated, this actually just makes the problem easier as it increases our budget. When we move backward, we just use whichever head is closest from any of our simulated heads, and we don't remove any head at level i if it is within 2^{i+1} steps behind any simulated head. Jumps are cheap since we can share heads between the simulated head being jumped and the simulated head it is jumping to. In particular, you can't adversarially jump simulated heads to right before expensive backward operations as the resulting new heads will be shared with the simulated head that just jumped.

To make argument more formal in the multiple head case, we can count the time needed to trigger a particular head at a particular location, call it j , in level i . A head can only be triggered if there is a simulated head in front of it, and there is no head in a lower level at the same location. If the closest head in level $i-1$ was 2^{i-1} in front of j , then there must have been no simulated head within 2^i of j at some point, otherwise level $i-1$ would have kept a head at location j . Thus it would have taken 2^{i-1} steps to move a head backwards enough to trigger level i at location j . If the closest head in level $i-1$ was 2^i in front of j , there must have been no simulated head within $3 \cdot 2^{i-1}$ of j for a similar reason, and thus a simulated

head must have moved 2^{i-1} times to be in position to trigger the head in level i at j . We call these backward movements of simulated heads within the interval $[j + 2^{i-1}, j + 3 \cdot 2^{i-1})$ as being dedicated to the head at j in level i .

So each position j at a level i takes at least 2^{i-1} backward movements dedicated to it specifically to trigger that head. From there, a similar argument holds as the single head case. ◀

Thus the reversible and non-reversible input are equivalent up to log factors. A similar phenomenon happens with reversible and non reversible computation. Bennett [9] proved that any time T space S non-reversible computation can be turned into a reversible algorithm running in time $T^{1+\epsilon}$ and space $O(S \log(T))$ for any constant $\epsilon > 0$.

On the other hand, the jumping feature can not be efficiently simulated with non-jumping heads. It provably requires polynomially more time or space to solve some problems with non-jumping heads than jumping ones. One reason for this is that jumping can make it very efficient to move many heads far distances at once by moving one head, then jumping the others. The following result assumes our upper and lower bounds on codes in Theorem 2 and Theorem 3.

► **Lemma 55 (Jumping Can Not Be Efficiently Simulated With Reversibility).** *There exists a problem solvable in time $N^{1+o(1)}$ and space $N^{1/2+o(1)}$ with $O(\sqrt{N})$ jumping, non-reversible sequential heads to access the input, but for any constant ϵ can not be solved in time $o(N^{5/4-\epsilon})$ and space $o(N^{1/2+2\epsilon})$ with any number of non-jumping, reversible sequential heads to access the input.*

Proof. The problem is to just encode the second half of the input with the code of Theorem 3 using constant distance. This can be done easily with jumping, non-reversible heads in the time, space, and number of heads stated.

Now by Theorem 2, to encode this second half in time T and space S would require a number of heads that is at least

$$h \geq \Omega\left(\frac{N^2}{ST}\right).$$

But to actually use any of these heads, they must first pass the first half of the input. Since we have to move the heads one at a time, this takes time

$$\begin{aligned} T &\geq \Omega(hN) \\ &\geq \Omega\left(\frac{N^3}{ST}\right) \\ ST^2 &\geq \Omega(N^3) \end{aligned}$$

But suppose for way of contradiction that we had such a time $T = o(N^{5/4-\epsilon})$ and space $S = o(N^{1/2+2\epsilon})$ algorithm with non-jumping sequential access to the input. Then we have that

$$\begin{aligned} ST^2 &\leq o(N^{1/2+2\epsilon+2(5/4-\epsilon)}) \\ &= o(N^3) \end{aligned}$$

which is a contradiction. ◀

Taking $\epsilon = \frac{1}{8}$, we see that the algorithm with non-jumping heads gets both polynomially more time and polynomially more space than the algorithm with jumping heads, but still cannot compute the code.

► **Remark 56 (Non-Jumping Heads With Preprocessing).** The lower bounds of Lemma 55 depend on the fact that it takes a long time to perform initial positioning of heads. This lower bound no longer holds if we allow all the heads to be positioned in the second half of the input before the algorithm starts.

But we can still get something even if we allow the algorithm to use any initial positioning of heads that does not depend on the input. The idea is essentially the same, encode some specific subset of the bits in a code, but have the specific set of bits to be encoded be part of the input. We quickly sketch a proof outline here.

To encode a random interval with $N^{3/4}$ fraction of message can be done by Theorem 3 with $h = S = N^{1/4}$ in time around $\frac{(N^{3/4})^2}{Sh} = N$ with sequential access to the input. Now consider an algorithm with non-jumping sequential access to the input with the same space and number of heads, with some initial, static positioning of the heads. With high probability, only a constant number of heads will be in the interval, so to encode it fast, our encoder needs to move more heads to that interval.

To move K heads to the interval to encode, in expectation, would take time around $\frac{NK}{h}K = N^{3/4}K^2$. Then by Theorem 2, to encode with these K heads would take time $\frac{(N^{3/4})^2}{SK} = \frac{N^{5/4}}{K}$. No matter the setting of K , this encoding will take time $\Omega(N^{13/12})$.

Thus jumping sequential access still has advantage over non-jumping sequential access even if a non-jumping algorithm is allowed to move heads into an initial position for free before it starts reading.

So we have proven that jumping heads are polynomially more powerful than non-jumping heads, and that jumping, non-reversible heads are within a log factor as powerful as jumping, reversible heads. Finally, one can show that non-jumping, reversible sequential heads are polynomially more powerful than non-jumping, non-reversible sequential heads. This is a direct consequence of Theorem 2.

► **Lemma 57 (Without Jumps, Reversible Heads Are More Powerful than Non-Reversible Heads).** *Encoding codes with relative distance δ with non-jumping, non-reversible heads requires space S and number of heads h such that*

$$h^2 S = \Omega(\delta N).$$

Proof. This follows from the fact that Theorem 2 is actually a bound on the number of head movements. Since heads without backwards movements or jumps cannot move backward, there are only hN movements before all input heads are at the end of input. Thus $hN \geq T$ in Theorem 2. This gives that

$$h^2 S \geq \frac{hST}{N} = \Omega(\delta N). \quad \blacktriangleleft$$

In particular, without jumps or backwards moves, one cannot compute an asymptotically good code with only $S = n^{1/3}$ space and only $h = o(n^{1/3})$ heads, whereas with either jumping or reversibility², this can be achieved by using $N^{4/3+o(1)}$ time by Theorem 3.

² This uses the fact that the only time jumping is used in Theorem 3 is for efficient, initial positioning. Initial positioning of h heads only takes $O(hN)$ time without jumping.

References

- 1 Karl Abrahamson. Time-space tradeoffs for algebraic problems on general sequential machines. *Journal of Computer and System Sciences*, 43(2):269–289, 1991. doi:10.1016/0022-0000(91)90014-V.
- 2 N. Alon, J. Bruck, J. Naor, M. Naor, and R.M. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38(2):509–516, 1992. doi:10.1109/18.119713.
- 3 Laasya Bangalore, Rishabh Bhaduria, Carmit Hazay, and Muthuramakrishnan Venkatasubramanian. On black-box constructions of time and space efficient sublinear arguments from symmetric-key primitives. In *Theory of Cryptography: 20th International Conference, TCC 2022, Chicago, IL, USA, November 7–10, 2022, Proceedings, Part I*, pages 417–446. Springer-Verlag, 2022. doi:10.1007/978-3-031-22318-1_15.
- 4 L.M.J. Bazzi and S.K. Mitter. Endcoding complexity versus minimum distance. *IEEE Transactions on Information Theory*, 51(6):2103–2112, 2005. doi:10.1109/TIT.2005.847727.
- 5 P. Beame. A general sequential time-space tradeoff for finding unique elements. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, STOC '89*, pages 197–203, New York, NY, USA, 1989. Association for Computing Machinery. doi:10.1145/73007.73026.
- 6 Paul Beame and Niels Kornerup. Cumulative memory lower bounds for randomized and quantum computation. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.17.
- 7 Paul Beame, Michael Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003. doi:10.1145/636865.636867.
- 8 Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, STOC '13*, pages 585–594. Association for Computing Machinery, 2013. doi:10.1145/2488608.2488681.
- 9 Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM J. Comput.*, 18(4):766–776, 1989. doi:10.1137/0218053.
- 10 Nir Bitansky and Alessandro Chiesa. Succinct arguments from multi-prover interactive proofs and their efficiency benefits. In *Proceedings of the 32nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417*, pages 255–272, Berlin, Heidelberg, 2012. Springer-Verlag. doi:10.1007/978-3-642-32009-5_16.
- 11 A. Borodin and S. Cook. A time-space tradeoff for sorting on a general sequential model of computation. *SIAM J. Comput.*, 11(2):287–297, 1982. doi:10.1137/0211022.
- 12 M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings 17th IEEE Annual Conference on Computational Complexity*, pages 8–8, 2002. doi:10.1109/CCC.2002.1004327.
- 13 Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing, STOC '16*, pages 285–298. Association for Computing Machinery, 2016. doi:10.1145/2897518.2897547.
- 14 M. Cheraghchi, F. Didier, and A. Shokrollahi. Invertible extractors and wiretap protocols. *IEEE Trans. Inf. Theor.*, 58(2):1254–1274, 2012. doi:10.1109/TIT.2011.2170660.
- 15 Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. *J. Cryptol.*, 30(1):191–241, 2017. doi:10.1007/s00145-015-9219-z.
- 16 Joshua Cook and Dana Moshkovitz. Time and space efficient deterministic decoders. *Electronic Colloquium on Computational Complexity*, 2024. URL: <https://eccc.weizmann.ac.il/report/2024/110/>.
- 17 Yotam Dikstein, Irit Dinur, and Shiri Sivan. The linear time encoding scheme fails to encode, 2023. arXiv:2312.16125.




- 18 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*, STOC 2022, pages 357–374. Association for Computing Machinery, 2022. doi:10.1145/3519935.3520024.
- 19 Dariush Divsalar, Hui Jin, and Robert J. McEliece. Coding theorems for ‘turbo-like’ codes. In *Proceedings 36th Annual Allerton Conference on Communication, Control, and Computing*, pages 201–210, 1998. URL: <https://api.semanticscholar.org/CorpusID:1045655>.
- 20 R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1):21–28, 1962. doi:10.1109/TIT.1962.1057683.
- 21 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *J. ACM*, 62(4), September 2015. doi:10.1145/2699436.
- 22 André Gronemeier. A note on the decoding complexity of error-correcting codes. *Inf. Process. Lett.*, 100(3):116–119, 2006. doi:10.1016/j.ipl.2006.06.006.
- 23 V. Guruswami and P. Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005. doi:10.1109/TIT.2005.855587.
- 24 Venkatesan Guruswami and Widad Machmouchi. Explicit interleavers for a repeat accumulate (raa) code construction. In *2008 IEEE International Symposium on Information Theory*, pages 1968–1972, 2008. doi:10.1109/ISIT.2008.4595333.
- 25 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC’07)*, pages 96–108, 2007. doi:10.1109/CCC.2007.38.
- 26 Anna Gál, Kristoffer Arnsfelt Hansen, Michal Koucký, Pavel Pudlák, and Emanuele Viola. Tight bounds on computing error-correcting codes by bounded-depth circuits with arbitrary gates. *IEEE Transactions on Information Theory*, 59(10):6611–6627, 2013. doi:10.1109/TIT.2013.2270275.
- 27 R. W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, 1950. doi:10.1002/j.1538-7305.1950.tb00463.x.
- 28 Tzvikia Hartman and Ran Raz. On the distribution of the number of roots of polynomials and explicit weak designs. *Random Struct. Algorithms*, 23(3):235–263, October 2003. doi:10.1002/rsa.10095.
- 29 F.C. Hennie. One-tape, off-line turing machine computations. *Information and Control*, 8(6):553–578, 1965. doi:10.1016/S0019-9958(65)90399-2.
- 30 Justin Holmgren and Ron Rothblum. Delegating computations with (almost) minimal time and space overhead. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 124–135, 2018. doi:10.1109/FOCS.2018.00021.
- 31 R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC ’89, pages 12–24. Association for Computing Machinery, 1989. doi:10.1145/73007.73009.
- 32 S. Jukna. A nondeterministic space-time tradeoff for linear codes. *Information Processing Letters*, 109(5):286–289, 2009. doi:10.1016/j.ipl.2008.11.001.
- 33 Itay Kalev and Amnon Ta-Shma. Unbalanced Expanders from Multiplicity Codes. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*, volume 245 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:14, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.12.
- 34 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC ’16, pages 202–215. Association for Computing Machinery, 2016. doi:10.1145/2897518.2897523.
- 35 Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *J. ACM*, 61(5), 2014. doi:10.1145/2629416.

- 36 S. Rao Kosaraju. Real-time simulation of concatenable double-ended queues by double-ended queues (preliminary version). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 346–351. Association for Computing Machinery, 1979. doi:10.1145/800135.804427.
- 37 Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 1144–1156. Association for Computing Machinery, 2017. doi:10.1145/3055399.3055486.
- 38 D. J. C. Mackay and Radford M. Neal. Near shannon limit performance of low density parity check codes. *Electronics Letters*, 33:457–458, 1996. URL: <https://api.semanticscholar.org/CorpusID:122801915>.
- 39 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996. doi:10.1006/jcss.1996.0004.
- 40 Wolfgang J. Paul, Joel I. Seiferas, and Janos Simon. An information-theoretic approach to time bounds for on-line computation (preliminary version). In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, STOC '80, pages 357–367. Association for Computing Machinery, 1980. doi:10.1145/800141.804685.
- 41 Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in trevisan's extractors. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, STOC '99, pages 149–158. Association for Computing Machinery, 1999. doi:10.1145/301250.301292.
- 42 I.S. Reed. A brief history of the development of error correcting codes. *Computers & Mathematics with Applications*, 39(11):89–93, 2000. doi:10.1016/S0898-1221(00)00112-7.
- 43 O. Reingold, R. Shaltiel, and A. Wigderson. Extracting randomness via repeated condensing. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 22–31, 2000. doi:10.1109/SFCS.2000.892008.
- 44 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*, STOC '16, pages 49–62. Association for Computing Machinery, 2016. doi:10.1145/2897518.2897652.
- 45 Nandakishore Santhi and Alexander Vardy. Minimum distance of codes and their branching program complexity. In *2006 IEEE International Symposium on Information Theory*, pages 1490–1494, 2006. doi:10.1109/ISIT.2006.262116.
- 46 Walter J. Savitch and Paul M. B. Vitányi. Linear time simulation of multihead turing machines with head-to-head jumps. In *Proceedings of the Fourth Colloquium on Automata, Languages and Programming*, pages 453–464, Berlin, Heidelberg, 1977. Springer-Verlag.
- 47 M. Sipser and D.A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996. doi:10.1109/18.556667.
- 48 D.A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996. doi:10.1109/18.556668.
- 49 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '95, pages 388–397. Association for Computing Machinery, 1995. doi:10.1145/225058.225165.
- 50 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 143–152. Association for Computing Machinery, 2001. doi:10.1145/380752.380790.
- 51 Luca Trevisan. Construction of extractors using pseudo-random generators (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, STOC '99, pages 141–148. Association for Computing Machinery, 1999. doi:10.1145/301250.301289.
- 52 Yaacov Yesha. Time-space tradeoffs for matrix multiplication and the discrete fourier transform on any general sequential random-access computer. *Journal of Computer and System Sciences*, 29(2):183–197, 1984. doi:10.1016/0022-0000(84)90029-1.

The Entangled Quantum Polynomial Hierarchy Collapses

Sabee Grewal   

The University of Texas at Austin, TX, USA

Justin Yirka   

The University of Texas at Austin, TX, USA

Abstract

We introduce the entangled quantum polynomial hierarchy, QEPH, as the class of problems that are efficiently verifiable given alternating quantum proofs that may be entangled with each other. We prove QEPH collapses to its second level. In fact, we show that a polynomial number of alternations collapses to just two. As a consequence, $\text{QEPH} = \text{QRG}(1)$, the class of problems having one-turn quantum refereed games, which is known to be contained in PSPACE. This is in contrast to the *unentangled* quantum polynomial hierarchy, QPH, which contains QMA(2).

We also introduce DistributionQCPH, a generalization of the quantum-classical polynomial hierarchy QCPH where the provers send probability distributions over strings (instead of strings). We prove $\text{DistributionQCPH} = \text{QCPH}$, suggesting that only quantum superposition (not classical probability) increases the computational power of these hierarchies. To prove this equality, we generalize a game-theoretic result of Lipton and Young (1994) which says that, without loss of generality, the provers can send uniform distributions over a polynomial-size support. We also prove the analogous result for the polynomial hierarchy, i.e., $\text{DistributionPH} = \text{PH}$.

Finally, we show that PH and QCPH are contained in QPH, resolving an open question of Gharibian et al. (2022).

2012 ACM Subject Classification Theory of computation \rightarrow Interactive proof systems; Theory of computation \rightarrow Complexity classes; Theory of computation \rightarrow Quantum complexity theory

Keywords and phrases Polynomial hierarchy, Entangled proofs, Correlated proofs, Minimax

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.6

Related Version *Previous Version:* <https://arxiv.org/abs/2401.01453v1>

Funding Supported via Scott Aaronson by a Vannevar Bush Fellowship from the US Department of Defense, the NSF QLCI program (Grant No. OMA-2016245), and a Simons Investigator Award, the Simons “It from Qubit” collaboration. This material is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Quantum Systems Accelerator.

Acknowledgements We thank Khang Le, Daniel Liang, William Kretschmer, Siddhartha Jain, and Scott Aaronson for helpful conversations. Joshua Cook was especially helpful at early stages of this project. We thank John Watrous for identifying an error in an earlier draft of this work.

1 Introduction

The polynomial hierarchy [26, 31] is a hierarchy of complexity classes that are known to equal P if and only if $P = NP$. The hierarchy, denoted by PH, is a natural generalization of efficient proof verification and nondeterminism and plays a central role in complexity theory. Given its significance, it is natural to explore quantum generalizations of PH, yet such generalizations remain understudied.

Before discussing quantum polynomial hierarchies, let us first informally define PH. Intuitively, PH is a hierarchy of complexity classes that can solve progressively harder problems, extending beyond both NP and coNP. One can think of PH as a public debate



© Sabee Grewal and Justin Yirka;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 6; pp. 6:1–6:23

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



between Alice and Bob, who take turns presenting polynomial-sized proofs (bit strings) to a referee. At the end of the debate, the referee takes the proofs, performs a polynomial-time classical computation, and decides a winner.

More formally, a problem is in the k -th level of the polynomial hierarchy, Σ_k^P , if there is a deterministic polynomial-time verifier M (the referee) that takes proofs y_1, \dots, y_k and satisfies the following conditions. On yes-instances, $\exists y_1 \forall y_2 \exists y_3 \dots$ such that $M(y_1, \dots, y_k) = 1$, and, on no-instances, $\forall y_1 \exists y_2 \forall y_3 \dots$ such that $M(y_1, \dots, y_k) = 0$. PH is comprised of every level Σ_k^P for all natural numbers k , and it is strongly believed that PH is infinite.

Gharibian, Santha, Sikora, Sundaram, and Yirka [11] studied two quantum generalizations of PH. They generalized the class QCMA to the quantum-classical polynomial hierarchy QCPH, the class of problems for which a quantum verifier can efficiently verify solutions given a constant number of classical proofs from competing provers. Note that this is the same as PH except the verifier can perform a polynomial-time *quantum* computation. In the same work, they generalized the class QMA(2) to the *unentangled* quantum polynomial hierarchy QPH, for which the verifier is still quantum, but the proofs are quantum mixed states and promised to be unentangled from each other. Notably, Gharibian et al. did not introduce a hierarchy in which the proofs can be *entangled*, and they did not establish a relationship between QPH and QCPH (or even QPH and PH), leaving it unclear whether or not QPH was at least as powerful as its classical counterpart.¹ More generally, if QCPH and QPH are indeed more powerful, it prompts the question of why: is it quantum verification, quantum proofs, unentanglement, or some nuanced combination?

In this work, we address all of these questions. First, we ask (and answer) what problems admit a PH-style protocol where the provers can send potentially entangled proofs. We show that this new hierarchy – the entangled quantum polynomial hierarchy (QEPH) – behaves drastically differently from what we believe about PH, QCPH, and QPH.

Second, we prove that $\text{PH} \subseteq \text{QCPH} \subseteq \text{QPH}$, confirming the intuitive relationship between these hierarchies.

Lastly, to understand the power of quantum proofs, we introduce a generalization of QCPH where the provers send probability distributions over classical proofs and denote the class by DistributionQCPH. We prove that $\text{DistributionQCPH} = \text{QCPH}$, despite the intuition from game theory that optimal strategies are usually mixed. Note that the *only* difference between DistributionQCPH and QPH is that the proofs in QPH involve quantum superposition. Hence, our result establishes that the increased computational power of QPH comes only from the quantum superposition in the proofs.

1.1 Our Results

Our first main result is a characterization of our newly defined hierarchy QEPH (Definition 19) via a collapse to its second level. This collapse is in stark contrast to our belief that PH is infinite.

► **Theorem 1** (Combination of Lemma 22 and Theorem 23). *QEPH collapses to its second level and equals QRG(1).*

This collapse is similar to others known in quantum complexity theory, such as $\text{QIP} = \text{QIP}(3) = \text{QMAM}$ [20, 25], in which the protocols rely on the prover’s ability to entangle their messages. We further compare QEPH to other complexity classes involving entangled proofs in Related and Concurrent Work.

¹ While these containments are what one might guess to be true, proving them is nontrivial.

We show that QEPH equals QRG(1), the class of problems having one-turn quantum-refereed games.² QRG(1) involves a game between two competing players that each privately sends a quantum state to a referee, who then performs a polynomial-time quantum computation to determine a winner. In 2009, Jain and Watrous [18] proved $\text{QRG}(1) \subseteq \text{PSPACE}$. However, it is conjectured that QRG(1) is strictly less powerful than $\text{QRG}(2) = \text{PSPACE}$ [15, 12]. Yet despite effort, no improved upper bounds on QRG(1) have been proven in over a decade. We suggest a new approach to improving the upper bound on QRG(1) (via the connection to QEPH) in Open Problems.

Our collapse result is stronger than stated above. It is well-known that if one extends PH to a polynomial number of rounds (rather than a constant number), then the resulting class equals PSPACE [4, Theorem 4.11]. In contrast, we show that extending QEPH to a polynomial number of rounds does not increase the power of the class.

► **Theorem 2** (Informal version of Corollary 24). *Even with a polynomial number of rounds, QEPH collapses to its second level.*

One interpretation of our collapse result is that allowing provers to entangle their proofs gives them too much opportunity to cheat. Hence, receiving a single proof from each prover is just as useful as receiving many entangled proofs.

Before this work, it was unclear how the quantum polynomial hierarchies compared to one another, and if QPH even contained PH. In our second result, we establish the following containments between the quantum and classical hierarchies, resolving an open question of Gharibian et al. [11].

► **Theorem 3** (Restatement of Theorem 26). $\text{PH} \subseteq \text{QCPH} \subseteq \text{QPH}$.

We emphasize that even $\text{PH} \subseteq \text{QPH}$ is not obvious. Placing restrictions on the provers can sometimes increase computational power, as was the case in, e.g., the recent results showing that $\text{QMA}^+ = \text{QMA}(2)^+ = \text{NEXP}$ [19, 5]. Meanwhile, the permissiveness of QEPH, where we allow the provers to entangle their proofs, seems to yield a weaker class than QPH.

In our third result, we show that the power of QCPH does not change if the provers are allowed to send probability distributions (instead of a fixed classical proof).

► **Theorem 4** (Restatement of Corollary 31). $\text{DistributionQCPH} = \text{QCPH}$.

Our motivation for studying DistributionQCPH is to better understand the power of quantum proofs. In particular, let pureQPH be the same as QPH except the quantum proofs are pure states rather than mixed states.³ Then the *only* difference between pureQPH and DistributionQCPH is that the former involves proofs that are quantum superpositions over bit strings while the latter involves proofs that are classical distributions over bit strings. Yet $\text{DistributionQCPH} = \text{QCPH}$ is in the counting hierarchy [11], and pureQPH contains QMA(2) and is contained in EXP^{PP} [2]. Conceptually, our result says that any increase in computational power only comes from the quantum superposition in the proofs.

Theorem 4 also goes through for PH.

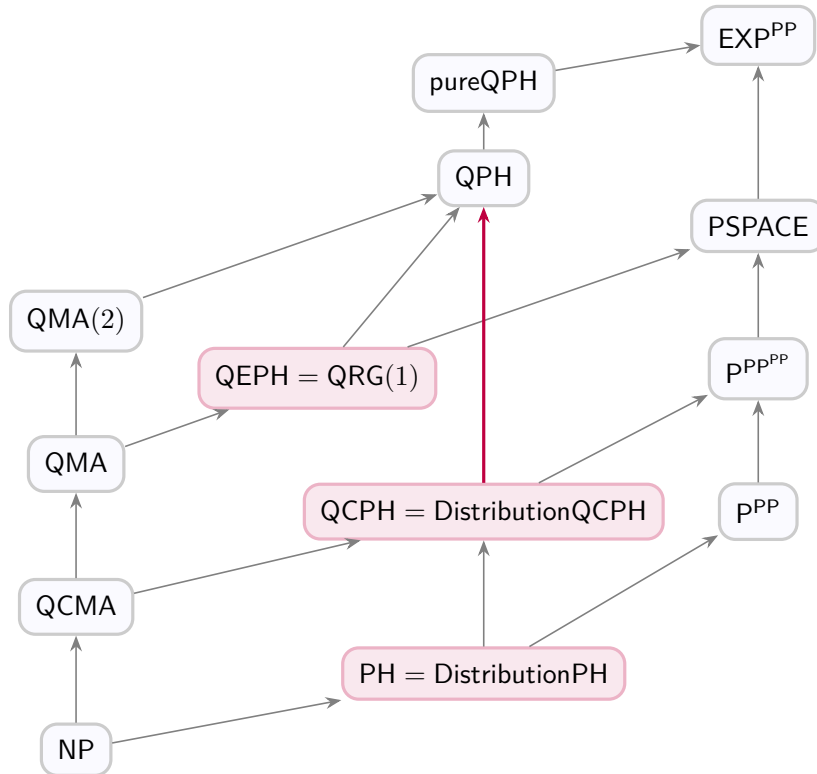
► **Theorem 5** (Restatement of Theorem 27). $\text{DistributionPH} = \text{PH}$.

² The class QRG(k) and its classical analogue RG(k) have been numbered differently by different authors. We follow recent conventions where the provers' and the referee's messages are counted separately. So, e.g., in QRG(2) the referee sends one message and then the provers each simultaneously send a message.

³ It is easy to see that $\text{QPH} \subseteq \text{pureQPH}$ since the provers can send purifications of their mixed proofs.

An easy consequence of our result is that DistributionPH collapses if and only if PH collapses.⁴ Therefore, any attempts to collapse QCPH, QPH, or pureQPH must not collapse DistributionPH, and so Theorem 5 rules out some approaches to collapsing these hierarchies. In particular, one line of attack to showing $\text{QMA}(2) = \text{NEXP}$ is to show that the \forall quantifier in $\text{Q}\Sigma_3$ does not add any computational power, because $\text{QMA}(2) \subseteq \text{Q}\Sigma_3 \subseteq \text{NEXP}$ [11]. Theorems 4 and 5 are evidence that this line of attack will not work straightforwardly, since showing the analogous result for DistributionPH would collapse the polynomial hierarchy.

We give a graphical description of our results and the quantum polynomial hierarchy landscape in Figure 1.



■ **Figure 1** (Color) The quantum polynomial hierarchy landscape in light of our work. The containments and complexity classes shown in gray were previously known, and the containments and complexity classes in red are contributions of this work.

1.2 Main Ideas

Let us consider QEPH on an intuitive level (see Definition 19 for a formal definition). QEPH can be thought of as a constant-round non-interactive game between two competing provers, Alice and Bob, who take turns sending quantum registers, i.e., collections of qubits, to a verifier. Alice and Bob are allowed to entangle their own quantum registers across turns. The verifier then performs a polynomial-time quantum computation, measures a fixed output

⁴ Distribution \mathcal{C} is not to be confused with the notation $\text{Dist}\mathcal{C}$, which has been used in average-case complexity theory, e.g. DistNP and DistPH . Also, similar names like STOCHASTIC SAT or PROBABILISTIC QBF have appeared in the study of randomized quantifiers. These are more similar to the Arthur-Merlin (AM) hierarchy.

qubit in the computational basis, and, if the verifier sees 1, they accept (Alice wins), and reject otherwise (Bob wins). QEPH contains the decision problems for which Alice always wins with high probability on yes-instances and Bob always wins with high probability on no-instances. We note that in this game the moves are *public*, which means that Alice knows the state of the quantum registers sent by Bob and vice versa. See Remark 21 for further discussion of public vs. private moves in a quantum world.

To highlight the key technique in our proof that QEPH collapses (Lemma 22), we explain how to simulate the third level of QEPH, denoted by $\text{QE}\Sigma_3$, inside of the second level $\text{QE}\Sigma_2$. The proof for higher levels proceeds by induction. As we will explain formally in Section 3, a $\text{QE}\Sigma_i$ protocol can be written as an optimization problem with a value equal to the probability the verifier accepts when both players use optimal strategies. In particular, Alice selects proofs that maximize the probability of the verifier accepting, while Bob selects proofs to minimize that probability. For $\text{QE}\Sigma_3$, given a problem instance in which the verifier's action is encoded by an observable R , the corresponding optimization problem is

$$\max_{\rho_1 \in \mathbf{D}(\mathcal{X}_1)} \min_{\sigma \in \mathbf{D}(\mathcal{Y})} \max_{\rho_2 \in \mathcal{A}} \text{tr}(R(\rho_2 \otimes \sigma)),$$

where $\mathbf{D}(\mathcal{H})$ denotes the set of density operators on the Hilbert space \mathcal{H} and $\mathcal{A} := \{\rho \in \mathbf{D}(\mathcal{X}_1 \otimes \mathcal{X}_2) \mid \text{tr}_{\mathcal{X}_2}(\rho) = \rho_1\}$. The restriction of the second maximization to the set \mathcal{A} is to enforce that Alice's second move is consistent with her first.

A straightforward analysis shows that when focusing on the inner two operators, a min-max theorem applies, allowing us to swap the ordering of the inner minimization and maximization. Then, because we allow entangled states, we can combine the two sequential maximization operators into one, leaving an optimization problem corresponding to a two-round protocol. Notably, both the three-round and two-round protocols are over the same input and verifier, so the reduction does not increase the problem size or change the error parameters.

It is natural to ask why our technique does not also collapse PH. In short, the above approach fails immediately, since, for one, our collapse theorem relies on the fact that Alice and Bob are choosing quantum proofs from compact and convex sets (see Facts 9 and 10). In contrast, the set of classical strings is not convex.

To show that $\text{QEPH} = \text{QRG}(1)$, we build on a previous characterization of Gharibian et al. [11] where they showed that the second level of the *unentangled* quantum polynomial, denoted by $\text{Q}\Sigma_2$, equals $\text{QRG}(1)$. We extend their result in Proposition 20 to show that $\text{QE}\Sigma_2 = \text{Q}\Sigma_2 = \text{QRG}(1)$, which yields our characterization that $\text{QEPH} = \text{QE}\Sigma_2 = \text{QRG}(1)$. $\text{QE}\Sigma_2 = \text{Q}\Sigma_2$ because, after two turns, each prover has only sent a single proof, so there is no distinction yet to be made between the entangled versus entangled hierarchies.

We now turn to the containment $\text{QCPH} \subseteq \text{QPH}$, which are both defined formally in Section 2.3. In QCPH, the verifier receives classical proofs, whereas the proofs in QPH are unentangled quantum mixed states. One naïve approach to simulating QCPH inside of QPH – which does not work – is for the verifier to immediately measure the quantum proofs to get classical strings and then run the QCPH verification protocol. The reason this fails is that the dishonest prover (i.e., the player without a winning strategy) can cheat by sending a quantum state, rather than a classical proof. In more detail, while the honest prover has perfect knowledge of the quantum states sent by the dishonest prover, they do not know which particular classical strings the verifier will observe upon measurement, making it unclear what their response should be. The definition of QCPH guarantees the correct player has an effective response conditioned on any particular proof sent from the other player, but

this does not guarantee the correct player can succeed against a mixture of potential moves. Unfortunately, the equilibrium point of a zero-sum game which allows for such mixed moves will generally be mixed, rather than pure.

To overcome this, we simulate the i -th level of QCPH in the $2ki$ -th level of QPH, for some constant k . We ask the provers to send k copies of each of the proofs they would send in the QCPH protocol, which increases the number of turns by a factor of $2k$. Using the groups of k proofs, we give a simple test to ensure that no player cheats, which works as follows. Measure each of the k proofs in the standard basis. If the outcomes are all equal, then the test passes, and, otherwise, the test fails. We prove that this is enough to force the provers to send computational basis states with high probability.

We remark that this bears some similarity to other protocols involving unentanglement. Harrow and Montanaro [16] used unentanglement to force Merlin to send k -partite states, and, recently, Jeronimo and Wu [19] use unentanglement to force Merlin to send many copies of (approximately) the same quantum state. Both of these results fundamentally rely on the swap test, which tests for equality between two quantum states [6]. In a similar fashion, we use unentanglement to force the provers to send standard basis states, i.e., classical strings. With that, we design a simulation of any QCPH protocol inside of QPH.

Finally, we discuss our proof that $\text{DistributionQCPH} = \text{QCPH}$ (the same techniques will also show $\text{DistributionPH} = \text{PH}$). In DistributionQCPH , the provers take turns sending probability distributions over polynomial-length classical proofs. Once all of the distributions have been sent, the verifier draws one sample from each distribution and substitutes the samples into the verification procedure. The model is somewhat subtle. The provers have perfect knowledge of the distributions sent by their opponent. However, they do not know which sample the verifier will see, because the distributions are not sampled until the end of the game. If one prefers, one can think of the distributions as quantum states that are always measured in the computational basis by the verifier.

For classical proofs of length m , the distributions sent in DistributionQCPH can have support of size exponential in m . Our key lemma says that the provers can send *much simpler* distributions without changing the acceptance probability of the verifier too much. In particular, we prove that the distributions sent by the provers can be *uniform* over $\text{poly}(m)$ many classical proofs and, even with this simplification, the acceptance probability of the verifier will change by at most a small constant. This simplification lemma (Lemma 30) generalizes a result due to Lipton and Young [24] and Althöfer [3] who showed the result in the special case of a one-turn game. Our contribution is to generalize their result to any constant number of turns.

With the simplification lemma, one can prove $\text{DistributionQCPH} \subseteq \text{QCPH}$ as follows. To send a distribution in QCPH, the provers send every classical string that is in the support of their distribution. By our simplification lemma, there are only a polynomial number of such strings, so all of them can be sent in a polynomially-sized classical proof. Then, since the simplified distributions are uniform, the verifier can randomly sample one of the strings uniformly at random. The other direction $\text{DistributionQCPH} \supseteq \text{QCPH}$ follows from the same techniques that prove $\text{QCPH} \subseteq \text{QPH}$.

1.3 Related and Concurrent Work

Early efforts to define quantum hierarchies include [34, 10].

We choose to use alternating \exists and \forall quantifiers to define QEPH (as was the case for QCPH and QPH in [11]). In addition to a quantifier definition, PH can be *equivalently* defined in the oracle model via constant-height towers of the form $\text{NP}^{\text{NP}^{\text{NP}^{\dots}}}$. The oracular

definition gives rise to natural definitions of quantum polynomial hierarchies, some of which have been studied recently. Vinkhuijzen [32] and Aaronson, Ingram, and Kretschmer [1] study the “QMA hierarchy”, QMAH, which consists of constant-depth towers of the form $\text{QMA}^{\text{QMA}^{\text{QMA}^{\dots}}}$.⁵ [32, Theorem 5] shows that QMAH is contained in the counting hierarchy CH, while the best upper bounds for the quantifier-based hierarchies, QEPH and QPH, are PSPACE and EXP^{PP} , respectively.

The method of showing equivalence between the quantifier-based and oracle-based definitions of PH does not appear to carry over to QEPH, QPH, or even QCPH. This seems related to the inability to “pull quantumness out of a quantum algorithm” as we can for randomness from randomized algorithms [1] as well as a lack of study of quantum oracle machines. We further discuss questions regarding QMAH vs. QEPH in Open Problems.

There are several quantum complexity classes that involve provers sending possibly entangled proofs to a quantum polynomial-time verifier. We do not attempt to survey them here, but, for convenience, we summarize quantum complexity classes involving entangled proofs (and their classical counterparts) in Table 1.

Our work on DistributionPH builds on previous game-theoretic characterizations in complexity theory (see e.g., [9]). PH-style classes involve a debate with public communication (perfect information), and a non-interacting, passive referee. RG-style classes involve private communication (imperfect information) with provers sending particular strings to the referee (perfect recall). A consequence of imperfect information is that the players must model their competitor’s moves as probability distributions (mixed strategies) because they are never sure which move is made. Our class DistributionPH fits into this framework in a nuanced way. Specifically, the distributions sent are public (similar to PH); they represent a mixture of pure moves (similar to RG); but, uniquely, the provers do not know which string will be sampled by the referee (reminiscent of imperfect recall). This is a novel game-theoretic model, and as we discuss further in Section 6, it is naturally motivated by a game of quantum mixed states sent to a non-interacting referee.

Finally, the independent work of Agarwal, Gharibian, Koppula, and Rudolph [2] also studies generalizations of the polynomial hierarchy. They prove $\text{QCPH} \subseteq \text{pureQPH}$, which is similar to our Theorem 26 that $\text{QCPH} \subseteq \text{QPH}$. Since $\text{QPH} \subseteq \text{pureQPH}$ is straightforward (the provers send purifications of their proofs), our Theorem 26 implies $\text{QCPH} \subseteq \text{pureQPH}$. In this sense, our containment is stronger. However, their containment has the nice (and nontrivial) feature that the k -th level of QCPH is contained in the k -th level of pureQPH, whereas our containment requires blowing up to the ck -th level of QPH for a constant integer c . Besides this, Agarwal et al. contribute several more results including a theorem that if $\text{QC}\Sigma_i = \text{QC}\Pi_i$ then QCPH collapses (see also [7]); a Karp-Lipton style result that $\text{QCMA} \subseteq \text{BQP}/\text{mpoly}$ implies QCPH collapses; a new upper bound $\text{QPH} \subseteq \text{pureQPH} \subseteq \text{EXP}^{\text{PP}}$, improving on the previous upper bound of EXPH; and a method for one-sided error-reduction of pureQPH.

1.4 Open Problems

It is well-known that PH can equivalently be defined via oracle Turing machines. This suggests oracular definitions of quantum polynomial hierarchies, such as QMAH discussed in Section 1.3. One could similarly define QCMAH as $\text{QCMA}^{\text{QCMA}^{\text{QCMA}^{\dots}}}$ and QMA(2)H as $\text{QMA}(2)^{\text{QMA}(2)^{\text{QMA}(2)^{\dots}}}$. We ask how these oracular hierarchies compare to the quantifier-based ones.

⁵ Vinkhuijzen only allows recursive queries to QMA, whereas Aaronson, Ingram, and Kretschmer allow recursive queries to PromiseQMA.

■ **Table 1** Complexity classes characterizing proof verification that are related to QEPH. “C” means classical and “Q” means quantum. For every class below, multiple provers are always competing, and, for multi-round quantum protocols, the quantum proofs can be entangled across rounds. Public means that the provers have full knowledge of their opponent’s previous turns.

Class	# of Rounds	# of Provers	Proofs	Verifier	Interaction from referee?	Public or Private	Equals
NP	1	1	C	C	no	N/A	
QMA	1	1	Q	Q	no	N/A	
IP	poly	1	C	C	yes	N/A	PSPACE [28]
QIP(3)	3	1	Q	Q	yes	N/A	PSPACE [17]
PH	const	2	C	C	no	pub.	
QEPH	const	2	Q	Q	no	pub.	QRG(1)
RG(1)	1	2	C	C	no	priv.	S ₂ P [3, 24]
RG(2)	2	2	C	C	yes	priv.	PSPACE [8]
RG	poly	2	C	C	yes	priv.	EXP [8]
RG(pub)	poly	2	C	C	yes	pub.	PSPACE [8]
QRG(1)	1	2	Q	Q	no	priv.	
QRG(2)	2	2	Q	Q	yes	priv.	PSPACE [15]
QRG	poly	2	Q	Q	yes	priv.	EXP [14]

► **Question 6.** Does $\text{QEPH} = \text{QMAH}$? $\text{QPH} = \text{QMA(2)H}$? $\text{QCPH} = \text{QCMAH}$?

It is unclear if these hierarchies are equal, as in the classical world, or if one version would be stronger than the other. One immediate obstacle is the fact that QEPH and QPH are quantifying over quantum states, so perhaps it is easier to begin with QCPH, which still quantifies over classical bits. Alas, it is still unclear if an oracle machine definition of QCPH would be equal to a quantifier definition, since, in the oracular case, queries can be made in superposition.

Answering Question 6 could yield progress towards characterizing QRG(1). Jain and Watrous showed that $\text{QRG(1)} \subseteq \text{PSPACE}$ in 2009 [18], and, since then, no improved upper bounds have been proven despite effort [12]. Our work shows that $\text{QRG(1)} = \text{QEPH}$. If one can show $\text{QEPH} \subseteq \text{QMAH}$, then that would imply $\text{QRG(1)} \subseteq \text{CH}$, because $\text{QMAH} \subseteq \text{CH}$ [32].

More broadly, proving better upper or lower bounds on the quantum polynomial hierarchies and finding more connections to other parts of complexity theory are important directions for future work. For example, does any level of QPH contain PSPACE? Can one improve the containment $\text{QPH} \subseteq \text{EXP}^{\text{PP}}$? Or, how can these hierarchies be used to better understand the relationships between QCMA, QMA, and QMA(2)?

2 Preliminaries

We introduce notation, definitions, and background that are central to our results. For the most part, we assume familiarity with common concepts and classes in quantum and classical complexity theory as well as quantum computing and quantum information. For a thorough discussion of these topics, see [4, 33, 21, 27].

We will need the following version of Hoeffding’s inequality.

► **Fact 7** (Hoeffding's inequality). Let X_1, \dots, X_n be independent random variables subject to $a_i \leq X_i \leq b_i$ for all i . Let $X = \sum_{i=1}^n X_i$ and let $\mu = \mathbf{E}[X]$. Then it holds that

$$\Pr[X - \mu \geq t] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right)$$

and

$$\Pr[X - \mu \leq -t] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right). \quad \lrcorner$$

2.1 Quantum Information

A *quantum register* refers to a collection of qubits. Associated with each register is a complex Hilbert space, and the state of a quantum register is described by a Hermitian, positive semi-definite matrix with trace one called a *density matrix*. We denote the set of n -qubit density matrices by $\mathbf{D}(n)$, and the sets of linear operators and density matrices on a complex Hilbert space \mathcal{H} by $\mathbf{L}(\mathcal{H})$ and $\mathbf{D}(\mathcal{H})$, respectively.

For two quantum registers (X, Y) with Hilbert spaces \mathcal{X} and \mathcal{Y} , the combined space is the tensor product space $\mathcal{X} \otimes \mathcal{Y}$. The partial trace $\text{tr}_{\mathcal{Y}} : \mathbf{L}(\mathcal{X} \otimes \mathcal{Y}) \rightarrow \mathbf{L}(\mathcal{X})$ is the unique linear map that satisfies $\text{tr}_{\mathcal{Y}}(A \otimes B) = \text{tr}(A)B$ for all $A \in \mathbf{L}(\mathcal{X})$ and $B \in \mathbf{L}(\mathcal{Y})$. If the compound register (X, Y) is in the state $\rho \in \mathbf{D}(\mathcal{X} \otimes \mathcal{Y})$, then the state of register X is $\text{tr}_{\mathcal{Y}}(\rho) \in \mathbf{D}(\mathcal{X})$. That is, operationally speaking, the partial trace is the act of ignoring (or discarding) a quantum register. We note that the partial trace $\text{tr}_{\mathcal{X}}$ can be defined similarly, and, in general, the context in which the partial trace is used should clarify which spaces are being “traced out”.

A *quantum measurement* of a quantum register is described by a finite collection of Hermitian, positive semi-definite matrices that sum to identity. Let X be a quantum register with Hilbert space \mathcal{X} whose state is described by ρ . Let $\mathcal{M} = \{E_i \mid i \in \Sigma\}$ be a quantum measurement, where Σ is a finite alphabet. Upon measuring X with \mathcal{M} , we observe $i \in \Sigma$ with probability $\text{tr}(E_i \rho)$.

2.2 A Min-Max Theorem

To prove our collapse theorem, we use a weaker version of Sion's min-max theorem.

► **Theorem 8** (A weaker version of Sion's min-max theorem [29]). Let \mathcal{X} and \mathcal{Y} be complex Euclidean spaces, let $\mathcal{A} \subseteq \mathcal{X}$ and $\mathcal{B} \subseteq \mathcal{Y}$ be convex and compact subsets, and let $f : \mathcal{A} \times \mathcal{B} \rightarrow \mathbb{R}$ be a bilinear function. Then

$$\max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} f(a, b) = \min_{b \in \mathcal{B}} \max_{a \in \mathcal{A}} f(a, b). \quad \lrcorner$$

It is a well-known fact that the space of density matrices is compact and convex.

► **Fact 9** ([33, Chapter 1]). Let $\mathbf{D}(\mathcal{H})$ be the set of density matrices on a complex Hilbert space \mathcal{H} . $\mathbf{D}(\mathcal{H})$ is compact and convex.

It is critical for us that, even if we impose partial trace constraints on the set of density matrices, the set remains compact and convex. We include a proof for completeness.

► **Fact 10.** Let X, Y be two quantum registers with Hilbert spaces \mathcal{X} and \mathcal{Y} , respectively, and let $\mathbf{D}(\mathcal{X} \otimes \mathcal{Y})$ be the corresponding set of density operators. Let $\rho' \in \mathbf{D}(\mathcal{X})$ be some fixed density matrix. Then the set

$$\mathbf{S} = \{\rho \in \mathbf{D}(\mathcal{X} \otimes \mathcal{Y}) \mid \text{tr}_{\mathcal{Y}}(\rho) = \rho'\}$$

is compact and convex.

6:10 The Entangled Quantum Polynomial Hierarchy Collapses

Proof. Let $\rho_1, \rho_2 \in \mathbf{S}$, and define $\sigma := \theta\rho_1 + (1 - \theta)\rho_2$ for $\theta \in [0, 1]$. Then

$$\begin{aligned} \text{tr}_{\mathcal{Y}}(\sigma) &= \text{tr}_{\mathcal{Y}}(\theta\rho_1 + (1 - \theta)\rho_2) \\ &= \theta \text{tr}_{\mathcal{Y}}(\rho_1) + (1 - \theta) \text{tr}_{\mathcal{Y}}(\rho_2) && \text{(By the linearity of the partial trace.)} \\ &= \theta\rho' + (1 - \theta)\rho' && \text{(Because } \rho_1, \rho_2 \in \mathbf{S}\text{.)} \\ &= \rho', \end{aligned}$$

so \mathbf{S} is convex.

To show that \mathbf{S} is compact, we must show that it is closed and bounded. Without loss of generality, let \mathbf{X} be an n -qubit register and \mathbf{Y} be an m -qubit register. Then we can identify \mathbf{S} with the vector space $\mathbb{C}^{4^{n+m}}$ and observe that all entries are bounded in magnitude by 1. Therefore, \mathbf{S} is bounded. To see that \mathbf{S} is closed, we need the following definitions. For $x \in \mathbb{C}$, define $f_x : \mathbb{C}^{4^{n+m}} \rightarrow \mathbb{C}$ as $f_x(A) = \langle x, Ax \rangle$, which is continuous because the inner product is continuous; define $g : \mathbb{C}^{4^{n+m}} \rightarrow \mathbb{C}^{4^{n+m}}$ as $g(A) = A - A^\dagger$, which is a polynomial and therefore continuous; and, finally, define $h : \mathbb{C}^{4^{n+m}} \rightarrow \mathbb{C}^{4^n}$ as $h(A) = \text{tr}_{\mathcal{Y}}(A)$, which is a linear map on a finite-dimensional vector space and therefore continuous. Then

$$\mathbf{S} = \bigcap_{x \in \mathbb{C}} f_x^{-1}([0, \infty)) \cap g^{-1}(\{0\}) \cap h^{-1}(\{\rho'\}) \cap \text{tr}^{-1}(\{1\}).$$

The preimage of a continuous function on a closed set is closed, and the intersection of closed sets is closed. Therefore, \mathbf{S} is closed. \blacktriangleleft

2.3 Previously Studied Hierarchies

Here, we give formal definitions of the polynomial hierarchy PH, the quantum-classical polynomial hierarchy QCPH, and the unentangled quantum polynomial hierarchy QPH, the latter two of which were both introduced by Gharibian et al. [11]. These classes will appear again in Section 5 when we prove $\text{QCPH} \subseteq \text{QPH}$ and in Section 6 when we prove $\text{DistributionQCPH} = \text{QCPH}$. We defer definitions of our new classes until later, with QEPH studied in Section 4 and DistributionQCPH in Section 6.

► **Definition 11** (Σ_i^p). *A language L is in the i -th level of the polynomial hierarchy Σ_i^p if there exists a polynomial-time deterministic Turing Machine M such that for any n -bit input x ,*

$$\begin{aligned} x \in L &\iff \exists y_1 \forall y_2 \exists y_3 \dots Q_i y_i \text{ such that } M(x, y_1, \dots, y_i) = 1, \\ x \notin L &\iff \forall y_1 \exists y_2 \forall y_3 \dots \overline{Q}_i y_i \text{ such that } M(x, y_1, \dots, y_i) = 0, \end{aligned}$$

where Q_i denotes \exists if i is odd and \forall otherwise, \overline{Q}_i denotes the complement of Q_i , and $|y_i| \leq p(n)$ for some fixed polynomial p for all i .

► **Definition 12** (The polynomial hierarchy (PH) [31]). *The Polynomial-time Hierarchy is defined as*

$$\text{PH} := \bigcup_{i=0}^{\infty} \Sigma_i^p. \quad \lrcorner$$

Note the union which defines PH is over values of i which are constant, independent of a problem's input size. Observe also that for all i , $\Sigma_i^p \subseteq \Sigma_{i+1}^p$. Additionally, PH is closed under complement, in particular because $\overline{\Sigma_i^p} \subseteq \Sigma_{i+1}^p \subseteq \text{PH}$. The complement of Σ_i^p is defined to be Π_i^p , and for all i we have $\Sigma_i^p \subseteq \Pi_{i+1}^p \subseteq \Pi_{i+2}^p$.

The definition of PH is particularly robust. The class can be defined equivalently by $\Sigma_{i+1}^P = \text{NP}^{\Sigma_i^P}$, giving a constant-height tower of NP oracles. The model of alternating nondeterministic Turing Machines also can be used to define each level of the hierarchy. In another direction, the Sipser–Lautemann theorem shows $\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P \subseteq \text{PH}$ [30, 23]. So, natural bounded-error or probabilistic definitions of PH collapse to the standard, deterministic definition given above. This is also true for oracle definitions, where we know $\text{MA}^{\text{MA}^{\dots}} = \text{PH}$.

Even a partial survey of results regarding PH would be impossible to fit here. We finally note that $\Sigma_i^P = \Sigma_{i+1}^P$ or $\Sigma_i^P = \Pi_i^P$ would both “collapse” the hierarchy so that $\text{PH} = \Sigma_i^P$. These two events are analogous to $\text{P} = \text{NP}$ or $\text{NP} = \text{coNP}$. Conversely, if PH collapses to any finite level, it implies analogs of $\text{P} = \text{NP}$ and $\text{NP} = \text{coNP}$ must be true for some degree of nondeterminism, at some level of the hierarchy. So, the strongly-believed conjecture that PH is not equal to any Σ_i^P for fixed i is a generalization of those other strongly-believed conjectures.

The uniform circuit model is standard for quantum complexity classes, so we give the definition below.

► **Definition 13** (Polynomial-time uniform family of quantum circuits). *A polynomial-time uniform family of quantum circuits is a family $\{V_n\}_{n \in \mathbb{N}}$ such that there exists a polynomial bounded function $t : \mathbb{N} \rightarrow \mathbb{N}$ and a deterministic Turing machine M acting as follows. For every n -bit input x , M outputs in time $t(n)$ a description of a quantum circuit V_n , which has a designated output qubit. We say V_n accepts when we observe a 1 upon measuring the designated output qubit in the standard basis.*

We generally leave the subscript implicit and just write V . Additionally, we often consider a single problem instance defined by an input x for the full length of an analysis. So instead of writing $V(x, y)$ for input x and proof y , we simply refer to $V(y)$.

As with most quantum complexity classes, we will be working with promise problems. Briefly, a promise problem A is a pair of non-intersecting subsets $(A_{\text{yes}}, A_{\text{no}})$ of $\{0, 1\}^*$. A decision problem, or language, is a promise problem where $A_{\text{yes}} \cup A_{\text{no}} = \{0, 1\}^*$.

We are now ready to define QCPH.

► **Definition 14** ($\text{QC}\Sigma_i$ [11]). *A promise problem $L = (L_{\text{yes}}, L_{\text{no}})$ is in i -th level of the quantum-classical polynomial hierarchy $\text{QC}\Sigma_i(c, s)$ for polynomial-time computable functions $c, s : \mathbb{N} \rightarrow [0, 1]$ if there exists a polynomial-time uniform family of quantum circuits $\{V_n\}_{n \in \mathbb{N}}$ such that for every n -bit input x , V_n takes in proofs $y_1, \dots, y_i \subseteq \{0, 1\}^{m(n)}$ for fixed polynomial m and measures a fixed output qubit to decide to accept or reject, such that*

- *Completeness: $x \in L_{\text{yes}} \Rightarrow \exists y_1 \forall y_2 \exists y_3 \dots Q_i y_i$ such that $\Pr[V(y_1, \dots, y_i) \text{ accepts}] \geq c$,*
 - *Soundness: $x \in L_{\text{no}} \Rightarrow \forall y_1 \exists y_2 \forall y_3 \dots \overline{Q_i} y_i$ such that $\Pr[V(y_1, \dots, y_i) \text{ accepts}] \leq s$,*
- where Q_i denotes \exists if i is odd and \forall otherwise, $\overline{Q_i}$ denotes the complement of Q_i , and, for all i , $|y_i| \leq p(n)$ for a fixed polynomially bounded function p . When the completeness and soundness parameters c, s are not specified, define

$$\text{QC}\Sigma_i := \bigcup_{c-s \in \Omega(1/\text{poly}(n))} \text{QC}\Sigma_i(c, s). \quad \lrcorner$$

► **Definition 15** (The quantum-classical polynomial hierarchy (QCPH) [11]). *The quantum-classical polynomial hierarchy is defined as*

$$\text{QCPH} := \bigcup_{i=0}^{\infty} \text{QC}\Sigma_i. \quad \lrcorner$$

Observe that $\text{QC}\Sigma_0 = \text{BQP}$ and $\text{QC}\Sigma_1 = \text{QCMA}$. Gharibian et al. [11] proved that QCPH is contained in P^{PPP} , the second level of the counting hierarchy CH.

6:12 The Entangled Quantum Polynomial Hierarchy Collapses

The definition of $\text{QC}\Sigma_i$ to generically include $\text{QC}\Sigma_i(c, s)$ for all $c - s \geq 1/\text{poly}(n)$ is justified in part by the result of [11] that for any such c and s , we may reduce the error such that for any polynomially bounded function r , we have $\text{QC}\Sigma_i(c, s) = \text{QC}\Sigma_i(1 - 2^{-r}, 2^{-r})$.

The unentangled quantum polynomial hierarchy QPH is defined similarly. The only difference is that the classical proofs are replaced by unentangled quantum proofs.

► **Definition 16** ($\text{Q}\Sigma_i$ [11]). *A promise problem $L = (L_{\text{yes}}, L_{\text{no}})$ is in the i -th level of the unentangled quantum polynomial hierarchy $\text{Q}\Sigma_i(c, s)$ for polynomial-time computable functions $c, s : \mathbb{N} \rightarrow [0, 1]$ if there exists a polynomial-time uniform family of quantum circuits $\{V_n\}_{n \in \mathbb{N}}$ such that for every n -bit input x , V_n takes in quantum proofs ρ_1, \dots, ρ_i and measures a fixed output qubit to decide to accept or reject, such that*

- *Completeness: $x \in L_{\text{yes}} \Rightarrow \exists \rho_1 \forall \rho_2 \exists \rho_3 \dots Q_i \rho_i$ such that $\Pr[V(\rho_1, \dots, \rho_i) \text{ accepts}] \geq c$,*
 - *Soundness: $x \in L_{\text{no}} \Rightarrow \forall \rho_1 \exists \rho_2 \forall \rho_3 \dots \overline{Q_i} \rho_i$ such that $\Pr[V(\rho_1, \dots, \rho_i) \text{ accepts}] \leq s$,*
- where Q_i denotes \exists if i is odd and \forall otherwise, $\overline{Q_i}$ denotes the complement of Q_i , and, for all i , ρ_i is a $p(n)$ -qubit state for a fixed polynomially bounded function p . When the completeness and soundness parameters c, s are not specified, define

$$\text{Q}\Sigma_i := \bigcup_{c-s \in \Omega(1)} \text{Q}\Sigma_i(c, s). \quad \lrcorner$$

► **Definition 17** (QPH [11]). *The unentangled quantum polynomial hierarchy is defined as*

$$\text{QPH} := \bigcup_{i=0}^{\infty} \text{Q}\Sigma_i. \quad \lrcorner$$

Interestingly, $\text{QMA}(2) \subseteq \text{Q}\Sigma_3$, since the verifier can simply ignore the second proof.

Here, we let $\text{Q}\Sigma_i = \text{Q}\Sigma_i(c, s)$ for $c - s \geq \Omega(1)$, rather than $1/\text{poly}(n)$, because we do not currently have an error reduction result for QPH similar to the one known for QCPH (although, [2] recently made progress in this direction).

3 The Entangled Quantum Polynomial Hierarchy

We formally define the entangled quantum polynomial hierarchy. The definition appears more technical than for QCPH and QPH, but this is mostly just an issue of notation.

► **Definition 18** (i -th level of the entangled quantum polynomial hierarchy ($\text{QE}\Sigma_i$)). *A promise problem $L = (L_{\text{yes}}, L_{\text{no}})$ is in $\text{QE}\Sigma_i(c, s)$ for polynomial-time computable functions $c, s : \mathbb{N} \rightarrow [0, 1]$ if there exists a polynomial-time uniform family of quantum circuits $\{V_n\}_{n \in \mathbb{N}}$ such that for every n -bit input x , V_n takes quantum proofs, measures a fixed output qubit to decide to accept or reject, and satisfies*

- *Completeness: $x \in L_{\text{yes}} \Rightarrow \exists \rho_1 \forall \rho_2 \exists \rho_3 \dots Q_i \rho_i$ such that $\Pr[V(\rho_{i-1}, \rho_i) \text{ accepts}] \geq c$,*
 - *Soundness: $x \in L_{\text{no}} \Rightarrow \forall \rho_1 \exists \rho_2 \forall \rho_3 \dots \overline{Q_i} \rho_i$ such that $\Pr[V(\rho_{i-1}, \rho_i) \text{ accepts}] \leq s$,*
- where each ρ_j is chosen from the set

$$\mathcal{A}_j := \begin{cases} \{\rho \in \mathbf{D}(\mathcal{X}_1 \otimes \mathcal{X}_3 \otimes \dots \otimes \mathcal{X}_j) \mid \text{if } j > 1, \text{tr}_{\mathcal{X}_j}(\rho) = \rho_{j-2}\} & \text{if } j \text{ is odd} \\ \{\rho \in \mathbf{D}(\mathcal{X}_2 \otimes \mathcal{X}_4 \otimes \dots \otimes \mathcal{X}_j) \mid \text{if } j > 2, \text{tr}_{\mathcal{X}_i}(\rho) = \rho_{j-2}\} & \text{if } j \text{ is even} \end{cases}.$$

Here, Q_i denotes \exists if i is odd and \forall otherwise, and $\overline{Q_i}$ denotes the complement of Q_i . For all i , the corresponding Hilbert space \mathcal{X}_i is a space of at most $p(n)$ qubits for a fixed polynomial p . When the completeness/soundness parameters are not specified, define

$$\text{QE}\Sigma_i := \bigcup_{c-s \in \Omega(1/\text{poly}(n))} \text{QE}\Sigma_i(c, s). \quad \lrcorner$$

► **Definition 19** (The entangled quantum polynomial hierarchy (QEPH)). *The entangled quantum polynomial hierarchy is defined as*

$$\text{QEPH} = \bigcup_{i=0}^{\infty} \text{QE}\Sigma_i. \quad \lrcorner$$

When introducing a complexity class, perhaps the first question one should ask is whether or not the choice of completeness and soundness parameters actually matter. In [11, Theorem 2.6], it was shown that QCPH is robust to the choice of error parameters, but no such result is known for QPH. In Section 4, we show that the choice of parameters does not matter for any level of QEPH, i.e., for c, s such that $c - s \geq 1/\text{poly}(n)$, $\text{QE}\Sigma_i(c, s) = \text{QE}\Sigma_i(\frac{2}{3}, \frac{1}{3})$ for all $i \in \mathbb{N}$ (see Theorem 25).

Let us also make several remarks on our definition. As for PH, the indices i in the definition of QEPH are constants, independent of a problem's input size, and, as one should expect, $\text{BQP} = \text{QE}\Sigma_0$ and $\text{QMA} = \text{QE}\Sigma_1$. One can also define $\text{QE}\Pi_i := \overline{\text{QE}\Sigma_i}$ and $\text{QE}\Delta_i := \text{QE}\Sigma_i \cap \text{QE}\Pi_i$. The players also have no incentive to entangle their moves with their opponent because $\text{QE}\Sigma_i$ can be modeled as a zero-sum game. Therefore, we may assume the even and odd indexed states are unentangled.

Informally, $\text{QE}\Sigma_i$ can be thought of as the following game, where we assume i is even to simplify the exposition. Alice has (possibly entangled) quantum registers $(A_1, \dots, A_{i/2})$, and Bob has (possibly entangled) quantum registers $(B_1, \dots, B_{i/2})$, where each register is a number of qubits that is polynomial in the input size. The game commences as follows. In the first round, Alice reveals the state ρ_1 of A_1 , and then Bob reveals the state σ_1 of B_1 . In the second round, Alice reveals the state ρ_2 of (A_1, A_2) , and Bob reveals the state σ_2 of (B_1, B_2) . To ensure Alice and Bob do not change their “moves” from previous rounds, we demand that $\text{tr}_{A_2}(\rho_2) = \rho_1$ and $\text{tr}_{B_2}(\sigma_2) = \sigma_1$. That is, Alice and Bob cannot modify the state of subsystems that have been revealed in previous rounds. In general, for the i -th round, it must be that $\text{tr}_{A_i}(\rho_i) = \rho_{i-1}$ and $\text{tr}_{A_i}(\sigma_i) = \sigma_{i-1}$. The game continues like this until the global states of $(A_1, \dots, A_{i/2})$ and $(B_1, \dots, B_{i/2})$ are known to both players and the referee.

At this point, the referee must accept or reject. The referee's action is determined by a polynomial-time quantum circuit and a single-qubit measurement. This action can be equivalently expressed as a two-outcome quantum measurement $\{R, I - R\}$, where the first observable corresponds to accepting. Then, the probability the referee accepts is equal to $\text{tr}(R(\rho_{i/2} \otimes \sigma_{i/2}))$. We emphasize that we do not intend to actually write the observable R corresponding to some verification circuit V . Rather, the observable R is a convenient way to express the action of the referee.

Alice's goal is to maximize the acceptance probability, and Bob's goal is to minimize the acceptance probability. Therefore, given an instance of a $\text{QE}\Sigma_i$ problem with corresponding observable R , we can express the acceptance probability achieved by both players playing optimal strategies as

$$v = \max_{\rho_1 \in \mathcal{A}_1} \min_{\sigma_1 \in \mathcal{B}_1} \dots \max_{\rho_{i/2} \in \mathcal{A}_{i/2}} \min_{\sigma_{i/2} \in \mathcal{B}_{i/2}} \text{tr}(R(\rho_{i/2} \otimes \sigma_{i/2})), \quad (1)$$

where \mathcal{A}_i and \mathcal{B}_i are defined as in Definition 18, and each alternating max/min operator corresponds to an alternation of quantifiers in Definition 18. In this work, we intend to use Equation (1) as a tool for proving the equality of one game/problem instance to another.

Finally, as an application of Equation (1), we observe that the second levels of both QEPH and QPH are equal to QRG(1), which is known to be contained in PSPACE.

6:14 The Entangled Quantum Polynomial Hierarchy Collapses

► **Proposition 20** (Extension of [11, Corollary 1.9]).

$$\text{QE}\Sigma_2 = \text{QE}\Pi_2 = \text{Q}\Sigma_2 = \text{Q}\Pi_2 = \text{QRG}(1) \subseteq \text{PSPACE}. \quad \lrcorner$$

Proof. In [11], it was observed that $\text{Q}\Sigma_2 = \text{QRG}(1)$. Here, we use the same reasoning to conclude $\text{QE}\Sigma_2 = \text{QE}\Pi_2 = \text{Q}\Sigma_2 = \text{Q}\Pi_2 = \text{QRG}(1)$. The equivalence is clear given that the value of a $\text{QRG}(1)$ protocol is described by an expression identical to Equation (1) when $i = 2$, which corresponds to $\text{QE}\Sigma_2$ (see [18] for a formal definition of $\text{QRG}(1)$). Then, note that $\text{QRG}(1)$ is closed under complement, by a min-max theorem, implying $\text{QE}\Sigma_2 = \text{QE}\Pi_2$. Second, because entanglement is not a concern until one of the players makes multiple moves, the second levels of the entangled and unentangled hierarchies are equal (similarly, the first levels are equal to each other, as are the zeroth levels). Finally, the containment of $\text{QRG}(1)$ in PSPACE is due to [18, Proposition 4]. ◀

The fact that $\text{QE}\Sigma_2 = \text{QE}\Pi_2 = \text{Q}\Sigma_2 = \text{Q}\Pi_2$ is somewhat striking, since such an equality in the classical setting would imply a collapse of PH [4, Theorem 5.6].⁶

► **Remark 21** (Public vs. private quantum proofs). While the quantum polynomial hierarchies are well-defined, some may object that the classes are unphysical because the provers have full knowledge of each other’s density matrices, even though the verifier only receives a single copy of each proof. The quantum no-cloning theorem also begs the question of how exactly the information is communicated between the provers. This is not an issue for PH because it is trivial to learn classical proofs given a single copy, and, for QRG , this is not an issue because communication is private. Even in quantum complexity theory, provers which are considered “all-powerful” are still usually considered to be bound by the laws of quantum mechanics.

Despite being unphysical, we are content with the definition for two reasons. First, it is a well-defined, useful theoretical tool for studying quantum information. Second, in the case of QEPH , we show that it collapses to $\text{QE}\Sigma_2$, where it is known, by a min-max theorem, that public vs. private communication is irrelevant. So, despite starting with an unphysical definition, we show equivalence with a class that adheres entirely to the laws of quantum mechanics. ◻

4 The Entangled Quantum Polynomial Hierarchy Collapses

We prove several results about the entangled quantum polynomial hierarchy. Specifically, we prove that QEPH collapses to its second level, is equal to $\text{QRG}(1)$, and that every level of QEPH is robust to the choice of completeness and soundness parameters (i.e., for c, s such that $c - s \geq 1/\text{poly}(n)$, $\text{QE}\Sigma_i(c, s) = \text{QE}\Sigma_i(\frac{2}{3}, \frac{1}{3})$ for all $i \in \mathbb{N}$). We begin by proving that the hierarchy collapses.

► **Lemma 22.** For all constants $i \geq 2$, $\text{QE}\Sigma_2 = \text{QE}\Sigma_i$.

Proof. Note that for all i , $\text{QE}\Sigma_{i-1}$ is trivially contained in $\text{QE}\Sigma_i$. We will show that for all $i > 2$, $\text{QE}\Sigma_i \subseteq \text{QE}\Sigma_{i-1}$ by an induction argument, beginning with $\text{QE}\Sigma_3 \subseteq \text{QE}\Sigma_2$.

Recall from Equation (1) in Section 3 that the value of a $\text{QE}\Sigma_3$ protocol is equal to

$$\hat{v} = \max_{\rho_1 \in \mathcal{D}(\mathcal{X}_1)} \min_{\sigma_1 \in \mathcal{D}(\mathcal{Y}_1)} \max_{\rho_2 \in \mathcal{A}} \text{tr}(R(\rho_2 \otimes \sigma_1)),$$

⁶ This phenomenon of the second levels being equal is also true for TFPH , the hierarchy generalizing the class TFNP [22].

where R is the observable corresponding to the verifier accepting, \mathcal{X}_1 , \mathcal{Y}_1 , and \mathcal{X}_2 are the Hilbert spaces containing the three proofs, and $\mathcal{A} = \{\rho \in \mathbf{D}(\mathcal{X}_1 \otimes \mathcal{X}_2) \mid \text{tr}_{\mathcal{A}_2}(\rho) = \rho_1\}$, which enforces that Alice's second proof is consistent with her first.

For any choice of $\rho_1 \in \mathbf{D}(\mathcal{X}_1)$, define

$$v(\rho_1) = \min_{\sigma_1 \in \mathbf{D}(\mathcal{Y}_1)} \max_{\rho_2 \in \mathcal{A}} \text{tr}(R(\rho_2 \otimes \sigma_1)),$$

so that $\hat{v} = \max_{\rho_1 \in \mathbf{D}(\mathcal{X}_1)} v(\rho_1)$. Consider that $\mathbf{D}(\mathcal{Y}_1)$ and \mathcal{A} are compact and convex by Facts 9 and 10. Additionally, the function $\text{tr}(R(\rho_2 \otimes \sigma_1))$ is a composition of bilinear functions and so itself is bilinear in σ_1 and ρ_2 . Therefore, by Theorem 8, a min-max theorem applies and

$$v(\rho_1) = \max_{\rho_2 \in \mathcal{A}} \min_{\sigma_1 \in \mathbf{D}(\mathcal{Y}_1)} \text{tr}(R(\rho_2 \otimes \sigma_1)) = \min_{\sigma_1 \in \mathbf{D}(\mathcal{Y}_1)} \max_{\rho_2 \in \mathcal{A}} \text{tr}(R(\rho_2 \otimes \sigma_1)),$$

changing the optimization problem without changing the value.

Substituting this back into \hat{v} , we find

$$\begin{aligned} \hat{v} &= \max_{\rho_1 \in \mathbf{D}(\mathcal{X}_1)} v(\rho_1) \\ &= \max_{\rho_1 \in \mathbf{D}(\mathcal{X}_1)} \max_{\rho_2 \in \mathcal{A}} \min_{\sigma_1 \in \mathbf{D}(\mathcal{Y}_1)} \text{tr}(R(\rho_2 \otimes \sigma_1)) \\ &= \max_{\rho_2 \in \mathbf{D}(\mathcal{X}_1 \otimes \mathcal{X}_2)} \min_{\sigma_1 \in \mathbf{D}(\mathcal{Y}_1)} \text{tr}(R(\rho_2 \otimes \sigma_1)), \end{aligned} \tag{2}$$

where the final equality is clear given the definition of \mathcal{A} .

We observe that Equation (2) matches the characterization of a $\text{QE}\Sigma_2$ protocol given in Equation (1). Therefore, we have shown the value \hat{v} of an arbitrary $\text{QE}\Sigma_3$ protocol is equivalent to the value of a $\text{QE}\Sigma_2$ protocol. Given an instance of a $\text{QE}\Sigma_3$ problem verified by some polynomial-time uniform circuit V – corresponding to the observable R above – whether V is satisfiable by a $\text{QE}\Sigma_3$ protocol is equivalent to whether V is satisfiable by a $\text{QE}\Sigma_2$ protocol, i.e. $\text{QE}\Sigma_3 \subseteq \text{QE}\Sigma_2$ and indeed they are equal.

By way of induction, assume $\text{QE}\Sigma_2 = \text{QE}\Sigma_i$ for some constant $i > 2$. By the same min-max argument as just before, we may show the equivalence of the value of any $\text{QE}\Sigma_{i+1}$ protocol to the value of a $\text{QE}\Sigma_i$ protocol, thus showing the equivalence of the classes. Therefore, the hierarchy QEPH collapses to $\text{QE}\Sigma_2$. ◀

The equality between QEPH and $\text{QRG}(1)$ is a straightforward consequence of the collapse lemma.

► **Theorem 23.** $\text{QRG}(1) = \text{QEPH} = \text{QE}\Sigma_2$.

Proof. Combining the results $\text{QRG}(1) = \text{QE}\Sigma_2$ from Proposition 20 and $\text{QE}\Sigma_2 = \text{QEPH}$ from Lemma 22 proves the equality. ◀

Next, we note that our collapse theorem can be strengthened to $\text{QE}\Sigma_i = \text{QE}\Sigma_2$ for any polynomially bounded i , rather than just constant. Like classical PH, we define QEPH as the union of $\text{QE}\Sigma_i$ for any constant i . This is a natural way of defining PH as it is key to proving that if $\text{P} = \text{NP}$, then PH collapses. However, in contrast to collapse techniques for classical PH, our reduction of $\text{QE}\Sigma_i$ to $\text{QE}\Sigma_2$ does not increase the problem size. In our proof of Lemma 22, the $\text{QE}\Sigma_2$ problem in Equation (2) optimizes over the same quantity as the original $\text{QE}\Sigma_i$ problem. Therefore, our proof applies even to a super-constant number of rounds. The reduction is valid up to a polynomial number of rounds, after which the concatenation of the proof registers would lead to a proof too large for the polynomial-time verifier to accept.

6:16 The Entangled Quantum Polynomial Hierarchy Collapses

► **Corollary 24.** $\text{QE}\Sigma_i = \text{QE}\Sigma_2$ for any polynomially-bounded i .

Finally, our results also prove that $\text{QE}\Sigma_i$ is robust to the choice of error parameters.

► **Theorem 25.** For any choice of c, s such that $c - s \geq 1/\text{poly}(n)$, it holds that $\text{QE}\Sigma_i(c, s) = \text{QE}\Sigma_i(\frac{2}{3}, \frac{1}{3})$.

Proof. The reverse containment is trivial, so we focus on proving the forward direction, reducing $\text{QE}\Sigma_i(c, s)$ to $\text{QE}\Sigma_i(\frac{2}{3}, \frac{1}{3})$. Again appealing to the fact that our proof of Lemma 22 shows that a $\text{QE}\Sigma_3$ problem is equivalent to a $\text{QE}\Sigma_2$ problem with the same game value, we observe that our proof implies $\text{QE}\Sigma_2(c, s) = \text{QE}\Sigma_i(c, s)$. Then, because the equality of $\text{QRG}(1)$ and $\text{QE}\Sigma_2$ (Theorem 23) is also based on the optimization definition from Equation (1), the acceptance probability remains preserved and $\text{QE}\Sigma_2(c, s) = \text{QRG}(1)(c, s)$. We may then appeal to the result of [13] that a parallel repetition theorem holds for $\text{QRG}(1)$, so that $\text{QRG}(1)(c, s) = \text{QRG}(1)(\frac{2}{3}, \frac{1}{3})$. By the same reasoning as a moment ago, this last class equals $\text{QE}\Sigma_2(\frac{2}{3}, \frac{1}{3})$. Contracting this sequence of equalities, we conclude that $\text{QE}\Sigma_i(\frac{2}{3}, \frac{1}{3})$ equals our original class $\text{QE}\Sigma_i(c, s)$. ◀

5 PH and QCPH Are Contained in QPH

We prove that $\text{QCPH} \subseteq \text{QPH}$. While this result is what one might expect, proving this containment was left as an open question by Gharibian et al. [11]. It is trivial to see that $\text{PH} \subseteq \text{QCPH}$, and, combining these two containments, we have $\text{PH} \subseteq \text{QCPH} \subseteq \text{QPH}$, establishing that quantifying over unentangled quantum proofs is at least as powerful as quantifying over classical proofs.

The central challenge in proving that $\text{QCPH} \subseteq \text{QPH}$ is that the proofs in QPH are allowed to be quantum states, which, upon measurement, give rise to a distribution over classical strings. A flawed idea is to simply measure the quantum proofs to get classical proofs, and then run the QCPH verification protocol with no modifications. Suppose, however, that Alice has a winning strategy in the QCPH protocol, so she always has a winning response to any classical proof that Bob sends. When simulating this in QPH , Bob can instead send a *quantum state* – a superposition over many classical proofs – preventing Alice from sending an optimal response. In particular, Alice may not know which response to send, since she does not know which classical proof the verifier will observe upon measurement.

We prevent this potential cheating by requiring each player to send multiple copies of each of their proofs. We prove that this is enough to force both players to send classical strings with high probability.

► **Theorem 26.** $\text{PH} \subseteq \text{QCPH} \subseteq \text{QPH}$.

The exact error parameters for Theorem 26 are stated in Equation (3) below. In particular, the reduction is only capable of producing a QPH instance with a constant promise gap. However, the containment does hold for any QCPH instance with at least an inverse-polynomial promise gap, due to known error reduction for QCPH [11].

Proof. Consider any level $\text{QC}\Sigma_i$ of QCPH . We show that for any integer $k \geq 1$,

$$\text{QC}\Sigma_i(c, s) \subseteq \text{QC}\Sigma_{2ki}(c(1 - 2^{-k}), s + 2^{-k}(1 - s)). \quad (3)$$

We simulate any $\text{QC}\Sigma_i$ protocol in $\text{Q}\Sigma_{2ki}$ as follows. After the first $2k$ turns, the verifier has k proofs from Alice and k proofs from Bob, and the verifier discards all k proofs from Bob. For the next $2k$ turns, the verifier repeats this process, except they keep Bob's proofs rather than Alice's, which we denote by $\sigma_{1,1}, \dots, \sigma_{1,k}$. This is repeated i times in total until all $2ki$ turns are over. At the end of the game, the verifier has kept the following ki proofs:

$$\rho_{1,1}, \dots, \rho_{1,k}, \sigma_{1,1}, \dots, \sigma_{1,k}, \rho_{2,1}, \dots, \rho_{2,k}, \dots$$

For each chunk of k proofs, the verifier measures each quantum state in the standard basis to get k classical strings. If all k classical strings are equal, we say that the player passed the check, and failed otherwise. If a player fails any check, then the other player is declared the winner. If both players pass all checks, then the verifier keeps one copy of each classical proof from each chunk and runs the QCPH verification procedure to determine the winner.

Let $A = (A_{\text{yes}}, A_{\text{no}})$ be a promise problem in $\text{QC}\Sigma_i(c, s)$, and let x be some fixed input. If $x \in A_{\text{yes}}$, then Alice has no incentive to cheat and so we refer to her as the honest prover, while if $x \in A_{\text{no}}$, then we consider Bob the honest prover. We will define a strategy for the honest prover and show that no matter the strategy of the dishonest prover, the honest prover will win high probability. In particular, the honest prover's strategy will be to always send classical proofs, and when replying to a dishonest prover's proof $\rho = \sum_j p_j |j\rangle\langle j|$, the honest prover will respond as if only the string \hat{j} with the maximum probability $p_{\hat{j}}$ was sent (we arbitrarily choose to break ties by lexicographic order).

If the dishonest prover fails any check, they lose, so we assume now that the dishonest prover passes every check. Then, since both provers pass every check, the verifier has the i classical proofs y_1, \dots, y_i , where the proofs with odd indices are from Alice and the others are from Bob. In one case, suppose that each of the dishonest prover's moves turns out to be as the honest prover expected. Then the situation is identical to the original QCPH instance, and so the honest prover wins with the probability of the original protocol.

In the second case, at least one chunk of k proofs (sampled independently from k distributions) are equal to each other but not to the proof \hat{j} expected by the honest prover. Any string besides \hat{j} has $p_j \leq 1/2$, so the probability of this case occurring, with all k samples matching, is at most 2^{-k} .

Therefore, in the QPH protocol, if $x \in A_{\text{yes}}$, Alice wins with probability at least $c(1 - 2^{-k})$. If $x \in A_{\text{no}}$, then Bob wins with probability at least $(1 - s)(1 - 2^{-k})$, so Alice wins with probability at most

$$1 - (1 - s)(1 - 2^{-k}) = s + 2^{-k}(1 - s).$$

To summarize, the dishonest prover is unable to affect the outcome of the game with more than a small probability. We conclude that $\text{QC}\Sigma_i \subseteq \text{Q}\Sigma_{2ki}$, and therefore $\text{QCPH} \subseteq \text{QPH}$. ◀

6 Distribution Hierarchies

We introduce another generalization of the polynomial hierarchy where the provers send probability distributions over bit strings. This gives rise to two new hierarchies: the distributional polynomial hierarchy **DistributionPH** and its quantum analogue **DistributionQCPH**, which is the same as **DistributionPH** but with a quantum verifier. We will focus primarily on **DistributionPH** since the techniques used to analyze **DistributionPH** will work for **DistributionQCPH** as well.

6:18 The Entangled Quantum Polynomial Hierarchy Collapses

DistributionPH is similar to all of the hierarchies studied in this work. In DistributionPH, the distributions are public (the provers have full knowledge of the distributions that have been sent), but none of the distributions are sampled until every distribution has been sent. One can think of this as a non-interactive game, where the players use public, mixed strategies. Importantly, the distributions are not correlated across rounds.

While DistributionPH is a classical complexity class, our motivation for studying it is to further understand the quantum polynomial hierarchies. In particular, DistributionPH involves proofs that are classical mixtures of bit strings. This complements pureQPH, where the proofs are quantum superpositions of bit strings, and QPH, where the proofs are both (classical mixtures of quantum superpositions). Does the computational power of the polynomial hierarchy increase when the proofs only involve classical probability distributions? Or does the increased computational power come only from the quantum superposition allowed in QPH and pureQPH? In this section, we resolve these questions.

► **Theorem 27.** DistributionPH = PH.

That is, if the proofs are distributions over classical proofs, PH does not increase in power. The proof of Theorem 27 relies on a technical lemma that says the distributions sent in DistributionPH can be sparse and uniform. This lemma generalizes a result due to Lipton and Young [24] and Althöfer [3].

In the remainder of this section, we will formally define DistributionPH, prove the technical lemma, and prove Theorem 27. Finally, we will discuss DistributionQCPH (the same as DistributionPH but with a quantum verifier) and the power of classical versus quantum proofs.

We begin by formally defining DistributionPH. Let \mathcal{D}_m denote the set of all probability distributions over $\{0, 1\}^m$. For a computation M which takes length- m strings as input and a distribution $\rho \in \mathcal{D}_m$, let $M(\rho)$ implicitly refer to $M(y)$ for $y \sim \rho$, and any probability or expectation expressed in terms of $M(\rho)$ implicitly incorporates this sampling.

► **Definition 28** (*i -th level of the distribution polynomial hierarchy (Distribution Σ_i)*). *A promise problem $L = (L_{yes}, L_{no})$ is in Distribution $\Sigma_i(c, s)$ for polynomial-time computable functions $c, s : \mathbb{N} \rightarrow [0, 1]$ if there exists a classical polynomial-time randomized Turing Machine M such that*

- *Completeness: $x \in L_{yes} \Rightarrow \exists \rho_1 \forall \rho_2 \exists \rho_3 \dots Q_i \rho_i$ such that $\Pr [M(\rho_1, \dots, \rho_i) = 1] \geq c$,*
 - *Soundness: $x \in L_{no} \Rightarrow \forall \rho_1 \exists \rho_2 \forall \rho_3 \dots \overline{Q_i} \rho_i$ such that $\Pr [M(\rho_1, \dots, \rho_i) = 1] \leq s$,*
- where each ρ_k is a distribution in \mathcal{D}_m for some polynomially-bounded m , and each ρ_k is independent. Q_i is \exists if i is odd and \forall otherwise, and $\overline{Q_i}$ is the complement of Q_i . When the completeness/soundness parameters are not specified, define

$$\text{Distribution}\Sigma_i := \bigcup_{c-s \in \Omega(1)} \text{Distribution}\Sigma_i(c, s). \quad \lrcorner$$

► **Definition 29** (The distribution polynomial hierarchy (DistributionPH)). *The distribution polynomial hierarchy is defined as*

$$\text{DistributionPH} = \bigcup_{i=0}^{\infty} \text{Distribution}\Sigma_i. \quad \lrcorner$$

We make a few comments on our definition of DistributionPH. If we defined DistributionPH without the bounded-error condition (i.e., no error probability), then it would be equal to PH. We will also generally leave the input x implicit. Finally, if one prefers, they can equivalently

think of the provers sending quantum mixed states that are immediately measured in the computational basis (instead of probability distributions that are immediately sampled). This is why we choose to denote the probability distributions as ρ_i in our definition.

As we discussed in Section 3 for QEPH, one can think of **DistributionPH** as a game, where two competing provers take turns sending distributions over bit strings to a verifier. Then the verifier M draws one sample from each distribution and runs a polynomial-time randomized algorithm to determine a winner. Additionally, just like with QEPH, we can express the acceptance probability of the verifier as the following optimization problem:

$$\Pr[M \text{ accepts}] = \max_{\rho_1 \in \mathcal{D}_m} \min_{\rho_2 \in \mathcal{D}_m} \dots \mathbb{Q}^i \mathbf{E}[M(\rho_1, \dots, \rho_i)],$$

where \mathbb{Q}^i denotes max if i is odd and min otherwise. The expectation is over the randomness in the distributions ρ_1, \dots, ρ_i . Note that since $M(\rho_1, \dots, \rho_i)$ is a Bernoulli random variable, $\mathbf{E}[M(\rho_1, \dots, \rho_i)] = \Pr[M(\rho_1, \dots, \rho_i) = 1]$.

The distributions sent in **DistributionPH** are over $\{0, 1\}^m$ for some polynomially-bounded m , so, in general, the support can be exponentially large in m . We will prove a technical lemma that says the provers can send *uniform* distributions over $\text{poly}(m)$ bit strings without changing the outcome of the game too much.

► **Lemma 30.** *For any constant $k \in \mathbb{N}$ and any classical randomized Turing Machine M accepting k length- m inputs, if*

$$\max_{\rho_1 \in \mathcal{D}_m} \min_{\rho_2 \in \mathcal{D}_m} \max_{\rho_3 \in \mathcal{D}_m} \dots \mathbb{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1] = v,$$

then for any constant $\epsilon > 0$,

$$\max_{\rho_1 \in U_{t_k}} \min_{\rho_2 \in U_{t_{k-1}}} \max_{\rho_3 \in U_{t_{k-2}}} \dots \mathbb{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1] \in [v - k\epsilon, v + k\epsilon],$$

where $t_i := \lceil m^{2i}/2\epsilon^2 \rceil$, U_t denotes the set of uniform distributions over multi-sets of size at most t of strings in $\{0, 1\}^m$, and \mathbb{Q}^k denotes max if k is odd and min otherwise. The complement of this result also holds (i.e., when the sequence starts with min instead of max).

Proof. We will prove the claim by induction. The base case $k = 2$ is precisely [24, Theorem 2] (see also [3]). Our contribution is to generalize their result to larger k .

By way of induction, suppose the claim holds for $k - 1$, and consider an instance with k rounds:

$$v := \max_{\rho_1 \in \mathcal{D}_m} \min_{\rho_2 \in \mathcal{D}_m} \max_{\rho_3 \in \mathcal{D}_m} \dots \mathbb{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1].$$

Because the complement of this result (where a min is first instead of a max) follows in the same way, we omit the details.

Fix ρ_1 to a distribution that maximizes the acceptance probability (and think of ρ_1 as hardcoded into the input). Consider the inner $k - 1$ distributions ρ_2, \dots, ρ_k . By the inductive hypothesis, we can simplify these distributions to

$$\min_{\rho_2 \in U_{t_{k-1}}} \max_{\rho_3 \in U_{t_{k-2}}} \dots \mathbb{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1],$$

while only changing the acceptance probability v by $\pm(k - 1)\epsilon$. In particular, we have that

$$v' := \max_{\rho_1 \in \mathcal{D}_m} \min_{\rho_2 \in U_{t_{k-1}}} \max_{\rho_3 \in U_{t_{k-2}}} \dots \mathbb{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1] \in [v - (k - 1)\epsilon, v + (k - 1)\epsilon].$$

6:20 The Entangled Quantum Polynomial Hierarchy Collapses

We want to show that we can simplify the first distribution ρ_1 in a similar fashion. Specifically, we want to show

$$v'' := \max_{\rho_1 \in U_{t_k}} \min_{\rho_2 \in U_{t_{k-1}}} \max_{\rho_3 \in U_{t_{k-2}}} \dots \max_{\rho_k \in U_{t_1}} \mathbf{Q}^k \Pr[M(\rho_1, \dots, \rho_k) = 1] \in [v - k\epsilon, v + k\epsilon].$$

Observe that choosing ρ_1 from U_{t_k} instead of \mathcal{D}_m can only hurt the maximizing player. That is, the probability that M accepts can only *decrease*, so $v'' \leq v' + \epsilon \leq v + k\epsilon$ is trivial. All that remains is to show that $v'' \geq v - k\epsilon$. To prove this, it suffices to show that $v'' \geq v' - \epsilon$.

Let $\rho_1^* \in \mathcal{D}_m$ be a distribution that maximizes the acceptance probability of M . Form a multi-set S by drawing t_k independent samples from ρ_1^* . Consider a string $y \in S$. This gives rise to a random variable on the interval $[0, 1]$:

$$\mathbf{E}_{\rho_2, \dots, \rho_k} [M(y, \rho_2, \dots, \rho_k)],$$

where we are taking the expectation over optimal choices of ρ_2, \dots, ρ_k . In expectation over ρ_1^* , we have

$$\mathbf{E}_{y \sim \rho_1^*} \left[\mathbf{E}_{\rho_2, \dots, \rho_k} [M(y, \rho_2, \dots, \rho_k)] \right] = v'.$$

Therefore, by Hoeffding's inequality (Fact 7),

$$\Pr \left[\frac{1}{|S|} \sum_{y \in S} \mathbf{E}_{\rho_2, \dots, \rho_k} [M(y, \rho_2, \dots, \rho_k)] \leq v' - \epsilon \right] \leq \exp(-2t_k \epsilon^2).$$

To complete the proof, we must count the number of sequences of distributions the minimizing player can send. The minimizing player sends at most $k/2$ of the distributions ρ_2, \dots, ρ_k , each of which is a uniform distribution over at most t_{k-1} -sized subsets of $\{0, 1\}^m$. Therefore, in total, there are at most

$$\left(\sum_{i=1}^{t_{k-1}} \binom{2^m}{i} \right)^{k/2} \leq \left(\sum_{i=1}^{t_{k-1}} 2^{im} \right)^{k/2} \leq (t_{k-1} 2^{m t_{k-1}})^{k/2} = t_{k-1}^{k/2} 2^{k m t_{k-1}/2}$$

possible sequences. We want to choose t_k so that

$$\exp(-2t_k \epsilon^2) < \frac{1}{t_{k-1}^{k/2} 2^{k m t_{k-1}/2}}, \quad (4)$$

which would imply that strictly less than 1 of the minimizing player's sequences of distributions can decrease v' by more than ϵ . Or, more directly, it would imply that there are no sequences the minimizing player can send to decrease v' by more than ϵ . We will show that choosing $t_k = m^{2k}/2\epsilon^2$ suffices. Substituting the definitions of t_k and t_{k-1} , Equation (4) becomes

$$\exp(-m^{2k}) < \frac{\epsilon^k}{m^{k(k-1)}} 2^{\frac{k}{2} - \frac{k m^{2k-1}}{4\epsilon^2}} \iff \exp(-m^{2k}) \frac{m^{k(k-1)}}{\epsilon^k} 2^{\frac{k m^{2k-1}}{4\epsilon^2} - \frac{k}{2}} < 1. \quad (5)$$

We show that the inequality in Equation (5) holds, which proves that our setting of t_k is correct.

$$\begin{aligned} \exp(-m^{2k}) \frac{m^{k(k-1)}}{\epsilon^k} 2^{\frac{k m^{2k-1}}{4\epsilon^2} - \frac{k}{2}} &< \exp(-m^{2k}) m^{k^2} 2^{\frac{k m^{2k-1}}{4\epsilon^2} - \frac{k}{2}} \\ &< m^{k^2} 2^{\frac{k m^{2k-1}}{4\epsilon^2} - \frac{k}{2} - m^{2k}} \\ &= m^{k^2} 2^{m^{2k-1} \left(\frac{k}{4\epsilon^2} - \frac{k}{2m^{2k-1}} - m \right)} \\ &< m^{k^2} 2^{-m^{2k-1}} \\ &< 1. \end{aligned}$$

The first inequality holds because $m^k > \epsilon^{-k}$ for constant $\epsilon > 0$. The second-to-last inequality holds because $(\frac{k}{4\epsilon^2} - \frac{k}{2m^{2k-1}} - m) < -1$ for constant $\epsilon > 0$.

We conclude that $v'' \geq v' - \epsilon \geq v - k\epsilon$, which completes the proof. \blacktriangleleft

We can now prove that $\text{DistributionPH} = \text{PH}$.

Proof of Theorem 27. $\text{PH} \subseteq \text{DistributionPH}$ follows from the proof that $\text{PH} \subseteq \text{QPH}$. This only achieves containment in DistributionPH with constant promise gap, and it puts the k -th level of PH in some higher level of DistributionPH (see Theorem 26 for more detail).

To show $\text{DistributionPH} \subseteq \text{PH}$, we use Lemma 30. Set $\epsilon < \frac{1}{12k}$. For $\text{Distribution}\Sigma_k$, Lemma 30 implies that

$$\max_{\rho_1 \in U_{t_k}} \min_{\rho_2 \in U_{t_{k-1}}} \max_{\rho_3 \in U_{t_{k-3}}} \dots \mathbb{Q}^k \Pr [M(\rho_1, \dots, \rho_k)] \in [v - k\epsilon, v + k\epsilon] \subseteq \left[v - \frac{1}{12}, v + \frac{1}{12} \right].$$

Given the $\text{Distribution}\Sigma_k$ promise gap of $\frac{2}{3}, \frac{1}{3}$, this modified game has a promise gap of $\frac{7}{12}, \frac{5}{12}$.

We simulate this in PH as follows. To send the distribution ρ_i , the prover sends every string in the support of ρ_i , which is only $\text{poly}(n)$ many bits by Lemma 30. The verifier can then take the list of strings and sample one uniformly at random. This completes the proof since PH can simulate randomness [30, 23]. \blacktriangleleft

One can also define DistributionQCPH in the same way, and it follows from Theorem 27 that this class is equal to QCPH .

► **Corollary 31.** $\text{DistributionQCPH} = \text{QCPH}$.

The *only* difference between DistributionQCPH and pureQPH is that the former involves proofs that are classical distributions over bit strings and the latter involves proofs that are quantum superpositions over bit strings. $\text{DistributionQCPH} = \text{QCPH}$ is in the counting hierarchy [11], while the best known upper bound for pureQPH is EXP^{PP} [2] and it contains $\text{QMA}(2)$ and QPH . The conceptual takeaway is that it is only the quantum superposition in the proofs that gives the quantum hierarchies more computational power.

We also remark that if one allows the distributions in DistributionPH and DistributionQCPH to be correlated, then the techniques in Lemma 22 can be used to collapse the resulting hierarchies to the second level. The correlated version of DistributionPH collapses to S_2P . The correlated version of DistributionQCPH collapses to a quantum-classical version of $\text{QRG}(1)$, which, to our knowledge, has never been studied.

References

- 1 Scott Aaronson, DeVon Ingram, and William Kretschmer. The Acrobatics of BQP. In *37th Computational Complexity Conference (CCC 2022)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 20:1–20:17, 2022. doi:10.4230/LIPIcs.CCC.2022.20.
- 2 Avantika Agarwal, Sevag Gharibian, Venkata Koppula, and Dorian Rudolph. Quantum Polynomial Hierarchies: Karp-Lipton, error reduction, and lower bounds, 2024. arXiv:2401.01633.
- 3 Ingo Althöfer. On sparse approximations to randomized strategies and convex combinations. *Linear Algebra and its Applications*, 199:339–355, 1994. Special Issue Honoring Ingram Olkin. doi:10.1016/0024-3795(94)90357-3.
- 4 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. doi:10.1017/CB09780511804090.
- 5 Roozbeh Bassirian, Bill Fefferman, and Kunal Marwaha. Quantum Merlin-Arthur and proofs without relative phase, 2023. arXiv:2306.13247.


- 6 Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001. doi:10.1103/PhysRevLett.87.167902.
- 7 Chirag Falor, Shu Ge, and Anand Natarajan. A Collapsible Polynomial Hierarchy for Promise Problems, 2023. arXiv:2311.12228.
- 8 Uriel Feige and Joe Kilian. Making Games Short (Extended Abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 506–516, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258644.
- 9 J. Feigenbaum, D. Koller, and P. Shor. A Game-Theoretic Classification of Interactive Complexity Classes. In *Proceedings of Structure in Complexity Theory. Tenth Annual IEEE Conference*, pages 227–237, 1995. doi:10.1109/SCT.1995.514861.
- 10 Sevag Gharibian and Julia Kempe. Hardness of approximation for quantum problems. In *International Colloquium on Automata, Languages, and Programming*, pages 387–398. Springer, 2012. doi:10.1007/978-3-642-31594-7_33.
- 11 Sevag Gharibian, Miklos Santha, Jamie Sikora, Aarthi Sundaram, and Justin Yirka. Quantum generalizations of the Polynomial Hierarchy with applications to QMA(2). *Computational Complexity*, 31(2):13, 2022. doi:10.1007/s00037-022-00231-8.
- 12 Soumik Ghosh and John Watrous. Complexity limitations on one-turn quantum refereed games. *Theory of Computing Systems*, 67(2):383–412, 2023. doi:10.1007/s00224-022-10105-9.
- 13 Gus Gutoski and John Watrous. Quantum interactive proofs with competing provers. In *STACS 2005: 22nd Annual Symposium on Theoretical Aspects of Computer Science*, pages 605–616. Springer, 2005. doi:10.1007/978-3-540-31856-9_50.
- 14 Gus Gutoski and John Watrous. Toward a General Theory of Quantum Games. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, pages 565–574, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1250790.1250873.
- 15 Gus Gutoski and Xiaodi Wu. Parallel Approximation of Min-Max Problems. *Computational Complexity*, 22:385–428, 2013. doi:10.1007/s00037-013-0065-9.
- 16 Aram W. Harrow and Ashley Montanaro. Testing Product States, Quantum Merlin-Arthur Games and Tensor Optimization. *J. ACM*, 60(1), 2013. doi:10.1145/2432622.2432625.
- 17 Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. QIP = PSPACE. *Journal of the ACM (JACM)*, 58(6):1–27, 2011. doi:10.1145/2049697.2049704.
- 18 Rahul Jain and John Watrous. Parallel Approximation of Non-interactive Zero-sum Quantum Games. In *24th Annual IEEE Conference on Computational Complexity*, pages 243–253, 2009. doi:10.1109/CCC.2009.26.
- 19 Fernando Granha Jeronimo and Pei Wu. The power of unentangled quantum proofs with non-negative amplitudes. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 1629–1642, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3564246.3585248.
- 20 Alexei Kitaev and John Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 608–617, 2000. doi:10.1145/335305.335387.
- 21 Alexei Y. Kitaev, Alexander Shen, and Mikhail N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Soc., 2002. doi:10.1090/gsm/047.
- 22 Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos Papadimitriou. Total Functions in the Polynomial Hierarchy. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 44:1–44:18, 2021. doi:10.4230/LIPIcs.ITCS.2021.44.
- 23 Clemens Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17(4):215–217, 1983. doi:10.1016/0020-0190(83)90044-3.
- 24 Richard J. Lipton and Neal E. Young. Simple Strategies for Large Zero-Sum Games with Applications to Complexity Theory. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 734–740, 1994. doi:10.1145/195058.195447.

- 25 Chris Marriott and John Watrous. Quantum Arthur–Merlin Games. *Computational Complexity*, 14(2):122–152, 2005. doi:10.1007/s00037-005-0194-x.
- 26 A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *13th Annual Symposium on Switching and Automata Theory (SWAT 1972)*, pages 125–129, 1972. doi:10.1109/SWAT.1972.29.
- 27 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. doi:10.1017/CB09780511976667.
- 28 Adi Shamir. $IP = PSPACE$. *J. ACM*, 39(4):869–877, 1992. doi:10.1145/146585.146609.
- 29 Maurice Sion. On General Minimax Theorems. *Pacific Journal of Mathematics*, 1958. doi:10.2140/pjm.1958.8.171.
- 30 Michael Sipser. A Complexity Theoretic Approach to Randomness. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 330–335. Association for Computing Machinery, 1983. doi:10.1145/800061.808762.
- 31 Larry J. Stockmeyer. The Polynomial-Time Hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976. doi:10.1016/0304-3975(76)90061-X.
- 32 Lieuwe Vinkhuijzen. A Quantum Polynomial Hierarchy and a Simple Proof of Vyalyi’s Theorem. Master’s thesis, Leiden University, 2018. URL: <https://theses.liacs.nl/1505>.
- 33 John Watrous. *The Theory of Quantum Information*. Cambridge University Press, 2018. doi:10.1017/9781316848142.
- 34 Tomoyuki Yamakami. Quantum NP and a Quantum Hierarchy. In *Foundations of Information Technology in the Era of Networking and Mobile Computing*, volume 96 of *IFIP — The International Federation for Information Processing*, pages 323–336, Boston, MA, 2002. Springer. doi:10.1007/978-0-387-35608-2_27.

Polynomial Pass Semi-Streaming Lower Bounds for k -Cores and Degeneracy

Sepehr Assadi   

Cheriton School of Computer Science, University of Waterloo, Canada

Prantar Ghosh   


Department of Computer Science, Georgetown University, Washington, DC, USA

Bruno Loff   

Department of Mathematics and LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

Parth Mittal   

University of Waterloo, Canada

Sagnik Mukhopadhyay   

University of Sheffield, UK

Abstract

The following question arises naturally in the study of graph streaming algorithms:

Is there any graph problem which is “not too hard”, in that it can be solved efficiently with total communication (nearly) linear in the number n of vertices, and for which, nonetheless, any streaming algorithm with $\tilde{O}(n)$ space (i.e., a semi-streaming algorithm) needs a polynomial $n^{\Omega(1)}$ number of passes?

Assadi, Chen, and Khanna [STOC 2019] were the first to prove that this is indeed the case. However, the lower bounds that they obtained are for rather non-standard graph problems.

Our first main contribution is to present the first polynomial-pass lower bounds for natural “not too hard” graph problems studied previously in the streaming model: **k -cores** and **degeneracy**. We devise a novel communication protocol for both problems with near-linear communication, thus showing that k -cores and degeneracy are natural examples of “not too hard” problems. Indeed, previous work have developed single-pass semi-streaming algorithms for approximating these problems. In contrast, we prove that any semi-streaming algorithm for *exactly* solving these problems requires (almost) $\Omega(n^{1/3})$ passes.

The lower bound follows by a reduction from a generalization of the **hidden pointer chasing (HPC)** problem of Assadi, Chen, and Khanna, which is also the basis of their earlier semi-streaming lower bounds.

Our second main contribution is improved **round-communication** lower bounds for the underlying communication problems at the basis of these reductions:

- We improve the previous lower bound of Assadi, Chen, and Khanna for HPC to achieve optimal bounds for this problem.
- We further observe that all current reductions from HPC can also work with a generalized version of this problem that we call **MultiHPC**, and prove an even stronger and optimal lower bound for this generalization.

These two results collectively allow us to improve the resulting pass lower bounds for semi-streaming algorithms by a polynomial factor, namely, from $n^{1/5}$ to $n^{1/3}$ passes.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Lower bounds and information complexity; Theory of computation → Graph algorithms analysis

Keywords and phrases Graph streaming, Lower bounds, Communication complexity, k -Cores and degeneracy

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.7

Related Version *Full Version:* <http://arxiv.org/abs/2405.14835> [9]



© Sepehr Assadi, Prantar Ghosh, Bruno Loff, Parth Mittal, and Sagnik Mukhopadhyay; licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 7; pp. 7:1–7:16



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Funding *Sepehr Assadi*: Supported in part by a Sloan Research Fellowship, an NSERC Discovery Grant, a University of Waterloo startup grant, and a Faculty of Math Research Chair grant.

Prantar Ghosh: Supported in part by NSF under award 1918989. Part of this work was done while he was at DIMACS, Rutgers University, supported in part by a grant (820931) to DIMACS from the Simons Foundation.

Bruno Loff: Funded by the European Union (ERC, HOFGA, 101041696). 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. Neither the European Union nor the granting authority can be held responsible for them. He was also supported by FCT through the LASIGE Research Unit, ref. UIDB/00408/2020 and ref. UIDP/00408/2020, and by CMAFcIO, FCT Project UIDB/04561/2020, <https://doi.org/10.54499/UIDB/04561/2020>.

Parth Mittal: Supported by a David R. Cheriton Graduate Scholarship and Sepehr Assadi’s Sloan Research Fellowship, an NSERC Discovery grant, and a startup grant from University of Waterloo.

Sagnik Mukhopadhyay: Partially funded by the UKRI New Investigator Award, ref. EP/X03805X/1.

Acknowledgements The first named author would like to thank Yu Chen and Sanjeev Khanna for their collaboration in [7] that was the starting point of this project and Madhu Sudan for helpful conversations.

1 Introduction

Graph streaming algorithms process their inputs by making one or few passes over the edges of an input graph using limited memory. Algorithms that use space proportional to n , the number of vertices, are called *semi-streaming* algorithms. Since their introduction by [36], graph streaming algorithms have become one of the main theoretical research areas on processing massive graphs. We refer the interested reader to [53] for an introductory survey of earlier results on this topic.

In this work, we prove a polynomial-pass lower bound for any graph streaming algorithm that computes k -cores or degeneracy of a given graph. Our result is of interest from the point of view of proving strong lower bounds in the graph streaming model in addition to their direct implications for these two specific problems.

1.1 Polynomial Pass Lower Bounds in Graph Streams

Even though the study of multi-pass graph streaming algorithms started hand in hand with single-pass algorithms in [36], our understanding of powers and limitations of multi-pass algorithms, even for most basic problems, lags considerably behind. On one hand, for a problem like minimum cut, we have algorithms that in just $\tilde{O}(n)$ space and two passes can solve the problem *exactly* [8]¹ (see [7, Table 1] for a list of several such results). On the other hand, for some other basic problems such as undirected shortest path, directed reachability, and bipartite matching, the best known semi-streaming algorithms require $O(n^{1/2})$ [23], $n^{1/2+o(1)}$ [10, 51], and $n^{3/4+o(1)}$ [10] passes, respectively; yet, despite significant efforts, the best lower bound for any of these problems is still (even slightly below) $\Omega(\log n)$ passes [14, 22, 25, 40].

A key reason behind our weaker understanding of multi-pass streaming algorithms can be attributed to the lack of techniques for proving *super-logarithmic* pass lower bounds for semi-streaming algorithms. At this point, such lower bounds are only known for a handful

¹ See [61] for an implicit algorithm with the same bounds and [55] for the extension to weighted cuts in $O(\log n)$ passes.

of problems: clique and independent set [41], dominating set [1], Hamiltonian path [15], maximum cut [15, 46], vertex cover and coloring [3], exact Boolean CSPs [46], triangle detection [17, 58], and diameter computation [37]. Although, for all these problems, we can actually prove close-to- n pass lower bounds. Let us examine this dichotomy.

A quick glance at the list of problems above may suggest an intuitive difference between these problems and the ones like reachability or shortest path: the above list consists of problems that are computationally hard in a classical sense², suggesting that we are dealing with a “harder” class of problems in their case. While this intuition should not be taken as a formal evidence – as classical computational hardness does *not* imply streaming lower bounds (which are unconditional and information-theoretic) – [7] showed that one can also formally explain this dichotomy.

In particular, [7] observed that these strong streaming lower bounds happen only when the communication complexity of the problem at hand is $\Omega(n^2)$. Such a high lower-bound on the communication complexity *immediately* gives an $\tilde{\Omega}(n)$ -pass lower bound for semi-streaming algorithms via standard reductions. Whereas, for almost all problems of interest in the semi-streaming model, including shortest path, reachability, and bipartite matching, we already know an $\tilde{O}(n)$ communication upper bound³ (the protocol for bipartite matching was only discovered in [20] after the work of [7], but $\tilde{O}(n^{3/2})$ communication protocols were known already [31, 42]). We refer the reader to [7, Section 1.1] for more context regarding these observations and prior techniques for $o(\log n)$ pass lower bounds.

Toward Stronger Streaming Lower Bounds

A natural question in light of these observations, already posed in [7], is the following:

Motivating question. *Is there any graph problem which is “not too hard”, in that it can be solved efficiently with communication (nearly) linear in the number n of vertices, and for which, nonetheless, any semi-streaming algorithm needs a polynomial $n^{\Omega(1)}$ number of passes?*

To address this question, [7] introduced a new (four-player) communication problem called **Hidden Pointer Chasing (HPC)**, which acts as a cross between *Set-Intersection* and *Pointer Chasing* problems, which are the main problems for, respectively, proving $\Omega(n^2)$ communication lower bounds on graphs, and $o(\log n)$ -pass lower bounds for semi-streaming algorithms.

Roughly speaking, the HPC problem is defined as follows. There are four players paired into two groups. Each pair of players inside a group shares m instances of the Set-Intersection problem on m elements (each of the two players holds a subset of $[m]$ and they need to identify the unique intersecting element). The intersecting element in each instance of each group “points” to an instance in the other group. The goal is to start from a fixed instance, follow these pointers for a fixed number of steps, and then return the last element reached. See (full version [9], Definition 3.3) for the formal description.

This problem admits an efficient communication protocol with no limit on its number of rounds, but [7] showed that any r -round protocol that aims to find the $(r + 1)$ -th pointer in HPC requires $\Omega(m^2/r^2)$ communication. This places HPC squarely in the middle of previous

² These are standard NP-hard problems or admit some fine-grained hardness (for the latter two) [47, 60].

³ This perhaps can be seen as this: a problem whose (unbounded round) communication complexity is already high has almost no place in the streaming model, which is a much weaker model algorithmically.

techniques and quite suitable for performing reductions to prove streaming lower bounds even for not-too-hard graph problems. Using this, [7] proved the first set of polynomial-pass graph streaming lower bounds in this class of problems: computing *lexicographically-first maximal independent set (LFMIS)* and *s - t minimum cut* on graphs with exponential edge-capacities both require $\tilde{\Omega}(n^{1/5})$ passes to be solved by semi-streaming algorithms.

Despite the significant advances on multi-pass streaming lower bounds in the last couple of years [2, 11–14, 22, 25–27, 46], there is still no other known (not-too-hard) problem that admits a polynomial-pass lower bounds beside those of [7]. In addition, it is worth mentioning that, strictly speaking, neither LFMIS nor the version of s - t minimum cut in [7] completely fit the premise of our original question: LFMIS is not purely a graph problem as it is not invariant under labeling of the vertices, and s - t minimum cut studied in [7] involves (i) making the non-standard assumption of exponential capacities, and (ii) even for unit-capacity graphs, is not known to admit an $\tilde{O}(n)$ communication protocol (see [20]).

We prove polynomial-pass lower bounds for two natural graph problems, k -cores and degeneracy, by reduction from a harder variant of the HPC problem which we call MultiHPC. We also present novel $\tilde{O}(n)$ communication protocols for these two problems. These two results together give us the first natural instances of a positive answer to our motivating question.

These results further demonstrate the power of reductions from the HPC problem, as a technique for proving strong lower bounds in the graph streaming model, which are beyond the reach of other techniques. With this in mind, we improve the lower bound of [7] for the HPC problem to an optimal bound of $\Omega(m^2/r)$ communication, i.e., we improve the known bound by a factor of r . This contribution alone results in a polynomial improvement in the number of passes, for all lower bounds that follow via reductions from HPC (for instance, it immediately improves the bounds of [7] for LFMIS and exponential-capacity s - t minimum cut to $\tilde{\Omega}(n^{1/4})$ passes).

But, as it turns out, all the known lower-bounds that follow by reduction from HPC also follow by reduction from MultiHPC. For this variant, we can prove an $\Omega(m^2)$ lower-bound for r rounds (since the input size for MultiHPC is $r \cdot m^2$), and this translates to an improved semi-streaming lower-bound of $\tilde{\Omega}(n^{1/3})$ passes for all of the above problems.

1.2 k -Cores and Degeneracy in Graph Streams

For any undirected graph $G = (V, E)$ and integer $k \geq 1$, a k -core in G is a maximal set S of vertices such that the induced subgraph of G on S , denoted by $G[S]$, has a minimum degree of at least k . In other words, any vertex in S has at least k other neighbors in S .

k -Cores provide a natural notion of well-connectedness in massive graphs, and as such, computing k -cores (and more generally k -core decompositions; see, e.g., [50]) has been widely studied in databases [21, 28, 49], social networks [29, 30, 44], machine learning [6, 33, 39], among others [38, 48, 62].

As a result, in recent years, there has been a rapidly growing body of work on computing k -cores on massive graphs in parallel and streaming models of computation [29, 30, 33, 39, 50, 62]. In particular, [33] presented a single-pass algorithm that for any $\varepsilon > 0$, computes a $(1 - \varepsilon)$ -approximation of every k -core in G (i.e., obtains a $(1 - \varepsilon)$ -approximate k -core decomposition) in $\tilde{O}(n/\varepsilon^2)$ space (see also [62] for an earlier streaming algorithm and [39] for a closely related parallel algorithm).

The *degeneracy* of a graph $G = (V, E)$, denoted by $\kappa(G)$, is the largest integer $k \geq 0$ such that G contains a non-empty k -core. The simple greedy algorithm that at every step peels off the smallest degree vertex results in the so-called *degeneracy ordering* of G and $\kappa(G)$

is equal to the largest degree of a vertex removed in this peeling process [52]. Degeneracy is a standard measure of uniform sparsity and is closely related to other such notions like arboricity (which is within a factor 2 of degeneracy). Moreover, computing degeneracy is a subroutine for approximating various other problems such as arboricity [5], densest subgraph [24], $(\kappa + 1)$ coloring [32].

The degeneracy problem, and closely related uniform-sparsity measures such as densest subgraph, have also been studied extensively in the graph streaming literature [4, 16, 18, 19, 34, 35, 54]. In particular, [34] provided an $O(\log n)$ -pass semi-streaming algorithm that outputs a constant factor approximation to degeneracy, and [35] subsequently improved this to a single-pass $(1 - \varepsilon)$ -approximation in $\tilde{O}(n/\varepsilon^2)$ space (see also [54] for densest subgraph and [4, 18] for degeneracy coloring).

In terms of lower bounds, [18] prove that any *single-pass* streaming algorithm that computes the exact value of degeneracy or approximates it within an additive factor of λ requires $\Omega(n^2)$ space or $\Omega(n^2/\lambda^2)$ space respectively. Our *polynomial-pass* lower bounds for k -cores and degeneracy, now in a very strong sense, rule out the possibility of extending any prior semi-streaming algorithms computing near-optimal solutions to these problems, to compute *exactly* optimal solutions.

1.3 Our Results

We give an informal presentation of our results here. The details can be found in the full version [9]. The first main result is our polynomial-pass lower bound for k -core computation and degeneracy.

► **Result 1** (Formalized in full version [9], Theorem 5.1). For any integer $p \geq 1$, any p -pass streaming algorithm for computing the degeneracy of an input n -vertex graph requires $\tilde{\Omega}(n^2/p^3)$ space. In particular, any semi-streaming algorithm for the problem requires $\tilde{\Omega}(n^{1/3})$ passes.

Moreover, the same lower bounds also apply to the algorithms that given any integer $k \geq 1$, can check whether or not the input graph contains a non-empty k -core.

Result 1 provides a strongly negative answer to the question of obtaining semi-streaming algorithms for exact computation of degeneracy and k -cores in a small number of passes. We obtain Result 1 via a detailed and technical reduction, presented in Section 5 of the full version [9], from a variant of the Hidden Pointer Chasing (HPC) problem of [7], which we call Boolean Multilayer Hidden Pointer Chasing (BMHPC).

In a standard HPC problem, we are given m instances of m -bit Set-Intersection ($O(m^2)$ bits in total) and interpret the intersection point of each instance as pointing to a different instance among these m . We then wish to know the position we end up in after following $r + 1$ pointers. In a Boolean variant, we only care to know the *parity* of the position we end up in. In a Multilayer variant, we are given r different layers, each layer with its own m instances (rm^2 bits in total), where we think of the intersection points at each layer as pointing to some instance in the next layer, and wish to know where we end up in the last layer by following these pointers.

In addition to the reduction from BMHPC, in Section 6 of the full version [9], we present a novel and non-trivial communication protocol that finds the degeneracy ordering (and thus degeneracy itself) and non-empty k -cores for any given k , using only $\tilde{O}(n)$ communication. This communication upper bound thus places the k -core and degeneracy problems as perfect

illustrations of a positive answer to our and [7]’s motivating question outlined earlier: namely, problems that prior techniques could not have proven any lower bound beyond $\log n$ passes. Result 1 thus constitutes the first set of natural graph problems with polynomial pass lower bounds for semi-streaming algorithms.

Our second main contribution is providing optimal lower bounds for the HPC problem and all its variants (Boolean, Multilayer, and Boolean Multilayer). A communication lower-bound of $\Omega(\frac{m^2}{r^2})$ was previously known for $(r - 1)$ -round protocols computing the r -th pointer in a (single layer) HPC problem. We prove the following.

► **Result 2** (Formalized in full version [9], Theorem 4.2). For any integer $1 \leq r = O(\sqrt{m})$, any $(r - 1)$ -round protocol for computing the r -th pointer in the HPC problem on a universe of size m requires $\tilde{\Omega}(m^2/r)$ communication.

For any integer $r \geq 1$, any $(r - 1)$ -round protocol for computing the r -th pointer in the Multilayer HPC problem on a universe of size m requires $\tilde{\Omega}(m^2)$ communication.

Moreover, the same lower bounds hold for the Boolean versions of the above.

Result 2, by strengthening the lower bound of [7], allows us to prove polynomially stronger bounds on the number of passes of semi-streaming algorithms via reductions from HPC. As it turns out, every known reduction from HPC [7] can be easily converted to a reduction from MHPC. Our results thus imply improved pass lower bounds (from $\tilde{\Omega}(n^{1/5})$ to $\tilde{\Omega}(n^{1/3})$) for semi-streaming algorithms solving these problems. We capture this in the next corollary.

► **Corollary 1.** For any integer $p \geq 1$, any p -pass streaming algorithm for the following problems on n -vertex graphs requires $\tilde{\Omega}(n^2/p^3)$ space. In particular, any semi-streaming algorithm for these problems require $\tilde{\Omega}(n^{1/3})$ passes.

- Computing the minimum s - t cut value in a weighted graph (with exponential edge capacities)
- Computing the lexicographically-first maximal independent set (LFMIS) of an undirected graph

We obtain Result 2 by following the elegant analysis of pointer chasing problems due to [63] via the *triangular discrimination distance* between distributions, as opposed to more standard measures such as KL-divergence and total variation distance typically used in this context. This in turn requires extending the notion of “almost solving” for the Set-Intersection problem introduced by [7] (and further refined in [14]), to the triangular discrimination distance: roughly speaking, this corresponds to proving a lower bound for communication protocols that, instead of finding the intersecting element, change its distribution slightly from uniform distribution. The analysis in [7] measured this change by total variation distance, but now we need to do so by triangular discrimination distance instead. Finally, we prove a nearly-optimal lower bound on the communication-distance tradeoff for almost solving Set-Intersection in terms of the triangular discrimination distance.

2 Overview

In this version, we only present a high-level and informal overview of our techniques and proofs. All the technical details and formal proofs are available in the full version [9].

2.1 Hidden Pointer Chasing

The Multilayer Hidden Pointer Chasing (MHPC) problem, the starting point of our reductions, is defined as follows. The problem operates on two disjoint universes $\mathcal{X} = \{x_1, \dots, x_m\}$ and $\mathcal{Y} = \{y_1, \dots, y_m\}$. There are four players P_A, P_B, P_C, P_D , out of which P_A and P_B each hold rm subsets of \mathcal{Y} , called A_x^j and B_x^j for $x \in \mathcal{X}$ and $j \in [r]$, and P_C and P_D each hold rm subsets of \mathcal{X} , called C_y^j and D_y^j for $y \in \mathcal{Y}$ and $j \in [r]$, with the promise $|A_x^j \cap B_x^j| = 1$ and $|C_y^j \cap D_y^j| = 1$ for every $j \in [r]$, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. This means that each pair of sets of two of the players, e.g. A_x^j and B_x^j defines a pointer $\{y\} = A_x^j \cap B_x^j$, which we think of as pointing to the pair of sets in the next layer, C_y^{j+1} and D_y^{j+1} , belonging to the other two players. Following these pointers, and writing a singleton set as the element it contains, we define a sequence $z_0 = x_1, z_1 = A_{z_0}^1 \cap B_{z_0}^1, z_2 = C_{z_1}^2 \cap D_{z_1}^2, z_3 = A_{z_2}^3 \cap B_{z_2}^3, \text{ etc.}$ In the $\text{MHPC}_{m,r}$ problem, the players wish to learn z_r . In the $\text{BMHPC}_{m,r}$ problem, the players only need to learn one bit about z_r , that is, $b(z_r) := i \bmod 2$ where i is the index of z_r in \mathcal{X} or \mathcal{Y} . Now, there is a very obvious way of doing this in r rounds, if the correct pair of players start: the players just follow the pointers, solving the necessary Set-Intersection instances. This costs them r rounds with $O(m)$ bits of communication per round, for a total of rm bits. However, we will show:

► **Theorem 2.** *Any randomized protocol with less than r rounds, or even any randomized protocol with r rounds which is misaligned, in that the “wrong” pair of players starts to speak, cannot solve $\text{BMHPC}_{m,r}$ correctly with fewer than $\Omega(m^2)$ bits of communication.*

This theorem is proven by combining ideas from three different previous works: [7], [14], and [63]. But first, let us give an overall intuition for why it should be expected to hold.

In a misaligned r -round protocol for $\text{BMHPC}_{m,r}$, it is players P_C and P_D who begin the protocol by talking with each other. This means that the “wrong” pair of players begin to speak, in the sense that they wish to compute the value $\{z_1\} = A_1^1 \cap B_1^1$, but this instance is with P_A and P_B , so they have no way to do this. So the first round cannot say anything about z_1 : the best P_C and P_D can do is send some information about all of their Set-Intersection instances, without knowing which one is important. This means that each bit that P_C and P_D communicate with P_A and P_B in the first round can only reveal $\frac{1}{m}$ bits of information about the average instance. But now in the next round, P_A and P_B , although they know z_1 , cannot have learned much information about $C_{z_1}^2$ or $D_{z_1}^2$. But then, how can they say anything about $\{z_2\} = C_{z_1}^2 \cap D_{z_1}^2$? The difficult situation is now reversed! This “always one step behind” situation is similar to what happens for pointer chasing [57, 59, 63], except now the pointers are “hidden” behind set intersection instances.

A previous paper of Assadi, Chen and Khanna [7] showed a lower-bound of $\Omega(\frac{m^2}{r^2})$ for the (single layer) Hidden Pointer Chasing problem $\text{HPC}_{m,r}$, which is a version of $\text{MHPC}_{m,r}$ where all the layers are identical ($A_i^j, B_i^j, C_i^j, D_i^j$ is the same for all $j \in [r]$). The lower-bound was proven via an information-theoretic argument. They first show that any low-round protocol for HPC must be “almost solving” a set intersection instance on one of the rounds. They then show that this is impossible via an information complexity argument, akin to the lower-bound on the information complexity for set disjointness. However, a later paper by Assadi and Raz [14] directly showed that any protocol that “almost solves” set intersection can be used to obtain a protocol that exactly solves set intersection (hence the term “almost solving”). This would allow us to replace the *ad hoc* information complexity argument in [7], and instead appeal, in a black-box fashion, to a previously known lower-bound on the information complexity of Set-Intersection [43].

One could take these previous lower-bounds for $\text{HPC}_{m,r}$, and prove a lower-bound of $\Omega(\frac{m^2}{r})$ for $\text{MHPC}_{m,r}$, but not the lower-bound of $\Omega(m^2)$ which we obtain here. The insufficiency of these previous proofs comes from the notion of “almost solving” that is used. There, a protocol is said to “almost solve” Set-Intersection if the distribution of the intersection point is sufficiently changed by the knowledge gained from the protocol’s execution. More precisely, if the distribution of the intersection point $\mu(A \cap B \mid \Pi)$, conditioned on knowing the transcript Π , is sufficiently far away, in total variation distance (TVD), from the distribution of the intersection point $\mu(A \cap B)$, as it is known before the protocol begins. The quadratic margin in terms of r is ultimately a result of the quadratic loss between TVD and Shannon information, in the use of Pinsker’s inequality.

This same issue was the cause of a decades-long open problem on the complexity of (non-hidden) pointer chasing. Nisan and Wigderson proved in 1991 [56] that any r -round protocol for pointer chasing, where the wrong player starts, needs to communicate $\omega(\frac{m}{r^2})$ bits. But there is a simple upper bound of $O(\frac{m}{r})$. In 2000, Klauck [45] gave a non-constructive proof of a matching lower-bound. That is, he showed that the randomized communication complexity is indeed $\Omega(\frac{m}{r})$, but without providing a hard distribution, which must exist via Yao’s Principle. This problem remained open until 2019, when Yehudayoff [63] showed that the distributional complexity of pointer chasing is $\Omega(\frac{m}{r})$ under the uniform distribution, whenever $r \ll \sqrt{m}$. The proof used a measure of information called *triangular discrimination*, which had never before been used in the lower-bounds literature.

Thus, being simultaneously aware of the three works of [7], [14], and [63], one is naturally led to ask if they can be combined in such a way as to improve the $\frac{m^2}{r^2}$ lower bound for HPC, to $\frac{m^2}{r}$? And could we then prove a lower bound of $\Omega(m^2)$ for Multilayer HPC?

This turns out to be the case. We are not only able to prove Theorem 2, but we also improve the lower bound for (single layer) HPC:

► **Theorem 3.** *Let $r = O(\sqrt{m})$. Then, any randomized protocol with less than r rounds, or even any misaligned randomized protocol with r rounds, cannot solve $\text{BHPC}_{m,r}$ correctly with fewer than $\Omega(\frac{m^2}{r})$ bits of communication.*

The key insight in the new lower bounds is that the notion of “almost solving” an instance of Set-Intersection can be adapted to use triangular discrimination instead of TVD. Two issues then need to be addressed.

First, we must show that a low-round protocol for HPC or MHPC must be “almost solving” (in the new sense) an instance of Set-Intersection in one of the rounds. The proofs follows the general outline of [7], but need to be adapted to use triangular discrimination instead of TVD. To see that it works, one must first understand that triangular discrimination obeys a property analogous to TVD, saying that the expected value of $f(x)$, when x is sampled by some distribution μ , is not too far from the expected value of $f(x)$ when x is sampled by a different distribution ν , if μ and ν are close with respect to triangular discrimination. This is obvious for TVD, but not as obvious for triangular discrimination. It is also not obvious how to adapt the proof to Multilayer HPC, in a way that works for any number of rounds $r \leq m$.

Second, we must show that a low-information protocol that “almost solves” (in the new sense) Set-Intersection can still be used to obtain a low-information protocol that exactly solves Set-Intersection. The proof is similar to [14]. In that paper, a reduction is given which solves a given Set-Intersection instance by sampling $O(1)$ runs of a protocol that “almost solves” Set-Intersection in terms of TVD. We reinterpret their reduction as using the almost-solving protocol to assign scores to elements (predicting how likely they are to be the intersecting element), and come up with a new scoring function which allows a reduction from set intersection to almost-solving with respect to positive triangular discrimination.

2.2 Reduction to Degeneracy

We give a high-level overview of the key idea behind the reduction from BMHPC to the streaming problem of finding the graph degeneracy. First, suppose that we want to show a streaming lower bound for the harder problem of finding a degeneracy ordering. In the classical offline setting, we can obtain such an ordering by the peeling algorithm that recursively removes the min-degree node from the graph and appends it to the end of the ordering. But naively implementing this algorithm in the semi-streaming setting seems difficult since it is inherently sequential. We can store the degree of each node in semi-streaming space and find the min-degree node v in the graph. After we remove v , we need to find a min-degree node v' in the new graph $G \setminus \{v\}$. But at the beginning of the stream, we did not know which node v is, and hence might not have stored enough of its neighbors so as to update their degrees and find v' . Hence, naively, we need to make a new pass for each peeled vertex, which takes $\Theta(n)$ passes in total for an n -node graph. One might wonder whether *any* semi-streaming algorithm for degeneracy ordering would need close to these many passes. If so, how do we prove it?

Consider just the basic problem of finding a min-degree node in an n -node graph, which is the primitive for finding a degeneracy ordering. It can be shown via a simple reduction that a streaming algorithm for this problem can be used to solve SetInt_n , the Set-Intersection communication problem with universe size n . As noted above, finding the degeneracy ordering translates to finding a sequence of nodes that have smallest degree in the remaining graph. This means we can use it to basically solve a sequence of $\text{SetInt}_{\Theta(n)}$ instances. These instances are, however, not independent. The solution to the first instance gives a min-degree node in the original graph, whose removal leads to the second instance; solving this instance reveals the third instance, and so on and so forth. This gives a flavor of a combination of SetInt and pointer chasing, where each pointer is revealed by solving a SetInt instance corresponding to the previous pointer. This is precisely the concept behind HPC (or MHPC for that matter)! Hence, it is plausible that the degeneracy ordering problem can be reduced from MHPC, and we embark on the journey to find such a reduction.

Recall the definition of MHPC from Section 2.1. Given an instance of MHPC, we construct the following layered graph with $r + 1$ layers L_0, \dots, L_r . Each layer has m nodes: the nodes in the even layers correspond to x_i 's and the ones in the odd layers correspond to y_i 's. The edges of the graph are always between two consecutive layers. The players P_A and P_B encode the sets $A_{x_i}^1$ and $B_{x_i}^1$ by adding edges between L_0 and L_1 . Consider the following encoding: for each $i, j \in [m]$, if $y_j \in A_{x_i}^1$, then P_A adds an edge from the i th node in L_0 to the j th node in L_1 . P_B does the analogous construction for the elements in $B_{x_i}^1$ (note that this can lead to parallel edges). P_C and P_D encode the sets $C_{y_i}^2$ and $D_{y_i}^2$ by adding edges between L_1 and L_2 in the analogous way. Again, P_A and P_B encode $A_{y_i}^3$ and $B_{y_i}^3$ with edges between L_2 and L_3 , and this proceeds alternately until the relevant players add the edges between L_r and L_{r+1} .

Let v_0 be the first node in L_0 ; recall that it corresponds to $x_1 = z_0$. Assume that v_0 is the min-degree node in the graph with $\deg(v_0) = d - 1$ and all other nodes have the same degree d . Again, recall that $z_1 = A_{z_0}^1 \cap B_{z_0}^1$. By construction and by the unique-intersection promise of the SetInt instances of MHPC, v_0 has two parallel edges to the node representing z_1 in L_1 ; call this node v_1 . To all other nodes in L_1 , v_0 has at most one edge. Hence, when the peeling algorithm deletes v_0 from the graph, only the degree of v_1 drops by 2, i.e., $\deg(v_1)$ becomes $d - 2$; all other nodes have at most a drop of 1 in degree, i.e., have degree $\geq d - 1$. Thus, v_1 becomes the new min-degree node in the graph. Now, when v_1 is deleted, by similar logic, the node v_2 in L_2 , corresponding to the element $z_2 = C_{z_1}^2 \cap D_{z_1}^2$, becomes a min-degree

node in the remaining graph with $\deg(v_2) = d - 2$. However, now some node in L_0 might also have degree $d - 2$; this is the case when $z_1 = A_{x_i}^1 \cap B_{x_i}^1$ for some $i \neq 1$ as well. Now assume that the peeling algorithm breaks ties by choosing a node in the highest layer among all min-degree nodes (and arbitrarily within the highest layer). Then, indeed it chooses v_2 as the next node to peel (since it is the unique min-degree node in L_2 , again by the `SetInt` promise). Thus, it follows inductively that the i th iteration of the peeling algorithm removes the node corresponding to z_{i-1} in L_{i-1} . Hence, the $(r + 1)$ th node in the degeneracy ordering can be used to identify z_r .

The above high-level idea has quite a few strong assumptions. The challenge is now to get rid of them. We list these challenges and then describe how we overcome them.

- (i) The constructed graph has parallel edges. Then the reduction would only prove a lower bound against algorithms that can handle multigraphs, which is much weaker than a lower bound against algorithms that work on simple graphs.
- (ii) We assume that the tie-breaking is done by the peeling algorithm so as to pick a vertex in the highest layer. It is not at all clear how to get rid of this assumption in a straightforward way.
- (iii) We also assume that we can set the initial degrees in such a way that v_0 has degree $d - 1$ and all other nodes have degree d . It is not clear that we can do this while preserving the relevant properties of the construction.
- (iv) Even if we can overcome the above challenges and the reduction goes through, then we prove a lower bound for finding degeneracy ordering, which is (at least formally) harder than the problem of finding the degeneracy value. Ideally we would like to show the lower bound for the simplest variant of the problem: checking whether degeneracy of the graph is smaller than a given value k or not.

To get around (i), we modify the construction to have a pair of nodes represent each element. The edge construction is done in the following way. Suppose the pair (u_1, u_2) represents an element x_i in layer $\ell - 1$, and (w_1, w_2) represents y_j in layer ℓ . If $y_j \in A_{x_i}^\ell$, then we add edges from u_1 to both w_1 and w_2 . Similarly, if $y_j \in B_{x_i}^\ell$, then we add edges from u_2 to both w_1 and w_2 . Note that if $y_j \in A_{x_i}^\ell \cap B_{x_i}^\ell$, then we have all 4 cross edges between the two pairs, and otherwise we have only 2 edges between them, one on each w_i . Hence, when u_1 and u_2 are removed, both w_1 and w_2 lose degree by 2 if y_j is the intersecting element, and otherwise they only lose degree by 1. This captures the property of the reduction that we want, without constructing parallel edges.

For (ii), we do something more elaborate. On a high level, we duplicate each of the layers L_1, \dots, L_r to provide a “padding” between two initially-consecutive layers. This padding has additional nodes that create an asymmetry between the layer preceding it and the one succeeding it. This asymmetry ensures that the degrees of the nodes in the higher layer drop more than those in the lower layer. Then, we can proceed with the peeling algorithm as planned.

To handle (iii), we show that once we are done with the construction based on the MHPC instance, we can consider each node, look at its degree, and add edges from it to some auxiliary vertices so as to reach its “target degree”. We need to be careful about two things: one, we preserve the properties of the construction so that the reduction goes through, and two, we do not add too many new nodes that might make the bound obtained from the reduction weak. We succeed in achieving a construction without violating the above.

Finally, for (iv), we observe that while we gave the above outline for a reduction from MHPC, the “easier” boolean version BMHPC has a similar lower bound. We then succeed in extending the ideas to reduce the boolean problem of “checking whether degeneracy $\leq k$ ”

from the BMHPC problem, thus obtaining the desired lower bound for this simple variant. For reduction from BMHPC, where the goal is to output just the bit $b(z_r)$ (see definition in Section 2.1) rather than z_r , we need to make non-trivial modifications in the graph: we join the nodes which represent the bit-1 elements in the last layer, to some “special nodes” S . The other nodes in the last layer are not joined to them. The special nodes are also adjacent to all nodes in the other layers. We show that if $b(z_r) = 1$, then after the peeling algorithm removes the nodes corresponding to z_r in the last layer, the degrees of the special nodes drop enough such that all the remaining vertices get peeled one by one, while having degree at most some value k during deletion. This implies that the graph has degeneracy $\leq k$. Otherwise, if $b(z_r) = 0$, we show that after peeling off the nodes representing z_r in the last layer, the minimum vertex-degree in the remaining subgraph is at least $k + 1$, implying that the degeneracy of the graph must be at least $k + 1$. We give the detailed reduction and proof in Section 5 of the full version [9].

2.3 Communication Upper Bounds for Degeneracy

We give a short overview of the $\tilde{O}(n)$ communication protocol for computing the degeneracy of a graph. In the two player communication model, the edges of the input graph G are split into two disjoint sets E_A and E_B given to the players Alice and Bob respectively, and they wish to find the degeneracy of G . Note that the search problem (finding the degeneracy) reduces to its decision counterpart (is the degeneracy $\leq k$?) by a binary search, costing only a $\log n$ multiplicative factor in the communication cost. Hence, we focus on the version where Alice and Bob are additionally given an integer k , and wish to decide if the degeneracy of G is at most k .

To solve this decision problem, we implement the following version of the peeling algorithm in a communication protocol: while there is a vertex of degree $\leq k$, remove it. If the graph is non-empty at the end, reject, otherwise accept. The main challenge in adapting this algorithm is that in the worst case, it seems to update the degree of almost all vertices in G after each deletion, and there is no way to do that without a lot of communication.

However, we observe that if a vertex has degree at least $k + \sqrt{n}$, then it cannot be deleted for the next \sqrt{n} iterations (since each iteration can reduce its degree by at most one). This observation alone gives us the following $\tilde{O}(n\sqrt{n})$ communication protocol:

1. Compute the degree of each vertex in G .
2. Ignore all vertices of degree $\geq k + \sqrt{n}$ while performing \sqrt{n} rounds of the trivial peeling algorithm.
3. Go to Step 1.

Note that the communication in Step 2 comes from Alice and Bob sending each other the low degree ($< k + \sqrt{n}$) neighbors of the vertex deleted in each iteration of the peeling algorithm. We observe that while a vertex has degree $\geq k + \sqrt{n}$, it is not listed in Step 2, and once its degree falls below the threshold of $k + \sqrt{n}$, it is listed at most \sqrt{n} times due to Step 2. Thus, the total communication due to Step 2 over the course of the entire protocol is bounded by $\tilde{O}(n\sqrt{n})$. Also, we recompute the degrees of *all* vertices (which costs $O(n \log n)$ communication each time) at most \sqrt{n} times; these two facts combined give us the desired bound.

To get an improved $\tilde{O}(n)$ communication protocol, we extend the idea above to partition the vertices into $\log n$ sets, where the i -th set contains vertices of degree between $k + 2^{i-1}$ and $k + 2^i$. While the global approach (of simply ignoring the high-degree vertices for \sqrt{n} steps) does not work any more, we are able to make a more local argument as follows: for a

vertex of degree $k + \ell$ to be deleted, it must lose at least ℓ neighbors, which means it must lose $\ell/2$ neighbors in either Alice's or Bob's edge set. But now the players can just track this "private" loss of degree of each vertex, and communicate to update the degree of a vertex in the i -th set only when either private degree falls by at least 2^{i-2} . We are able to show that the degree of each vertex is updated $O(\log n)$ times over the entire course of this new protocol, and hence the total communication is $\tilde{O}(n)$. We further show that this can be extended to finding a k -core of the graph with $\tilde{O}(n)$ communication. Thus, we establish finding degeneracy and k -core as not-too-hard problems.

References

- 1 Sepehr Assadi A. Tight space-approximation tradeoff for the multi-pass streaming set cover problem. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*, pages 321–335, 2017.
- 2 Sepehr Assadi A. A two-pass (conditional) lower bound for semi-streaming maximum matching. 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 708–742. SIAM, 2022.
- 3 Amir Abboud, Keren Censor-Hillel, Seri Khoury, and Ami Paz. Smaller cuts, higher lower bounds. *ACM Trans. Algorithms*, 17(4):30:1–30:40, 2021.
- 4 Noga Alon and Sepehr Assadi. Palette sparsification beyond $(\Delta+1)$ vertex coloring. In Jaroslav 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 6:1–6:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
- 5 Noga Alon, Raphael Yuster, and Uri Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.
- 6 J. Ignacio Alvarez-Hamelin, Luca Dall'Asta, Alain Barrat, and Alessandro Vespignani. Large scale networks fingerprinting and visualization using the k-core decomposition. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, pages 41–50, 2005.
- 7 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Polynomial pass lower bounds for graph streaming algorithms. In *STOC*, pages 265–276. ACM, 2019.
- 8 Sepehr Assadi and Aditi Dudeja. A simple semi-streaming algorithm for global minimum cuts. In *SOSA*, pages 172–180. SIAM, 2021.
- 9 Sepehr Assadi, Prantar Ghosh, Bruno Loff, Parth Mittal, and Sagnik Mukhopadhyay. Polynomial pass semi-streaming lower bounds for k-cores and degeneracy. *arXiv preprint*, 2024. [arXiv:2405.14835](https://arxiv.org/abs/2405.14835).
- 10 Sepehr Assadi, Arun Jambulapati, Yujia Jin, Aaron Sidford, and Kevin Tian. Semi-streaming bipartite matching in fewer passes and optimal space. In *SODA*, pages 627–669. SIAM, 2022.
- 11 Sepehr Assadi, Gillat Kol, Raghuvansh Saxena, and Huacheng Yu. Multi-pass graph streaming lower bounds for cycle counting, max-cut, matching size, and other problems. In *61st Annual IEEE Symposium on Foundations of Computer Science, FOCS (to appear)*, 2020.
- 12 Sepehr Assadi, Gillat Kol, and Zhijun Zhang. Rounds vs communication tradeoffs for maximal independent sets. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 – November 3, 2022*, pages 1193–1204. IEEE, 2022.
- 13 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.

- 14 Sepehr Assadi and Ran Raz. Near-quadratic lower bounds for two-pass graph streaming algorithms. In *FOCS*, pages 342–353. IEEE, 2020.
- 15 Nir Bachrach, Keren Censor-Hillel, Michal Dory, Yuval Efron, Dean Leitersdorf, and Ami Paz. Hardness of distributed optimization. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 – August 2, 2019*, pages 238–247. ACM, 2019.
- 16 Bahman Bahmani, Ravi Kumar, and Sergei Vassilvitskii. Densest subgraph in streaming and mapreduce. *PVLDB*, 5(5):454–465, 2012.
- 17 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, January 6-8, 2002, San Francisco, CA, USA.*, pages 623–632, 2002.
- 18 Suman K. Bera, Amit Chakrabarti, and Prantar Ghosh. Graph coloring via degeneracy in streaming and other space-conscious models. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, pages 11:1–11:21, 2020.
- 19 Sayan Bhattacharya, Monika Henzinger, Danupon Nanongkai, and Charalampos E. Tsourakakis. Space- and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 173–182, 2015.
- 20 Joakim Blikstad, Jan van den Brand, Yuval Efron, Sagnik Mukhopadhyay, and Danupon Nanongkai. Nearly optimal communication and query complexity of bipartite matching. In *FOCS*, pages 1174–1185. IEEE, 2022. doi:10.1109/FOCS54457.2022.00113.
- 21 Francesco Bonchi, Francesco Gullo, Andreas Kaltenbrunner, and Yana Volkovich. Core decomposition of uncertain graphs. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA – August 24–27, 2014*, pages 1316–1325. ACM, 2014.
- 22 Amit Chakrabarti, Prantar Ghosh, Andrew McGregor, and Sofya Vorotnikova. Vertex ordering problems in directed graph streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1786–1802, 2020.
- 23 Yi-Jun Chang, Martin Farach-Colton, Tsan-sheng Hsu, and Meng-Tsung Tsai. Streaming complexity of spanning tree computation. In *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, pages 34:1–34:19, 2020.
- 24 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In Klaus Jansen and Samir Khuller, editors, *Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, volume 1913 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2000.
- 25 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Almost optimal super-constant-pass streaming lower bounds for reachability. 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 570–583. ACM, 2021.
- 26 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Near-optimal two-pass streaming algorithm for sampling random walks over directed graphs. 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 52:1–52:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.



- 27 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Towards multi-pass streaming lower bounds for optimal approximation of max-cut. 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 878–924. SIAM, 2023.
- 28 Deming Chu, Fan Zhang, Xuemin Lin, Wenjie Zhang, Ying Zhang, Yinglong Xia, and Chenyi Zhang. Finding the best k in core decomposition: A time and space optimal solution. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pages 685–696. IEEE, 2020.
- 29 Laxman Dhulipala, Guy E. Blelloch, and Julian Shun. Julienne: A framework for parallel graph algorithms using work-efficient bucketing. In Christian Scheideler and Mohammad Taghi Hajiaghayi, editors, *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2017, Washington DC, USA, July 24-26, 2017*, pages 293–304. ACM, 2017.
- 30 Laxman Dhulipala, Guy E. Blelloch, and Julian Shun. Theoretically efficient parallel graph algorithms can be fast and scalable. In Christian Scheideler and Jeremy T. Fineman, editors, *Proceedings of the 30th on Symposium on Parallelism in Algorithms and Architectures, SPAA 2018, Vienna, Austria, July 16-18, 2018*, pages 393–404. ACM, 2018.
- 31 Shahar Dobzinski, Noam Nisan, and Sigal Oren. Economic efficiency requires interaction. *Games Econ. Behav.*, 118:589–608, 2019.
- 32 Paul Erdős and András Hajnal. On chromatic number of graphs and set-systems. *Acta Math. Acad. Sci. Hungar.*, 17(61-99):1, 1966.
- 33 Hossein Esfandiari, Silvio Lattanzi, and Vahab S. Mirrokni. Parallel and streaming algorithms for k -core decomposition. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1396–1405. PMLR, 2018.
- 34 Martin Farach-Colton and Meng-Tsung Tsai. Computing the degeneracy of large graphs. In Alberto Pardo and Alfredo Viola, editors, *LATIN 2014: Theoretical Informatics – 11th Latin American Symposium, Montevideo, Uruguay, March 31 – April 4, 2014. Proceedings*, volume 8392 of *Lecture Notes in Computer Science*, pages 250–260. Springer, 2014.
- 35 Martin Farach-Colton and Meng-Tsung Tsai. Tight approximations of degeneracy in large graphs. In *LATIN 2016: Theoretical Informatics – 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*, pages 429–440, 2016.
- 36 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2-3):207–216, 2005.
- 37 Silvio Frischknecht, Stephan Holzer, and Roger Wattenhofer. Networks cannot compute their diameter in sublinear time. 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 1150–1162. SIAM, 2012.
- 38 Edoardo Galimberti, Francesco Bonchi, Francesco Gullo, and Tommaso Lanciano. Core decomposition in multilayer networks: Theory, algorithms, and applications. *ACM Trans. Knowl. Discov. Data*, 14(1):11:1–11:40, 2020.
- 39 Mohsen Ghaffari, Silvio Lattanzi, and Slobodan Mitrovic. Improved parallel algorithms for density-based network clustering. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2201–2210. PMLR, 2019.
- 40 Venkatesan Guruswami and Krzysztof Onak. Superlinear lower bounds for multipass graph processing. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 287–298, 2013.

- 41 Magnús M. Halldórsson, Xiaoming Sun, Mario Szegedy, and Chengu Wang. Streaming and communication complexity of clique approximation. In *Automata, Languages, and Programming – 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 449–460, 2012.
- 42 Gábor Ivanyos, Hartmut Klauck, Troy Lee, Miklos Santha, and Ronald de Wolf. New bounds on the classical and quantum communication complexity of some graph properties. In *FSTTCS*, volume 18 of *LIPICs*, pages 148–159. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012.
- 43 T. S. Jayram, Ravi Kumar, and D. Sivakumar. Two applications of information complexity. In *STOC*, pages 673–682. ACM, 2003.
- 44 Wissam Khaouid, Marina Barsky, S. Venkatesh, and Alex Thomo. K-core decomposition of large networks on a single PC. *Proc. VLDB Endow.*, 9(1):13–23, 2015.
- 45 Hartmut Klauck. On quantum and probabilistic communication: Las vegas and one-way protocols. In *32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 644–651, 2000.
- 46 Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, and Huacheng Yu. Characterizing the multi-pass streaming complexity for solving boolean csp exactly. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPICs*, pages 80:1–80:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.
- 47 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1272–1287. SIAM, 2016.
- 48 Chao Li, Li Wang, Shiwen Sun, and Chengyi Xia. Identification of influential spreaders based on classified neighbors in real-world complex networks. *Appl. Math. Comput.*, 320:512–523, 2018.
- 49 Conggai Li, Fan Zhang, Ying Zhang, Lu Qin, Wenjie Zhang, and Xuemin Lin. Efficient progressive minimum k-core search. *Proc. VLDB Endow.*, 13(3):362–375, 2019.
- 50 Quanquan C. Liu, Jessica Shi, Shangdi Yu, Laxman Dhulipala, and Julian Shun. Parallel batch-dynamic algorithms for k-core decomposition and related graph problems. In Kunal Agrawal and I-Ting Angelina Lee, editors, *SPAA '22: 34th ACM Symposium on Parallelism in Algorithms and Architectures, Philadelphia, PA, USA, July 11-14, 2022*, pages 191–204. ACM, 2022.
- 51 Yang P. Liu, Arun Jambulapati, and Aaron Sidford. Parallel reachability in almost linear work and square root depth. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 1664–1686. IEEE Computer Society, 2019.
- 52 David W. Matula and Leland L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *J. ACM*, 30(3):417–427, 1983.
- 53 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Rec.*, 43(1):9–20, 2014.
- 54 Andrew McGregor, David Tench, Sofya Vorotnikova, and Hoa T. Vu. Densest subgraph in dynamic graph streams. In *Mathematical Foundations of Computer Science 2015 – 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II*, pages 472–482, 2015.
- 55 Sagnik Mukhopadhyay and Danupon Nanongkai. Weighted min-cut: sequential, cut-query, and streaming algorithms. In *STOC*, pages 496–509. ACM, 2020. doi:10.1145/3357713.3384334.
- 56 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 419–429, 1991.
- 57 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM J. Comput.*, 22(1):211–219, 1993.

7:16 Polynomial Pass Semi-Streaming Lower Bounds for K-Cores and Degeneracy

- 58 Christos H. Papadimitriou and Michael Sipser. Communication complexity. *J. Comput. Syst. Sci.*, 28(2):260–269, 1984.
- 59 Stephen Ponzio, Jaikumar Radhakrishnan, and Srinivasan Venkatesh. The communication complexity of pointer chasing: Applications of entropy and sampling. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 602–611, 1999.
- 60 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 515–524. ACM, 2013.
- 61 Aviad Rubinfeld, Tselil Schramm, and S. Matthew Weinberg. Computing exact minimum cuts without knowing the graph. In *ITCS*, volume 94 of *LIPICs*, pages 39:1–39:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.ITCS.2018.39.
- 62 Ahmet Erdem Sariyüce, Bugra Gedik, Gabriela Jacques-Silva, Kun-Lung Wu, and Ümit V. Çatalyürek. Streaming algorithms for k-core decomposition. *Proc. VLDB Endow.*, 6(6):433–444, 2013.
- 63 Amir Yehudayoff. Pointer chasing via triangular discrimination. *Comb. Probab. Comput.*, 29(4):485–494, 2020.

Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries

Gil Cohen  

Department of Computer Science, Tel Aviv University, Israel

Tal Yankovitz  

Department of Computer Science, Tel Aviv University, Israel

Abstract

Recently, Kumar and Mon reached a significant milestone by constructing asymptotically good relaxed locally correctable codes (RLCCs) with poly-logarithmic query complexity. Specifically, they constructed n -bit RLCCs with $O(\log^{69} n)$ queries. Their construction relies on a clever reduction to locally testable codes (LTCs), capitalizing on recent breakthrough works in LTCs. As for lower bounds, Gur and Lachish (SICOMP 2021) proved that any asymptotically-good RLCC must make $\tilde{\Omega}(\sqrt{\log n})$ queries. Hence emerges the intriguing question regarding the identity of the least value $\frac{1}{2} \leq e \leq 69$ for which asymptotically-good RLCCs with query complexity $(\log n)^{e+o(1)}$ exist.

In this work, we make substantial progress in narrowing the gap by devising asymptotically-good RLCCs with a query complexity of $(\log n)^{2+o(1)}$. The key insight driving our work lies in recognizing that the strong guarantee of local testability overshoots the requirements for the Kumar-Mon reduction. In particular, we prove that we can replace the LTCs by “vanilla” expander codes which indeed have the necessary property: local testability in the code’s *vicinity*.

2012 ACM Subject Classification Theory of computation → Error-correcting codes

Keywords and phrases Relaxed locally decodable codes, Relaxed locally correctable codes, RLCC, RLDC

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.8

Funding The research leading to these results has received funding from the ERC starting grant 949499 and from the Israel Science Foundation grant 1569/18.

Acknowledgements We are grateful to Marcel Dall’Agnol and Pedro Paredes for identifying an inaccuracy in our original proof, which has been corrected in this revision.

1 Introduction

Error correcting codes with “local guarantees” play a pivotal role in modern coding theory, and their study is highly motivated by applications to theoretical computer science. Of particular interest are locally decodable codes (LDCs), introduced by Katz and Trevisan [34], and locally correctable codes (LCCs) that originated in works on program checking [6, 38]. These are codes that admit highly efficient procedures for recovering a single data symbol. LDCs allow one to decode a specific symbol of the message while querying only a small number of symbols of the received, possibly corrupted, codeword. On the other hand, LCCs offer a method to recover any desired symbol of the codeword using only a few queries.

In their influential work, Ben-Sasson, Goldreich, Harsha, Sudan and Vadhan [2] introduced a natural relaxation of LDCs dubbed *relaxed locally decodable codes* (RLDCs). In essence, RLDCs allow the decoder to abort in the face of corruption, while still being required to always succeed when provided access to a codeword. The natural counterpart to LCCs, known as *relaxed locally correctable codes* (RLCCs), was later introduced by Gur, Ramnarayan, and Rothblum [28]. For linear codes, RLCCs directly induce RLDCs, and so in this case it can be



© Gil Cohen and Tal Yankovitz;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 8; pp. 8:1–8:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



easily seen that RLCCs are stronger objects.¹ LDCs, LCCs and their relaxed counterparts have attracted significant attention. The reader is referred to [31, 45, 16, 18, 26, 36, 39, 32, 25, 28, 17, 27, 10, 11, 14, 5, 19, 21] and references therein. RLDC have found applications to PCPs [40, 42], property testing [7], privacy [23], and probabilistic proof systems [22, 29, 30], to name a few.

For simplicity, in this introductory part we focus on binary codes. Formally, a (q, δ, ε) RLCC is an error correcting code $C \subseteq \{0, 1\}^n$ that is equipped with a randomized “correction procedure”

$$\text{Cor} : \{0, 1\}^n \times [n] \rightarrow \{0, 1\} \cup \{\perp\}$$

that makes at most q queries to its n -bit input, and have the following guarantees:

1. For every codeword $c \in C$, $\text{Cor}(c, i) = c_i$ for every $i \in [n]$, with certainty.
2. For every $w \in \{0, 1\}^n$ of distance at most δn from some codeword $c \in C$, and for every $i \in [n]$, it holds that $\text{Cor}(w, i) \in \{c_i, \perp\}$ with probability at least $1 - \varepsilon$.

We designate δ as the *correction radius* of the RLCC, emphasizing that, as a direct implication, δ serves as a lower bound on the code’s relative-distance. In this paper we consider asymptotically-good RLCCs, by which we mean RLCCs with a constant rate and a constant correction radius. The reader may consult [9], and references therein, to learn more about the constant query regime.

In their work, Gur, Ramnarayan and Rothblum [28] constructed asymptotically-good RLCCs and RLDCs with query complexity $(\log n)^{O(\log \log n)}$. This offers a significant saving over the query complexity of the state-of-the-art LCCs and LDCs, $q = 2^{\tilde{O}(\sqrt{\log n})}$, obtained by Kopparty, Meir, Ron-Zewi, and Saraf [35]. Interestingly, the RLCC of [28] draws inspiration from the ideas presented in the construction of locally testable codes (LTCs) that appears in [35], rather than building on the LCC construction from the same paper. The construction is based on a repeated application of tensoring and distance amplification.

Continuing along a similar framework, but employing a more rate-efficient ingredient instead of tensoring, Cohen and Yankovitz [12] obtained asymptotically-good linear RLCCs, hence also RLDCs, with query complexity $(\log n)^{O(\log \log \log n)}$. This somewhat unnatural looking function, also taking into account the $\Omega(\sqrt{\log n})$ lower bound on the query complexity of asymptotically-good RLCCs [27]² gave some hope that a query complexity of $(\log n)^{O(1)}$ is achievable.

Indeed, this hope was realized in an exciting recent work by Kumar and Mon [37] who obtained RLCCs with query complexity $O(\log^{69} n)$. Their proof builds on a reduction to LTCs, cementing the intuitive connection between RLCCs and LTCs, as hinted in the work of [28], and building on the recent breakthrough in LTCs construction by Dinur, Evra, Livne, Lubotzky, and Mozes [15]³.

1.1 Our result

The works of Kumar and Mon [37] and Gur and Lachish [27] leave us with the fundamental question regarding the identity of the least value $\frac{1}{2} \leq e \leq 69$ for which asymptotically-good RLCCs with $(\log n)^{e+o(1)}$ queries exist. In this work, we make significant progress in narrowing the gap by proving that $e \leq 2$.

¹ For the case of non-linear codes, see [4], Theorem A.6.

² In the case of non-adaptive RLDCs, a slightly stronger lower bound of $\Omega(\sqrt{\log n})$ is known [21]. By combining this result with [19], the strengthened lower bound of $\Omega(\sqrt{\log n})$ can be extended to encompass all linear RLDCs as well.

³ Kumar and Mon require LTCs with rate approaching 1, hence they could not use the independently discovered LTCs by Panteleev and Kalachev [41].

► **Theorem 1 (Main result).** *For every $\delta < 1$ and for infinitely many n -s there exists an explicit binary asymptotically-good linear RLCC (hence also RLDC) of block-length n having correction radius δ , rate $1 - \delta^{1-o(1)} - o_n(1)$, and query complexity*

$$q = (\log n)^{2+o(1)}.$$

Although Kumar and Mon did not explicitly focus on optimizing the exponent in their query complexity, it appears that achieving an exponent as low as 2 is not feasible using existing LTCs within their framework. We believe that the realization that a more “economical” primitive, substituting the LTCs employed by Kumar and Mon, can be employed, plays a pivotal role in achieving such a low query complexity. On the flip side, we believe that new ideas are required to go below $\log^2 n$ queries, if at all possible.

The exact asymptotic behavior of the query complexity q which is hidden, by design, under the $o_n(1)$ -notation is $q = (\log n)^{2+\varepsilon(n)}$, where $\varepsilon(n) = \frac{(\log \log \log n)^3}{\log \log n}$. Similarly, the precise asymptotic behavior underlying the term $\delta^{1-o(1)}$ that appears in the bound on the rate is $\delta \cdot 2^{O((\log \log \frac{1}{\delta})^3)}$. These expressions are derived from the parameters of the lossless expander utilized in our work [8]. While it is possible that slight improvements could be achieved by employing newer constructions of randomness extractors in place of the ingredients used within [8], we have not made any specific attempts to optimize the $o(1)$ terms. At any rate, the reader is referred to Theorem 12 for the formal statement.

We emphasize that even from an information theoretic standpoint, the question of the lowest achievable query complexity for an asymptotically-good RLCC is intriguing. Explicitness aside, we can obtain a slightly reduced query complexity, $q = O(\log^2 n \cdot \log \log n)$. Moreover, in such case the rate comes quite close to the Gilbert-Varshamov bound,

$$\rho = 1 - O\left(\delta \log \frac{1}{\delta}\right) - o(1).$$

In fact, we can construct RLCCs with these parameters in quasi-polynomial time, namely, $2^{(\log n)^{O(1)}}$ by instantiating our construction with another expander construction that appears in [8].

2 Proof Overview

An LTC (Locally Testable Code) is a type of error correcting code that incorporates a local tester—an algorithm that performs a limited number of queries on the received word $w \in \{0, 1\}^n$ and rejects it with a probability proportional to its distance from the code. Importantly, a tester never rejects a valid codeword. LTCs with such a guarantee are occasionally referred to as *strong* LTCs in the literature to differentiate them from an alternative, weaker definition, which only requires the tester to reject words that are sufficiently distant from the code. It is important to recognize that LTCs must in particular handle words that are very far from the code, which constitute the vast majority, “unstructured” portion of $\{0, 1\}^n$. For a more comprehensive exploration of LTCs, we recommend referring to Goldreich’s lecture notes [20].

The key insight driving our work lies in recognizing that the strong guarantee of local testability overshoots the requirements for the Kumar-Mon reduction. Expander codes, although provably not full-fledged LTCs in general, satisfy the required property, namely, all expander codes are locally testable in their vicinity. We make this more precise in Section 2.1 below where we also recall the definition of expander codes. Then, in Section 2.2, we explain how to obtain our RLCCs by instantiating the Kumar-Mon reduction with expander codes instead of with LTCs.

The fact that expander codes are locally testable in the vicinity of the code can be derived as a consequence of the analysis of the sequential decoding algorithm for expander codes. The reader is referred to Section 2.3.1 in Spielman's PhD Thesis [44] and to the discussion in Chapter 5. Interestingly, in his lecture notes, Goldreich [20] discusses offhand a variant of what we call local testability in the vicinity of the code (see Definition 10 in the notes), remarks that this definition may potentially be useful despite being highly non-intuitive in the context of PCPs, and refers to the abovementioned discussion in Spielman's thesis.⁴

For the sake of completeness, we provide a simple proof for the testability of expander codes in their vicinity without relying on a full decoding argument. This streamlined approach helps clarify the concept and establishes the essential property of local testability which is necessary for the reduction.

2.1 Expander codes are locally testable in their vicinity

2.1.1 Expander codes

Let us begin by revisiting the notion of expander codes, introduced by Sipser and Spielman [43]. Let $G = (L, R, E)$ be a bipartite d -left-regular graph. Denote $|L| = n$ and $|R| = \tau n$. The graph G is said to be a $(\gamma, (1 - \varepsilon)d)$ -lossless expander if for every $S \subseteq L$ of size $|S| \leq \gamma n$, the set of neighbors of S , denoted $\Gamma(S)$, is of size at least $(1 - \varepsilon)d|S|$. Additionally, we define $\Gamma_u(S)$ as the set of *unique neighbors* of S which consists of all vertices $v \in R$ such that $|\Gamma(v) \cap S| = 1$. It is easy to prove that

$$|\Gamma(S)| \geq (1 - \varepsilon)d|S| \quad \implies \quad |\Gamma_u(S)| \geq (1 - 2\varepsilon)d|S|.$$

Moving forward, we will assume that ε is a small enough constant such that the right-hand side of the aforementioned equation remains nontrivial. For instance, we can take $\varepsilon = \frac{1}{4}$ as one possible choice. Accordingly, we will refer to the graph G satisfying the condition for this chosen value of ε as a γ -lossless expander for brevity.

By employing the probabilistic method, it is possible to prove the existence of γ -lossless expanders for every desired sizes $|L| = n$, $|R| = \tau n$, where the left-degree $d = O(\log \frac{1}{\tau})$, and $\gamma = O(\frac{\tau}{d})$. For the sake of simplicity and convenience, we shall use such an expander throughout this informal section. In Section 2.2.4, we will briefly discuss the modifications in parameters if we choose to work with the explicit expander from the work of [8].

With the expander G , we associate a binary code $\text{EC}(G)$ on block-length n , dubbed the *expander code* associated with G as follows. Every vertex $v \in R$ is thought of as a constraint, namely, for $x \in \{0, 1\}^n$ to be a codeword, we require that for every $v \in R$, the parity of the bits $\{x_u \mid u \in \Gamma(v)\}$ equals 0 (where we identify the set L with the index set $[n]$). It readily follows that $\text{EC}(G)$ has rate at least $1 - \tau$, and it is not hard to show that the code has relative-distance at least γ .⁵

2.1.2 Expander codes are locally testable in their vicinity

We turn to show that $\text{EC}(G)$ is locally testable in its vicinity. Let $w \in \{0, 1\}^n$ be word of distance exactly $\gamma'n$ from $\text{EC}(G)$, let $c \in \text{EC}(G)$ be a word closest to w , and let $S \subseteq L$ be the set that corresponds to w and c , $S = \{i \mid w_i \neq c_i\}$. We assume that $\gamma' \leq \gamma$, reflecting the fact that we are in the vicinity of the code. Our tester will simply sample a right vertex v at random and rejects if the constraint associated with v is unsatisfied.

⁴ A notion similar, though not identical, to codes that are locally testable at their vicinity appears in [3] and is dubbed *semi-LTC*. We also remark that the given proof for Proposition 6.2 of [3] proves that expander codes are locally testable at their vicinity.

⁵ In fact, stronger bounds on the relative-distance are known though they will not be necessary for our purposes.

Note that the tester will reject whenever v is sampled from $\Gamma_u(S)$. Thus, the probability of rejection is bounded below by

$$\frac{|\Gamma_u(S)|}{|R|} \geq \frac{d|S|}{2\tau n} = \frac{d\gamma'}{2\tau}.$$

Plugging the parameters of the non-explicit expander above, we get that the rejection probability is bounded below by $\Omega(\frac{\gamma'}{\gamma})$. In particular, if w is at the “outskirts” of the expander code, namely, $\gamma' \leq \gamma$ yet $\gamma' = \Omega(\gamma)$, then the rejection probability is constant. Of course, the rejection probability can be amplified to $1 - 2^{-t}$ by repeating the process for $O(t)$ times.

As for the query complexity, for simplicity assume that G is also c -right regular. Then, the query complexity required for obtaining a constant rejection probability is

$$c = \frac{d}{\tau} = O\left(\frac{1}{\tau} \log \frac{1}{\tau}\right).$$

2.2 RLCCs from expander codes

2.2.1 The construction

The key distinction between RLCCs and LTCs, whether they are full-fledged LTCs or only guaranteed to work in their vicinity, lies in the fact that RLCCs are also provided with an index $i \in [n]$ indicating the specific bit to be corrected. To bridge this gap, following Kumar and Mon [37], we define our RLCC using a binary tree⁶ of expander codes so as to make sure that any index i participates in expander codes of increasing size. This allows one to “zoom in” on the i -th bit using expander codes. We elaborate on this next.

Assume for simplicity that $n = 2^m$. We take a sequence of m expander codes C_0, C_1, \dots, C_{m-1} on block-lengths $n, \frac{n}{2}, \frac{n}{4}, \dots$, respectively⁷. All these expander codes share the same parameters as in Section 2.1, namely, all expanders have the same left and right degrees d, c , hence the same τ , as well as the same parameter γ .

Our RLCC, denoted C' , is obtained by intersecting the code C_0 on the index set $[n]$ with the code C_1 on both the index set $[\frac{n}{2}]$ and $\frac{n}{2} + [\frac{n}{2}]$. Put differently, we impose the linear constraints of C_1 on both the first half and second half of the bits. The linear constraints of the code C_2 are enforced onto the four blocks $[\frac{n}{4}]$, $\frac{n}{4} + [\frac{n}{4}]$, $\frac{n}{2} + [\frac{n}{4}]$, and $\frac{3n}{4} + [\frac{n}{4}]$, and so forth in a binary tree fashion. It is evident that the rate of the resulting code, C' , is at least $1 - m\tau$, which implies that we need to select $\tau < \frac{1}{m} = \frac{1}{\log n}$ to satisfy the rate constraint.

2.2.2 The tester and its analysis

Our claim is that C' is an RLCC with correction radius $\frac{\gamma}{2} = \Omega(\frac{1}{\log n \cdot \log \log n})$ using the corrector we describe and analyze next. In Section 2.2.3 we explain how to modify the construction slightly so as to obtain any desired correction radius. Before we begin, we remark that it is readily seen that the corrector described below never aborts and always outputs the correct bit given oracle access to a codeword of C' . Therefore, we focus on the scenario where a word $w \in \{0, 1\}^n \setminus C'$ is given, with a distance at most $\frac{\gamma}{2} \cdot n$ from the code C' . In this case, our objective is to either abort or output the i -th bit of the unique codeword closest to w . Indeed, as $C' \subseteq C_0$, and since C_0 has relative-distance at least γ , there exists a unique codeword $c \in C'$ that is of distance at most $\frac{\gamma}{2} \cdot n$ from w .

⁶ Kumar and Mon work with larger arity. Moreover, their tree does not necessarily induce a sequence of partitions that are exactly cascaded as in our construction, but this is a mere technicality.

⁷ Technically, we will need to stop before reaching 1-bit block-length though this is a mere technicality which we ignore for the sake of simplicity in this informal discussion.

8:6 Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries

With this in mind, let us consider a specific index $i \in [n]$, and let B be either $\lfloor \frac{n}{2} \rfloor$ or $\lfloor \frac{n}{2} \rfloor + 1$, depending on which of these blocks contains i . We define ε such that $\varepsilon \cdot \frac{n}{2}$ is the distance between w_B and c_B - the projections of w, c onto block B , respectively. We know that $\varepsilon \leq \gamma$ as in the worst case all $\frac{\gamma}{2} \cdot n = \gamma \cdot |B|$ errors could fall into B . We consider the two possible cases based on whether the ratio of errors deteriorates or not when moving to block B , i.e., whether $\varepsilon \leq \frac{\gamma}{2}$ or not.

Assume that $\varepsilon > \frac{\gamma}{2}$. As we also know that $\varepsilon \leq \gamma$, namely, w_B is in the vicinity of the code C_1 , we may invoke C_1 -s tester, and by making

$$O\left(t \cdot \frac{1}{\tau} \log \frac{1}{\tau}\right) = O(t \cdot \log n \cdot \log \log n)$$

queries to w_B , reject with probability $1 - 2^{-t}$. Hence, if the tester ended up not aborting, we may assume that we are in the case $\varepsilon \leq \frac{\gamma}{2}$, and our assumption will be wrong with probability at most 2^{-t} . Thus, unless the tester aborted, we can safely recurse to B . In more detail, since w_B is of distance at most $\frac{\gamma}{2} \cdot |B|$ from c_B , and since C_1 is a code with relative-distance γ on the index set B that participates in the intersection defining C' , we know that c_B is the unique codeword of C_1 that is $\frac{\gamma}{2} \cdot |B|$ -close to w_B . This is precisely the same guarantee we started with and, importantly, we maintain the invariant that the projection of c to the block is the closest codeword to w 's projection to that block, with respect to the suitable code. Hence, if and when the time comes to return the i -th bit, it will be that bit of c that is returned rather than the bit of another codeword. This invariant allows us to recurse to B .

If in any of the $m = \log n$ levels of recursion the tester aborts, the corrector succeeds. Otherwise, the code C_m is invoked and returns the correct bit except in case where the corrector should have aborted. By a union bound over the m levels, this event occurs with probability at most $2^{-t}m$. Setting $t = O(\log m) = O(\log \log n)$, the total number of queries made is

$$O(mt \cdot \log n \cdot \log \log n) = O\left((\log n \cdot \log \log n)^2\right).$$

We remark that the factor of $t = O(\log \log n)$ can be removed as the union bound can be avoided with some care.

2.2.3 Improving the correction radius

To achieve any desired correction radius $\delta_0 < 1$, we can easily modify the construction. Simply take the expander code C_0 to have relative-distance $\gamma_0 = 2\delta_0$ and rate

$$1 - \tau_0 = 1 - O\left(\delta_0 \log \frac{1}{\delta_0}\right),$$

while keeping the parameters of the remaining codes C_1, \dots, C_m unchanged. The rate of the resulting code, denoted C'' , is given by $1 - (\tau_0 + (m-1)\tau)$, point being that we can afford taking C_0 to be a high-rate code as we only “pay” τ_0 once rather than m times.

In the modified construction, the corrector remains unchanged with the exception of an initial phase. In this initial phase, we invoke C_0 -s tester (which, as the perceptive reader may have noted, has not been used in Section 2.2.2) to check whether the number of errors is less than $\frac{\gamma}{2} \cdot |B|$. The probability to catch an unsatisfied constraint is no longer constant as before; instead, it becomes

$$\Omega\left(\frac{\gamma}{\gamma_0}\right) = \Omega\left(\frac{1}{\log n \cdot \log \log n}\right).$$

To ensure a constant rejection probability, we need to sample not just one but $\Theta\left(\frac{\gamma_0}{\gamma}\right)$ right vertices and query their neighbors. If we denote the right-degree of the expander underlying C_0 by c_0 , this will result in a total number of

$$O\left(\frac{\gamma_0}{\gamma} \cdot c_0\right) = O\left(\frac{1}{\gamma}\right) = O(\log n \cdot \log \log n)$$

queries. Note that we have used the fact that in the probabilistic construction, $c_0\gamma_0 = O(1)$.

If C_0 -s corrector does not reject, we maintain the same guarantee we had before regarding the number of errors in B , and we can proceed with the same strategy as previously described. Hence, with the same query complexity of $O(\log^2 n \cdot \log \log n)$, it is possible to obtain any distance δ_0 and rate $1 - O(\delta_0 \log \frac{1}{\delta_0}) - o(1)$.

A remark regarding the bi-regularity assumption

We wish to draw attention to an issue that might be easily overlooked regarding the initial phase discussed above in the absence of bi-regularity. Throughout this informal proof overview, we are working under the premise that the expander that is underlying the expander code is bi-regular. This can be assumed to be the case for the probabilistic construction though not necessarily for the expander that we are using for our RLCC construction [8].

In the absence of bi-regularity, one can proceed by defining the tester as follows: When sampling a right vertex, query its neighbors only if its degree is at most κc , where κ serves as a cutoff parameter and c now stands for the average right degree. That is, if the degree exceeds this threshold, the vertex is ignored for the purpose of testing. As a result, the “heavy” constraints are embedded in the code’s definition, yet they are not utilized by the tester. This seemingly minor technicality has a rather surprising impact on the parameters: the query complexity of the tester in the initial phase alone now becomes $(\log n)^{2+o(1)}$. However, this increase is affordable, given that it applies only to the initial phase. As we progress through the remaining $\log n$ levels, the query complexity for each level remains at $(\log n)^{1+o(1)}$.

2.2.4 Explicitness

Capalbo, Reingold, Vadhan and Wigderson [8] constructed explicit γ -lossless expanders with near-optimal parameters.⁸ Quantitatively, following the notation in Section 2.1.1, their construction has degree

$$d = 2^{O((\log \log \frac{1}{\tau})^3)} = \left(\frac{1}{\tau}\right)^{o(1)},$$

which should be compared with $d = O(\log \frac{1}{\tau})$ obtained using the probabilistic construction, while maintaining $\gamma = O(\frac{\tau}{d})$. As before, the probability of the expander code’s tester to reject a word from the outskirts of the code is constant. Hence, the query complexity is, again, the right degree, whose average is

$$c = \frac{d}{\tau} = \frac{1}{\tau} \cdot 2^{O((\log \log \frac{1}{\tau})^3)} = \left(\frac{1}{\tau}\right)^{1+o(1)}.$$

⁸ A lot of work has been done, much of it very recently, on simplifying the [8] construction and on obtaining different variants of lossless expanders such as unique neighbor expanders, however, none of these works seem to be sufficient for our needs. The reader may consult [1, 13, 24, 33] and references therein.

Recall that, due to rate considerations, τ is taken to be $\frac{1}{\log n}$, thus the query complexity of the expander code's tester is $(\log n)^{1+o(1)}$. The overall query complexity of the resulted RLCC's corrector is then $m \cdot (\log n)^{1+o(1)} = (\log n)^{2+o(1)}$, where the handling of the non-bi-regularity is as described in the previous paragraph (see the proof for Theorem 12 at the technical part).

3 Preliminaries

3.1 Notations and conventions

Unless stated otherwise, all logarithms in this paper are taken to the base 2. The set of natural numbers is $\mathbb{N} = \{0, 1, 2, \dots\}$. For $n \in \mathbb{N}$, $n \geq 1$, we use $[n]$ to denote the set $\{1, \dots, n\}$. For $q \in \mathbb{N}$, $q \geq 2$, we use H_q to denote the q -ary entropy function, and $H = H_2$ to denote the binary entropy function.

For a finite set N , we refer to a function $v \in \mathbb{F}^N$ as a *vector* and we say that it is *indexed by N* . For a vector $v \in \mathbb{F}^N$ and $i \in N$ we use v_i as a shorthand for $v(i)$. For a vector $v \in \mathbb{F}^N$ and a set $N' \subseteq N$ we denote by $v_{N'}$ the vector $v' \in \mathbb{F}^{N'}$ such that $v'_i = v_i$ for every $i \in N'$. For two vectors $u, v \in \mathbb{F}^N$, their (absolute) *hamming distance* is $|\{i \in N \mid u_i \neq v_i\}|$, which we denote by $\text{Dist}(u, v)$, and their *relative hamming distance* is $\frac{\text{Dist}(u, v)}{|N|}$, which we denote by $\text{RelDist}(u, v)$.

3.2 Error correcting codes

We start by recalling the definition of an error correcting code. In this work we only consider linear codes. The definition below is standard, however, for our purposes we find it convenient to work with an arbitrary index set rather than the usual set $[n]$, and so the reader may benefit from glancing over the definition.

► **Definition 2.** For a finite set N of size $|N| = n$ and a field \mathbb{F} , a *code* is a linear subspace $C \subseteq \mathbb{F}^N$. We say that the code C is indexed by N and that it is over \mathbb{F} . The length of the code is n . The dimension of the code, usually denoted by k , is the dimension of C over \mathbb{F} . The (non-local) distance of the code, denoted by d , is $\min_{c, c' \in C, c \neq c'} \text{Dist}(c, c')$. The rate of the code, typically denoted by ρ , is $\frac{k}{n}$. The (non-local) relative-distance of the code is defined to be $\frac{d}{n}$. The elements of C are called *codewords*.

3.3 Relaxed locally correctable codes

We turn to recall the definition of relaxed locally correctable codes as put forth by Gur, Ramnarayan and Rothblum [28].

► **Definition 3.** A code $C \subseteq \mathbb{F}^N$ is called a (q, δ, ε) -RLCC (*relaxed locally correctable code, abbreviated*) if there exists a randomized procedure $\text{Cor} : \mathbb{F}^N \times N \rightarrow \mathbb{F} \cup \{\perp\}$ with the following guarantees:

- For every $i \in N$, $c \in C$ and $w \in \mathbb{F}^N$, satisfying $\text{RelDist}(w, c) \leq \delta$, $\text{Cor}(w, i) \in \{c_i, \perp\}$ with probability at least $1 - \varepsilon$.
- $\text{Cor}(c, i) = c_i$ with probability one on any $c \in C$ and $i \in N$.
- $\text{Cor}(w, i)$ always makes at most q queries to w .

We refer to Cor as the *local corrector* (or the *corrector*). The parameter δ is called the *correction radius*, and the parameter q is called the *query complexity*.

The error parameter of an RLCC can be easily amplified at low cost to the query complexity, as stated in the following claim (for a simple proof see, e.g., [12]).

▷ **Claim 4.** Let $C \subseteq \mathbb{F}^N$ be a (q, δ, ε) -RLCC. Then, for any $h \in \mathbb{N}$, C is also an $(hq, \delta, \varepsilon^h)$ -RLCC.

3.4 Expanders and expander codes

We set some standard notation. Let $G = (V, E)$ be an undirected graph. For $v \in V$ we define $\Gamma(v)$ as the set of neighbors of v in G , and let $\deg(v)$ be the degree of v . For a set of vertices $S \subseteq V$, we let $\Gamma(S) = \cup_{v \in S} \Gamma(v)$, and define

$$\Gamma_u(S) = \{v \in V \mid v \text{ is adjacent to exactly one } u \in S\}.$$

► **Definition 5** (Unique-neighbor expanders). *A left- d -regular bipartite graph $G = (L, R, E)$ is a (γ, α) -unique-neighbor expander if for every $S \subseteq U$ such that $|S| \leq \gamma|L|$, it holds that $|\Gamma_u(S)| \geq \alpha d|S|$.*

The following theorem readily follows by the construction of lossless conductors as given by Theorem 7.3 in [8].

► **Theorem 6** ([8]). *There exist universal constants $c_0 \geq 1$ and $\beta \leq 1$ such that the following holds. For every n and $m \leq n$, there exists an explicit (γ, α) -unique-neighbor expander $G = (L, R, E)$ with $|L| = 2^n$, $|R| = 2^m$, having left degree*

$$d \leq 2^{c_0 \cdot \log^3(n-m)},$$

where $\alpha = \Omega(1)$, and $\gamma = \beta \cdot \frac{2^{m-n}}{d}$.

► **Definition 7** (Expander codes). *Let $G = (L, R, E)$ be a bipartite graph and let \mathbb{F} be a field. The expander code associated with G is defined by*

$$\text{EC}_{\mathbb{F}}(G) = \left\{ w \in \mathbb{F}^L \mid \forall v \in R \quad \sum_{u \in \Gamma(v)} w_u = 0 \right\}.$$

We usually omit the subscript \mathbb{F} when the field is clear from context.

It is easy to see that the rate of $\text{EC}_{\mathbb{F}}(G)$ is at least $1 - \frac{|R|}{|L|}$.

4 Vicinity Locally Testable Codes

In this section we give the formal definition of local testability in the vicinity of the code and prove that expander codes have this property.

► **Definition 8** (VLTCs). *A code $C \subseteq \mathbb{F}^N$ is called a $(q, \delta, \kappa, \sigma)$ -VLTC (vicinity locally testable code, abbreviated) if there exists a randomized procedure*

$$\text{Tes} : \mathbb{F}^N \rightarrow \{\circ, \perp\}$$

with the following guarantees:

■ For every $c \in C$ and $w \in \mathbb{F}^N$, satisfying $\text{RelDist}(w, c) \leq \delta$,

$$\Pr[\text{Tes}(w) = \perp] \geq \kappa \cdot \text{RelDist}(w, c) - \sigma;$$

8:10 Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries

- $\text{Tes}(c) = \circ$ with probability one on any $c \in C$.
- $\text{Tes}(w)$ always makes at most q queries to w .

We call Tes a local tester (or tester for short). The parameter q is referred to as the query complexity.

We move to show that expander codes constructed from unique-neighbor expanders are VLTCs.

► **Lemma 9.** *Let $G = (L, R, E)$ be a d -left-regular (γ, α) -unique-neighbor expander with average right-degree \bar{c} . Then, for every $b > 1$, $\text{EC}(G)$ is a $(b\bar{c}, \gamma, \alpha\bar{c}, \frac{1}{b})$ -VLTC.*

Proof. Define

$$R' = \{v \in R \mid \deg(v) \leq b\bar{c}\}.$$

By an averaging argument, $|R'| \geq (1 - \frac{1}{b})|R|$. The tester for $\text{EC}(G)$, given oracle access to $w \in \mathbb{F}^L$, proceeds as follows:

1. Sample $v \in R'$ uniformly at random.
2. Query w on $\Gamma(v)$.
3. Output \circ if $\sum_{u \in \Gamma(v)} w_u = 0$; and \perp otherwise.

As the sampled vertex v is in R' , the query complexity of the tester is indeed bounded above by $b\bar{c}$. Further, when $w \in \text{EC}(G)$, the tester outputs \circ with certainty.

Consider then a word $w \in \mathbb{F}^L$ such that $\text{RelDist}(w, c) \leq \gamma$ for some codeword $c \in \text{EC}(G)$. Let

$$S = \{v \in L \mid w_v \neq c_v\}.$$

As $|S| \leq \gamma|L|$ we have that $|\Gamma_u(S)| \geq \alpha d|S|$. Notice that if the vertex v that is sampled in Step 1 lies in $\Gamma_u(S)$ then the tester outputs \perp . Therefore, the probability of the tester to output \perp is at least

$$\begin{aligned} \frac{|\Gamma_u(S)| - |R \setminus R'|}{|R|} &\geq \frac{\alpha d|S|}{|R|} - \frac{1}{b} \\ &= \alpha \cdot \frac{d|L|}{|R|} \cdot \frac{|S|}{|L|} - \frac{1}{b} \\ &= \alpha\bar{c} \cdot \text{RelDist}(w, c) - \frac{1}{b}, \end{aligned}$$

which concludes the proof. ◀

We will use the following easy claim.

► **Claim 10.** Let $C \subseteq \mathbb{F}^N$ be a $(q, \delta, \kappa, \sigma)$ -VLTC with a tester Tes , and further let $c \in C$ and $w \in \mathbb{F}^N$ be such that $\alpha \leq \text{RelDist}(w, c) \leq \delta$. Assume that we run $\text{Tes}(w)$ for g times, independently. Then, the probability that one of the simulations outputted \perp is at least $1 - e^{-\beta g}$, where $\beta = \kappa\alpha - \sigma$.

Proof. The probability that a single simulation of $\text{Tes}(w)$ outputs \perp is at least

$$\kappa \cdot \text{RelDist}(w, c) - \sigma \geq \kappa\alpha - \sigma = \beta.$$

The probability that all the simulations output \circ is thus $(1 - \beta)^g \leq e^{-\beta g}$. ◀

5 RLCCs from VLTCs

Following a similar argument to the one underlying the Kumar-Mon reduction, the following proposition states that a sequence of VLTCs can be used to construct an RLCC.

► **Proposition 11.** *Let $C_1 \subseteq \mathbb{F}^{N_1}, \dots, C_m \subseteq \mathbb{F}^{N_m}$ be codes with rates ρ_1, \dots, ρ_m , respectively, such that for every $i \in [m-1]$, C_i is a $(q', \delta', \kappa', \sigma')$ -VLTC, and C_m is a $(q, \delta, \kappa, \sigma)$ -VLTC. Further assume that $|N_1| \leq \frac{1}{\delta'}$, $|N_m| = n$, and for every $1 < i \leq m$, $|N_i| = 2|N_{i-1}|$. Then, for every $g \in \mathbb{N}$, there exists an $((m-1)q' + gq + 1, \delta, \varepsilon)$ -RLCC $C \subseteq \mathbb{F}^{[n]}$ with rate*

$$\rho \geq 1 - \sum_{i=1}^m (1 - \rho_i),$$

where

$$\varepsilon \leq 1 - \min \left\{ \frac{\kappa' \delta'}{2} - \sigma', e^{g(\sigma - \frac{\kappa \delta'}{2})} \right\}.$$

Moreover, if the codes C_1, \dots, C_m are explicit, then so is the resulting code C .

Proof. We start by describing how the code C is constructed.

The code construction. Let P_1, \dots, P_m be an arbitrary fixed sequence of partitions of $[n]$, satisfying that for every $i \in [m]$, P_i has 2^{m-i} equal-size parts denoted $\{B_1^i, \dots, B_{2^{m-i}}^i\}$, and that for every $1 < i \leq m$, P_{i-1} is a sub-partition of P_i (that is, for every $B \in P_{i-1}$, there exists $B' \in P_i$ such that $B \subseteq B'$). For every $i \in [m]$ and $B \in P_i$ let $f_{i,B} : B \rightarrow N_i$ be an arbitrary bijection, and define

$$C_{i,B} = \{c \circ f_{i,B} \mid c \in C_i\}.$$

Finally, define the code

$$C = \left\{ w \in \mathbb{F}^{[n]} \mid \forall i \in [m], B \in P_i : w_B \in C_{i,B} \right\}.$$

The moreover part of the proof readily follows. The efficiency of the corrector will be self evident as well once the corrector is presented.

Rate analysis. For every $i \in [m]$ and $B \in P_i$, the number of linear constraints required to impose so that $w_B \in C_{i,B}$ is at most $(1 - \rho_i)|N_i|$. Therefore, the total number of constraints in the definition of the code C is bounded above by

$$\sum_{i=1}^m |P_i|(1 - \rho_i)|N_i| = \sum_{i=1}^m n(1 - \rho_i),$$

which establishes the lower bound on the rate of C .

The corrector. We turn to describe a corrector $\text{Cor} : \mathbb{F}^{[n]} \times [n] \rightarrow \mathbb{F} \cup \{\perp\}$ for C . As for every $i \in [m]$, C_i is a VLTC, it is immediate that so is $C_{i,B}$ for every $B \in P_i$, with the same parameters as C_i . The local tester for $C_{i,B}$ that is induced in the natural way from the local tester for C_i is denoted

$$\text{Tes}_{i,B} : \mathbb{F}^B \rightarrow \{\circ, \perp\}.$$

Let $w \in \mathbb{F}^{[n]}$ and $j \in [n]$. Let $r_1 = r_1(j), \dots, r_m = r_m(j)$ be the indices of blocks within the corresponding partitions P_1, \dots, P_m such that $j \in B_{r_1}^1 \subseteq \dots \subseteq B_{r_m}^m$. The corrector $\text{Cor}(w, j)$ proceeds as follows:

8:12 Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries

1. For $i = 1, \dots, m-1$, simulate $\text{Tes}_{i, B_{r_i}^i}(w_{B_{r_i}^i})$.
2. Simulate $\text{Tes}_{m, B_{r_m}^m}(w_{B_{r_m}^m})$ for g times.
3. If any of the simulations outputted \perp , output \perp ; otherwise, output w_j .

Query analysis. As Cor simulates $m-1$ testers with query complexity q' , and invokes g simulations of one tester with query complexity q , the overall query complexity is $(m-1)q' + gq + 1$, accounting also for querying w_j .

Correctness. Clearly, if w is a codeword of C then $w_{B_{r_i}^i} \in C_{i, B_{r_i}^i}$ for every $i \in [m]$, and so $\text{Tes}_{i, B_{r_i}^i}(w_{B_{r_i}^i}) = \circ$ with certainty. Therefore, $\text{Cor}(w, j) = w_j$ with certainty, as required. Assume that $w \in \mathbb{F}^{[n]}$ is such that $\text{Dist}(w, c) \leq \delta n$ for $c \in C$. Since $\text{Cor}(w, j)$ always either outputs \perp or w_j , it suffices to show that if $w_j \neq c_j$ then the corrector outputs \perp with probability at least $1 - \varepsilon$. Towards this end, assume $w_j \neq c_j$, and hence $w_{B_{r_1}^1} \neq c_{B_{r_1}^1}$, and further note that $w_{B_{r_m}^m} = w$ and $c_{B_{r_m}^m} = c$. For every $i \in [m-1]$ define $\delta_i = \delta'$, and let $\delta_m = \delta$. Since, per our assumption, $|N_1| \leq \frac{1}{\delta'}$, we have that

$$\text{RelDist}(w_{B_{r_1}^1}, c_{B_{r_1}^1}) \geq \delta' = \delta_1,$$

whereas

$$\text{RelDist}(w_{B_{r_m}^m}, c_{B_{r_m}^m}) \leq \delta = \delta_m.$$

Let $\iota \in \{2, 3, \dots, m\}$ be any index satisfying that

$$\begin{aligned} \text{RelDist}(w_{B_{r_{\iota-1}}^{\iota-1}}, c_{B_{r_{\iota-1}}^{\iota-1}}) &\geq \delta_{\iota-1} \\ \text{RelDist}(w_{B_{r_\iota}^\iota}, c_{B_{r_\iota}^\iota}) &\leq \delta_\iota. \end{aligned}$$

By the above account, ι is well-defined. Since $B_{r_{\iota-1}}^{\iota-1} \subseteq B_{r_\iota}^\iota$ and $|B_{r_{\iota-1}}^{\iota-1}| = \frac{1}{2}|B_{r_\iota}^\iota|$, we have that

$$\frac{\delta_{\iota-1}}{2} \leq \text{RelDist}(w_{B_{r_\iota}^\iota}, c_{B_{r_\iota}^\iota}) \leq \delta_\iota.$$

If $\iota < m$, as $\text{Tes}_{\iota, B_{r_\iota}^\iota}$ is a local tester for the $(q', \delta_\iota, \kappa', \sigma')$ -VLTC $C_{i, B_{r_\iota}^\iota}$ and since $c_{B_{r_\iota}^\iota} \in C_{\iota, B_{r_\iota}^\iota}$, it holds that $\text{Tes}_{\iota, B_{r_\iota}^\iota}(w_{B_{r_\iota}^\iota})$ outputs \perp with probability at least

$$\frac{\kappa' \delta_{\iota-1}}{2} - \sigma' = \frac{\kappa' \delta'}{2} - \sigma'.$$

If otherwise $\iota = m$ then we set

$$\beta = \frac{\kappa \delta_{m-1}}{2} - \sigma = \frac{\kappa \delta'}{2} - \sigma$$

and then by Claim 10 one of the g simulations of $\text{Tes}_{m, B_{r_m}^m}((w_{B_{r_m}^m}))$ with probability at least $1 - e^{-\beta g}$. Thus, $\text{Cor}(w, j)$ outputs \perp with probability at least $\min\{\frac{\kappa' \delta'}{2} - \sigma', 1 - e^{-\beta g}\}$, as required. \blacktriangleleft

We are now ready to prove our main theorem.

► **Theorem 12.** For every finite field \mathbb{F} , n which is a power of 2, and $\delta > 0$, there exists an explicit $(q, \delta, \frac{1}{3})$ -RLCC $C \subseteq \mathbb{F}^{[n]}$ with query complexity

$$q = (\log n)^{2+o(1)},$$

and rate

$$\rho = 1 - \delta \cdot 2^{O\left(\left(\log \log \frac{1}{\delta}\right)^3\right)} - o(1).$$

Proof. Write $n = 2^r$, and let

$$\ell = r \log r = \log n \cdot \log \log n.$$

Due to the claimed tradeoff between the rate and the correction radius, we may as well assume that $\delta \geq \frac{1}{2^{\lceil \log \ell \rceil}}$. We proceed to describe the sequence of expander codes that we will use, which consists of $s = r - \lceil \log \ell \rceil + 1$ codes. For every $i \in [s]$, the block-length of the i -th code is

$$n_i = 2^{\lceil \log \ell \rceil + i - 1}.$$

Note that, in particular, $n_s = n$. Further, the number of linear constraints defining the i -th code is $m_i = 2^{i-1}$ for $i \in [s-1]$, whereas the for the s -th code,

$$m_s = 2^{r - \lfloor \log \frac{1}{\delta} + \log \beta - c_0 \log^3(\log \frac{1}{\delta}) \rfloor},$$

where β is the constant from Theorem 6.

Invoking Theorem 6, for every $i \in [s]$ let $G_i = (L_i, R_i, E_i)$ be a d_i -left-regular bipartite graph with $|L_i| = n_i$ and $|R_i| = m_i$ which is a (δ_i, α) -unique-neighbor expander for $\alpha = \Omega(1)$, such that for $i \in [s-1]$,

$$d_i \leq 2^{c_0 \log^3(\lceil \log \ell \rceil)} \triangleq d,$$

$$\delta_i \geq \frac{\beta}{d \cdot 2^{\lceil \log \ell \rceil}} \triangleq \delta',$$

and

$$d_s \leq 2^{c_0 \log^3(\log \frac{1}{\delta})},$$

$$\delta_s \geq \frac{\beta}{d_s \cdot 2^{\lfloor \log(1/\delta) + \log(\beta) - c_0 \log^3(\log(1/\delta)) \rfloor}} \geq \delta.$$

The sequence of codes is defined by setting, for every $i \in [s]$, $C_i = \text{EC}(G_i)$.

We turn to address the VLTC-ness of C_1, \dots, C_s . Set $b = \frac{4}{\alpha\beta}$ and $b_s = \frac{4m_s}{\alpha\delta'd_s n_s}$. By Lemma 9, for every $i \in [s-1]$, C_i is a

$$\left(\frac{bd_i n_i}{m_i} \leq bd 2^{\lceil \log \ell \rceil}, \delta', \alpha d 2^{\lceil \log \ell \rceil}, \frac{1}{b} \right)\text{-VLTC},$$

and C_s is a

$$\left(\frac{b_s d_s n_s}{m_s}, \delta, \alpha \frac{d_s n_s}{m_s}, \frac{1}{b_s} \right)\text{-VLTC}.$$

We further set $g = \frac{4m_s}{\alpha\delta'd_s n_s}$. We can now invoke Proposition 11 (indeed, the proposition's prerequisites are met, i.e., $n_1 = 2^{\lceil \log \ell \rceil} < \frac{1}{\delta'}$, $n_i = 2n_{i-1}$, and $n_s = n$) with our choice of g to obtain a code $C \subseteq \mathbb{F}^{[n]}$ which is an

$$\left((s-1)bd 2^{\lceil \log \ell \rceil} + g \frac{b_s d_s n_s}{m_s} + 1 = O(sbd\ell + \delta/(\delta')^2), \delta, \varepsilon \right)\text{-RLCC},$$

8:14 Asymptotically-Good RLCCs with $(\log n)^{2+o(1)}$ Queries

where

$$\varepsilon \leq 1 - \min \left\{ \frac{1}{2} \alpha d 2^{\lceil \log \ell \rceil} \delta' - \frac{1}{b}, (1/e)^{g(\alpha d_s n_s \delta' / (2m_s) - 1/b_s)} = (1/e)^{g \alpha d_s n_s \delta' / (4m_s)} = 1/e \right\}.$$

As

$$\begin{aligned} \frac{1}{2} \alpha d 2^{\lceil \log \ell \rceil} \delta' - \frac{1}{b} &= \frac{\alpha \beta}{2} - \frac{1}{b} \\ &= \frac{\alpha \beta}{2} - \frac{\alpha \beta}{4} \\ &= \frac{\alpha \beta}{4}, \end{aligned}$$

we see that $\varepsilon \leq 1 - \min\{\frac{\alpha\beta}{4}, 1/e\}$. To decrease the error to $\frac{1}{3}$, we apply Claim 4 with $h = O(1)$, and get that C is also a $(q, \delta, \frac{1}{3})$ -RLCC for

$$q = O(sbd\ell + \delta/(\delta')^2) = O(sd^2\ell + \delta d^2\ell^2).$$

Recall that $s \leq \log n$,

$$d = 2^{O((\log \log r)^3)} = 2^{O((\log \log \log n)^3)}$$

and $\ell = O(\log n \cdot \log \log n)$. Therefore,

$$q = O(sd^2\ell + d^2\ell^2) = O(d^2\ell^2) = \log^2 n \cdot 2^{O((\log \log \log n)^3)} = (\log n)^{2+o(1)}.$$

Lastly, as the rate ρ_i of every code C_i in the sequence is at least $1 - \frac{m_i}{n_i}$, Proposition 11 implies that the rate ρ of C is lower bounded by

$$\begin{aligned} \rho &\geq 1 - \sum_{i=1}^s \left(\frac{m_i}{n_i} \right) \\ &= 1 - (s-1) \frac{1}{2^{\lceil \log \ell \rceil}} - \frac{1}{2^{\lceil \log \frac{1}{\delta} + \log \beta - c_0 \log^3(\log \frac{1}{\delta}) \rceil}} \\ &= 1 - O\left(\frac{s}{\ell}\right) - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))} \\ &= 1 - O\left(\frac{r}{\ell}\right) - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))} \\ &= 1 - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))} - o(1). \end{aligned}$$

This concludes the proof. ◀

References

- 1 Ron Asherov and Irit Dinur. Bipartite unique-neighbour expanders via Ramanujan graphs. *arXiv preprint*, 2023. [arXiv:2301.03072](#).
- 2 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.
- 3 Eli Ben-Sasson and Michael Viderman. Composition of semi-ltcs by two-wise tensor products. *computational complexity*, 24:601–643, 2015.
- 4 Arnab Bhattacharyya, Sivakanth Gopi, and Avishay Tal. Lower bounds for 2-query LCCs over large alphabet. *arXiv preprint*, 2016. [arXiv:1611.06980](#).
- 5 Alexander R. Block, Jeremiah Blocki, Kuan Cheng, Elena Grigorescu, Xin Li, Yu Zheng, and Minshen Zhu. On relaxed locally decodable codes for Hamming and insertion-deletion errors. In *38th Computational Complexity Conference*, volume 264 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Paper No. 14, 25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. [doi:10.4230/lipics.ccc.2023.14](#).


- 6 Manuel Blum and Sampath Kannan. Designing programs that check their work. *Journal of the ACM (JACM)*, 42(1):269–291, 1995.
- 7 Clément L Canonne and Tom Gur. An adaptivity hierarchy theorem for property testing. *Computational Complexity*, 27(4):671–716, 2018.
- 8 Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 659–668. ACM, New York, 2002. doi:10.1145/509907.510003.
- 9 Alessandro Chiesa, Tom Gur, and Igor Shinkar. Relaxed locally correctable codes with nearly-linear block length and constant query complexity. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1395–1411. SIAM, 2020.
- 10 Gil Cohen and Tal Yankovitz. Rate amplification and query-efficient distance amplification for linear LCC and LDC. In *36th Computational Complexity Conference (CCC 2021)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.
- 11 Gil Cohen and Tal Yankovitz. LCC and LDC: Tailor-made distance amplification and a refined separation. In *49th EATCS International Conference on Automata, Languages, and Programming*, volume 229 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 44, 20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/lipics.icalp.2022.44.
- 12 Gil Cohen and Tal Yankovitz. Relaxed locally decodable and correctable codes: beyond tensoring. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science—FOCS 2022*, pages 24–35. IEEE Computer Soc., Los Alamitos, CA, 2022.
- 13 Itay Cohen, Roy Roth, and Amnon Ta-Shma. HDX condensers. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2023.
- 14 Marcel Dall’Agnol, Tom Gur, and Oded Lachish. A structural theorem for local algorithms with applications to coding, testing, and privacy. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1651–1665. SIAM, 2021.
- 15 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.
- 16 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM Journal on Computing*, 40(4):1154–1178, 2011.
- 17 Zeev Dvir, Sivakanth Gopi, Yuzhou Gu, and Avi Wigderson. Spanoids—an abstraction of spanning structures, and a barrier for LCCs. *SIAM Journal on Computing*, 49(3):465–496, 2020.
- 18 Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM Journal on Computing*, 41(6):1694–1703, 2012.
- 19 Guy Goldberg. Linear relaxed locally decodable and correctable codes do not need adaptivity and two-sided error. In *Electron. Colloquium Comput. Complex.*, 2023.
- 20 Oded Goldreich. Lecture notes on locally testable codes and proofs, 2016. URL: <https://www.wisdom.weizmann.ac.il/~oded/PDF/pt-ltc.pdf>.
- 21 Oded Goldreich. On the lower bound on the length of relaxed locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2023.
- 22 Oded Goldreich and Tom Gur. Universal locally verifiable codes and 3-round interactive proofs of proximity for csp. *Theoretical Computer Science*, 878:83–101, 2021.
- 23 Oded Goldreich, Howard Karloff, Leonard J Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. In *Proceedings 17th IEEE Annual Conference on Computational Complexity*, pages 175–183. IEEE, 2002.
- 24 Louis Golowich. New explicit constant-degree lossless expanders. *arXiv preprint*, 2023. arXiv:2306.07551.
- 25 Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 64(8):5813–5831, 2018.

- 26 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 529–540. ACM, 2013.
- 27 Tom Gur and Oded Lachish. On the power of relaxed local decoding algorithms. *SIAM Journal on Computing*, 50(2):788–813, 2021.
- 28 Tom Gur, Govind Ramnarayan, and Ron Rothblum. Relaxed locally correctable codes. *Theory of Computing*, 16(1):1–68, 2020.
- 29 Tom Gur and Ron D Rothblum. A hierarchy theorem for interactive proofs of proximity. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.
- 30 Tom Gur and Ron D Rothblum. Non-interactive proofs of proximity. *computational complexity*, 27(1):99–207, 2018.
- 31 Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005.
- 32 Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local correctability of expander codes. *Information and Computation*, 243:178–190, 2015.
- 33 Jun-Ting Hsieh, Theo McKenzie, Sidhant Mohanty, and Pedro Paredes. Explicit two-sided unique-neighbor expanders. *arXiv preprint*, 2023. [arXiv:2302.01212](https://arxiv.org/abs/2302.01212).
- 34 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 80–86, 2000.
- 35 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *Journal of the ACM (JACM)*, 64(2):11, 2017.
- 36 Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM (JACM)*, 61(5):28, 2014.
- 37 Vinayak Kumar and Geoffrey Mon. Relaxed local correctability from local testing. In *Electron. Colloquium Comput. Complex.*, 2023.
- 38 Richard J Lipton. Efficient checking of computations. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 207–215. Springer, 1990.
- 39 Or Meir. Locally correctable and testable codes approaching the Singleton bound. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 21(107), page 14, 2014.
- 40 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *Journal of the ACM (JACM)*, 57(5):1–29, 2008.
- 41 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.
- 42 Noga Ron-Zewi and Ron D Rothblum. Local proofs approaching the witness length. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 846–857. IEEE, 2020.
- 43 Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Trans. Inform. Theory*, 42(6, part 1):1710–1722, 1996. Codes and complexity. [doi:10.1109/18.556667](https://doi.org/10.1109/18.556667).
- 44 Daniel Alan Spielman. *Computationally efficient error-correcting codes and holographic proofs*. ProQuest LLC, Ann Arbor, MI, 1995. Thesis (Ph.D.) – Massachusetts Institute of Technology. URL: http://gateway.proquest.com/openurl?url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:dissertation&res_dat=xri:pqdiss&rft_dat=xri:pqdiss:0576626.
- 45 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM (JACM)*, 55(1):1–16, 2008.

Lifting Dichotomies

Yaroslav Alekseev  

Technion – Israel Institute of Technology, Haifa, Israel

Yuval Filmus  

Technion – Israel Institute of Technology, Haifa, Israel

Alexander Smal  

Technion – Israel Institute of Technology, Haifa, Israel

Abstract

Lifting theorems are used for transferring lower bounds between Boolean function complexity measures. Given a lower bound on a complexity measure A for some function f , we compose f with a carefully chosen *gadget* function g and get essentially the same lower bound on a complexity measure B for the *lifted* function $f \diamond g$. Lifting theorems have a number of applications in many different areas such as circuit complexity, communication complexity, proof complexity, etc. One of the main questions in the context of lifting is how to choose a suitable gadget g . Generally, to get better results, i.e., to minimize the losses when transferring lower bounds, we need the gadget to be of a constant size (number of inputs). Unfortunately, in many settings we know lifting results only for gadgets of size that grows with the size of f , and it is unclear whether it can be improved to a constant size gadget. This motivates us to identify the properties of gadgets that make lifting possible.

In this paper, we systematically study the question “For which gadgets does the lifting result hold?” in the following four settings: lifting from decision tree depth to decision tree size, lifting from conjunction DAG width to conjunction DAG size, lifting from decision tree depth to parity decision tree depth and size, and lifting from block sensitivity to deterministic and randomized communication complexities. In all the cases, we prove the complete classification of gadgets by exposing the properties of gadgets that make lifting results hold. The structure of the results shows that there is no intermediate cases – for every gadget there is either a polynomial lifting or no lifting at all. As a byproduct of our studies, we prove the log-rank conjecture for the class of functions that can be represented as $f \diamond \text{OR} \diamond \text{XOR}$ for some function f .

In this extended abstract, the proofs are omitted. Full proofs are given in the full version [2].

2012 ACM Subject Classification Theory of computation \rightarrow Communication complexity; Theory of computation \rightarrow Oracles and decision trees

Keywords and phrases decision trees, log-rank conjecture, lifting, parity decision trees

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.9

Related Version *Full Version:* <https://eccc.weizmann.ac.il/report/2024/037/> [2]

Funding This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC. Alexander Smal has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No 852870-ERC-SUBMODULAR.

1 Introduction

For $f: \{0, 1\}^n \rightarrow V$ and $g: \{0, 1\}^m \rightarrow \{0, 1\}$, a (*block-*)*composition* $f \diamond g: \{0, 1\}^{n \times m} \rightarrow V$ is defined by

$$(f \diamond g)(z_1, z_2, \dots, z_n) := f(g(z_1), g(z_2), \dots, g(z_n)),$$

where each $z_i \in \{0, 1\}^m$. Usually *lifting theorems* have the following general form:

$$B(f \diamond g) = \Omega(A(f)),$$



© Yaroslav Alekseev, Yuval Filmus, and Alexander Smal;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 9; pp. 9:1–9:18

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



where A and B are two complexity measures. Note that the hidden constant in $\Omega(\cdot)$ may depend on g . In this context, we call the function g a *gadget* and say that *there is a (linear) lifting from A to B* . One of the first examples of a lifting theorem appeared in [19] where Raz and McKenzie proved a separation for the hierarchy of monotone circuit complexity classes within NC using a query-to-communication lifting theorem.

The need for such theorems is due to the fact that, in many cases, communication complexity lower bounds are much more difficult to prove compared to query complexity lower bounds. Lifting theorems proved to be useful in many other scenarios: proving communication complexity separations [10, 11, 12, 5], proof complexity separations [9, 14, 10, 7], monotone circuit complexity separations [19, 8], etc.

While the lifting technique is widely used in different areas we still do not quite understand its limitations. In the query-to-communication lifting theorems, we want to lower bound the deterministic/randomized communication complexity of the lifted function by deterministic/randomized query complexity of the original function. So, these theorems usually have the following form:

$$D(f \diamond g) = DT(f) \cdot \Theta(\log n),$$

where the gadget g has at most logarithmic communication complexity. In all the lifting theorems of this type, the *size* (the number of inputs) of the gadget g grows with the number of inputs of f (e.g., in [17] the size of the gadget is $n^{1+\epsilon}$, where n is a number of inputs of the lifted function). Even though some results do not depend on the gadget size (e.g., [11]), in many scenarios the use of non-constant size gadgets leads to weaker results in the applications (e.g., when lifting theorems are used to prove monotone circuit lower bounds via communication complexity). It is unknown whether it is possible to prove a query-to-communication lifting theorem with a constant size gadget, but we tend to believe that such lifting exists.

At the same time, for other complexity measures there are lifting theorems that can accommodate constant size gadgets. Here are some examples of such lifting theorems:

1. Lifting decision tree depth to decision tree size (XOR gadget [21]).
2. Lifting decision tree depth to parity decision tree depth/size (stifling gadgets [6]; INDEX gadget [3]).
3. Lifting from critical block sensitivity to communication complexity (VER gadget [10]).
4. Lifting parity decision tree depth to communication complexity (XOR gadget [13]).
5. Lifting AND decision tree depth to communication complexity (AND gadget [15]).
6. Lifting block sensitivity to randomized communication complexity (AND gadget [22]).

Note that in examples 4 and 5, the communication complexity is lower bounded by *some power* of the parity decision tree complexity or the AND decision tree complexity (in the latter case, with an additional $\log n$ factor). In such cases, we say that there is a *polynomial lifting*.

As mentioned earlier, lifting theorems which lower bound communication complexity by a *linear* function of query complexity are only known for non-constant size gadgets. But in case of a polynomial lifting, there are query to communication complexity lifting theorems with constant size gadgets. E.g., in example 6 the author lifts block sensitivity to randomized communication complexity with a constant size gadget. Given that block sensitivity is polynomially related to query complexity, this gives a polynomial lifting from query to communication complexity.

All the examples of the lifting theorems with constant size gadgets that we mentioned above use specific and simple gadgets. But what happens if we plug some other gadget? The same proof might not work, but would the lifting result still be true? That is not always clear. And this reflects our lack of understanding of what properties of the gadget make lifting possible. For these theorems it is natural to pose the following question:

For which gadgets does the lifting result hold?

A systematical study of this question will help us to better understand how lifting works and what are the requirements for the gadget. Especially it would be interesting to understand for which gadgets lifting fails. This is what we need, for example, if we want to find a good candidate for query-to-communication linear lifting with constant size gadget.

One of the areas that might benefit from new lifting theorem is the study of *the log-rank conjecture* [18]. The log-rank conjecture states that for any function f the deterministic communication complexity of a function is polynomially related to the logarithm of the real rank of its associated communication matrix. Currently there is an exponential gap between the lower bound $\Omega(\log^2(\text{rank}(f)))$ [11] and the upper bound $O(\sqrt{\text{rank}(f)} \cdot \log \text{rank}(f))$ [16]. It seems that the general case of this problem is out of reach now. So most of research in this area has concentrated on the study of various complexity measures and special cases such as composed functions (e.g., see [15]). Therefore, lifting in this area is one of the main proof techniques.

The other area craving for new lifting theorems is proof complexity. There is a tight connection between proof complexity and lifting theorems. For example, lifting decision tree depth to parity decision tree depth/size from [6, 3] provides a systematic way to prove tree-like $\text{Res}(\oplus)$ size lower bounds. It is important to mention that for proof complexity we usually need lifting theorems that hold for relations.

Finally, we think that a large number of different applications makes lifting a technique that is worth exploring on its own. A deeper understanding of how lifting works and what it requires from gadgets can lead to new applications and results.

1.1 Our results and methods

In this paper, we systematically explore the question “*For which gadgets does the lifting result hold?*” in several different settings. We prove complete classifications of gadgets in the following four settings:

- lifting from decision tree depth to decision tree size,
- lifting from certificate complexity to conjunction DAG size,
- lifting from decision tree depth to parity decision tree depth and size,
- lifting from block sensitivity to deterministic and randomized communication complexities.

All these results are formulated in the form of dichotomies (a trichotomy in one of the cases) that essentially state that there is either polynomial lifting or no lifting at all (no intermediate cases). As a byproduct of our studies, we prove the log-rank conjecture for the class of functions that can be represented as $f \diamond \text{OR} \diamond \text{XOR}$ for some function f .

Now we describe the results in more detail and give an overview of the proof methods. The proofs are omitted from this extended abstract. Full proofs are given in the full version [2].

1.1.1 Decision tree depth to size

In Section 3, we define a class of *resistant* gadgets (defined in Section 3.1) and prove a decision tree depth to decision tree size lifting theorem for this class of gadgets (see Theorem 2). We give two different proofs illustrating the ideas of two different approaches: proof by simulation

and proof using random projections. The proofs later in the paper refer to the proofs in this simple case. The proof by simulation is constructive – it shows how given a decision tree for the lifted function $f \diamond g$ one can construct a decision tree for the original function f , such that the depth of the new tree is bounded by a logarithm of the size of the given tree. In the proof using random projections, we use probabilistic method to show that there is a projection that converts the decision tree for $f \diamond g$ into a shallow decision tree for f .

Our goal is to prove a classification theorem, so we need to find a class of gadgets such that lifting works only for gadgets in this class. It appears that the class of resistant gadgets is too small for this. We define a wider class of *weakly resistant* gadgets (defined in Section 3.2) and prove a certificate complexity to decision tree size lifting theorem (see Theorem 4). The proof uses the random projections method. Note that the decision tree size is upper bounded by the certificate complexity squared, so as a corollary we get a polynomial lifting from decision tree depth to decision tree size (see Corollary 5).

Finally, we give a complete classification of gadget functions in the context of polynomial lifting from decision tree depth to decision tree size. It is stated as a dichotomy result (see Theorem 7): either a gadget is weakly resistant and there is a polynomial lifting or there is no lifting at all.

In Section 3.4, we briefly discuss that some of the results above can be generalized to the case of search problems. Unfortunately, that does not give a classification theorem because decision tree depth of a search problem can be exponentially greater than its certificate complexity. However, we state a conjecture there is a decision tree depth to decision tree size polynomial lifting for weakly resistant gadgets.

1.1.2 Conjunction DAG width to size

In Section 4, we generalize the results from the previous section to decision conjunction DAGs (defined in Section 4.1). First of all, in Section 4.2, we show that similarly to decision trees where depth and size are exponentially separated, there is an exponential separation between conjunction DAG width and size. The separation is achieved for the Tribes function. In Section 4.3, we show that there is an exponential separation between decision tree size and conjunction DAG size in the case of relations. Indeed, conjunction DAGs can capture the structure of Resolution proofs, while decision trees can only capture the structure of tree-like Resolution proofs. Thus, the separation between these proof systems implies the separation between the measures under consideration. Finally, we argue that the lifting theorems from Section 3 can be generalized to the case of conjunction DAGs (see Theorem 8 and Theorem 9) using essentially the same proofs. Thus, we have a dichotomy result (see Theorem 10).

1.1.3 Decision tree depth to parity decision tree depth and size

Recently, Chattopadhyay et al. [6] showed that if g is *stifling* (defined in Section 5.1) then $\log \text{DTSize}_{\oplus}(f \diamond g) = \Theta(\text{DT}(f))$. In Section 5, we extend their result providing the complete classification of the gadgets for decision tree depth to parity decision tree depth and size lifting. In Section 5.2, we state and prove a “minimum weight lemma” (see Lemma 13), the technical lemma that we will use several times. This lemma states that if the certificate complexity of some function f is large enough in comparison to the parity certificate complexity of the lifted function $f \diamond g$ at some input, then there is a substitution such that the minimal parity certificate of $f \diamond g$ at this input has a large Hamming weight.

In Section 5.3, we consider the most challenging case of this setting – the case of OR gadget. In Theorem 14, we show that there is at least a quadratic gap in the certificate to parity certificate complexity lifting for OR gadget. If we assume that the lifting result

in [6] holds for OR gadget as well then the quadratic separation is tight (see Proposition 15). In Section 5.3.2, we show a polynomial (cubic) lifting for OR gadget (see Theorem 16). The proof consists of three ingredients. First, we show that if the minimum weight of a parity certificate for the lifted function is at least the size of the parity certificate then this certificate covers the all-1 input for the inner function. Then we use it to upper bound the minimum weight of a parity certificate in terms of the sizes of parity certificates for 0 and 1. And finally, we apply the “minimum weight lemma”. In the proof of Theorem 16, we assume that the certificate complexity of the original function is much larger than the parity certificate complexity of the lifted function, but due to the “minimum weight lemma” it would contradict the upper bound on the minimum weight of the parity certificate.

In Section 5.4, we prove a lifting from certificate complexity to parity decision tree size for AND/OR gadgets, the gadgets that affine project to both binary AND and binary OR (see Theorem 17). The proof is by simulation similar to the simulation proof in [6].

Finally, in Section 5.5, we prove the trichotomy result that gives us a complete classification of gadgets (see Theorem 19). The classification is based on the fact that if a gadget is not AND/OR then it is a disjunction or a conjunction of affine forms (see Lemma 18). We show that there are only three cases possible: (1) there are simultaneously a lifting from decision tree depth to parity decision tree depth and a lifting from certificate complexity to parity certificate complexity (the case of AND/OR gadgets); (2) there is a lifting from decision tree depth to parity decision tree depth but no lifting between certificate complexities (the case of gadgets that affine project to OR); (3) there is no lifting (all other gadgets, constant or affine).

In Section 5.6, we show an alternative proof for the lifting from decision tree depth to parity decision tree depth using function degree and sparsity (see Theorem 20). In the proof we show that the degree of a function is upper bounded by the logarithm of the sparsity of the function lifted with OR gadget, and compose it with a number of previously known inequalities. As a byproduct, we get a proof of the log-rank conjecture for the class of functions that can be represented as $f \diamond \text{OR} \diamond \text{XOR}$ (see Theorem 21).

1.1.4 Block sensitivity to communication complexity

In Section 6, we provide a complete classification of gadgets for lifting from block sensitivity to deterministic and randomized communication complexities. Note that this classification also gives us a classification of gadgets for query-to-communication polynomial lifting since block sensitivity and query complexity are polynomially related to each other in the case of total functions. However, since the proof goes through block sensitivity, we classify block sensitivity to communication complexity lifting.

We start with the lifting theorem of Zhang [22] (see Theorem 23) that works for gadget g iff both AND and OR reduce to g via a communication complexity reduction (defined in Section 6.1). Then we show that if there is no reduction from OR to a gadget g then the communication matrix of g is (up to rearrangement) block diagonal (see Lemma 24). This gives us a classification of gadgets in the context of reductions from OR, AND, and XOR (see Corollary 26 and Corollary 27).

In Section 6.2, we prove a dichotomy result for randomized communication complexity (see Theorem 28). The proof is based on the fact that if OR does not reduce to a gadget g then $\text{AND}_n \diamond g$ is essentially an instance of the equality function, and hence its randomized communication complexity is logarithmic. If AND does not reduce to a gadget, the situation is symmetrical (see Lemma 29). Thus, the theorem of Zhang covers all gadgets for which there is a lifting.

In Section 6.3, we prove the dichotomy result for deterministic communication complexity (see Theorem 31). The proof of the dichotomy depends on a block sensitivity to deterministic communication complexity lifting theorem that works for gadget g iff both AND and XOR reduce to g (see Theorem 30). Together with the Zhang’s lifting theorem that give us a complete classification of gadgets: if at least two of three functions AND, OR, XOR reduce to gadget g then there is a block sensitivity to deterministic communication complexity polynomial lifting. Otherwise, if at least two functions from this list do not reduce to g then g is either a constant function or (essentially) one of the functions AND, OR, XOR. The proof of the lifting theorem requires a number of ingredients. First of all, we compose two results from [13, 23] to get a parity certificate to deterministic communication complexity polynomial lifting for XOR gadget. Then we show two lower bounds for specific cases: we show that the deterministic communication complexity of $f \diamond \text{XOR}$ is lower bounded by the size of any minimal sensitive block of the function f , and that the deterministic communication complexity of $f \diamond \text{AND}$ is lower bounded by the block sensitivity of f at the all-0 input. One of the ingredients is again the “minimum weight lemma”. And the last one, is the lemma that shows that parity certificates of high minimum weight always intersect with (regular) certificates of sufficiently small size. Putting all the ingredients together we get the desired lifting result.

2 Prerequisites

2.1 Notation

All the logarithms are base 2. We use OR_n , AND_n , XOR_n to denote, respectively, logical “and”, “or” and “exclusive or” of n Boolean inputs. For simplicity we also define $\text{OR} := \text{OR}_2$, $\text{AND} := \text{AND}_2$, and $\text{XOR} := \text{XOR}_2$. We use the notation x^B for $x \in \{0, 1\}^n$ and $B \subseteq [n]$ to denote x with all the coordinates in B flipped, i.e., $x_i = x_i^B \iff i \notin B$. For $g: \{0, 1\}^k \rightarrow \{0, 1\}$, we define¹ a function $g^n: \{0, 1\}^{n \times k} \rightarrow \{0, 1\}^n$ such that

$$g^n(x_1, x_2, \dots, x_n) := (g(x_1), g(x_2), \dots, g(x_n)).$$

For a function $f: \{0, 1\}^n \rightarrow \{0, 1\}$, an input $x \in \{0, 1\}^n$, and a partial assignment $\sigma \in \{0, 1, *\}^n$ that agrees with x on all non- $*$ coordinates, we use $f|_\sigma$ to denote a restriction of f to σ and $x|_\sigma$ to denote a projection of x to the $*$ -coordinates of σ . For a gadget $g: \{0, 1\}^k \rightarrow \{0, 1\}$ and a *blockwise* partial assignment $\sigma \in (\{0, 1\}^k \cup \{*\}^k)^n$ (i.e., in any block of variables that corresponds to one copy of g , the variables are either all set or all stars), we use $g^n(\sigma)$ to denote the partial assignment in $\{0, 1, *\}^n$ induced by applying g to non- $*$ blocks of σ . We use “ \sqcup ” instead of “ \cup ” to indicate a union of disjoint sets.

2.2 Complexity measures

Throughout the text, we will consider several complexity measures.

Decision tree complexity

A *decision tree* is a rooted binary tree with internal nodes labeled by input variables (represent queries), edges labeled with 0 and 1, and leaves labeled by some values. Decision tree evaluation is defined in the natural way: given an assignment for the input variables,

¹ In some papers lifting theorems are formulated for the classical composition operation “ \circ ” rather than for the block-composition “ \diamond ”. Note that for $f: \{0, 1\}^n \rightarrow \{0, 1\}$ and $g: \{0, 1\}^k \rightarrow \{0, 1\}$, $f \diamond g \equiv f \circ g^n$.

we traverse the tree starting from the root and then sequentially choose every next edge according to the given assignments until we reach some leaf. The label of this leaf is the result of the evaluation. A decision tree computes some function f if for all possible assignments to the input variables the decision tree evaluates to the value of the function. For a function f , $\text{DT}(f)$ denotes the minimal depth of a decision tree computing f , and $\text{DTSize}(f)$ denotes the minimal number of leaves in a decision tree computing f .

Parity decision tree complexity

A *parity decision tree* is a generalization of a decision tree where the queries are arbitrary linear combinations of the input variables. It can be described as a rooted binary tree with internal nodes labeled by linear combinations of the input variables, edges labeled with 0 and 1, and leaves labeled by some values. The evaluation is defined analogously. For a function f , $\text{DT}_{\oplus}(f)$ denotes the minimal depth of a parity decision tree computing f , and $\text{DTSize}_{\oplus}(f)$ denotes the minimal number of leaves in a parity decision tree computing f .

Certificate complexity

A *certificate complexity* is a non-deterministic analogue of the decision tree complexity. A *certificate* for an input $x \in \{0, 1\}^n$ to a function f is a set $S \subseteq [n]$ of indices such that f restricted to all inputs that match x on S is constant, i.e., $f(y) = f(x)$ whenever $y|_S = x|_S$. The certificate complexity $C(f, x)$ of f at input x is the size of the smallest certificate for x . Finally, the certificate complexity of function f is defined as $C(f) := \max_{x \in \{0, 1\}^n} C(f, x)$.

Parity certificate complexity

A *parity certificate* for an input $x \in \{0, 1\}^n$ to a function f is a set A of linear forms such that f is constant on the affine subspace defined by $A = A(x)$. A *parity certificate complexity* $C_{\oplus}(f, x)$ of function f at input x is the size (number of linear forms) of the smallest parity certificate for x . The parity certificate complexity of f is defined as $C_{\oplus}(f) := \max_{x \in \{0, 1\}^n} C_{\oplus}(f, x)$.

Block sensitivity

A block $B \subseteq [n]$ is *sensitive* for a function f at x iff $f(x) \neq f(x^B)$. The *block sensitivity* $\text{bs}(f, x)$ of f at x is the maximum number of disjoint sensitive blocks for f at x . The block sensitivity $\text{bs}(f)$ of f is the maximum sensitivity $\text{bs}(f, x)$ over all points $x \in \{0, 1\}^n$.

Degree and sparsity of the function

Every Boolean function defined on $\{0, 1\}^n \rightarrow \{0, 1\}$ can be considered as a function $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$ by substituting 0 and 1 with 1 and -1 , respectively. Every such function has a unique representation as a multilinear polynomial over \mathbb{R} of the following form

$$f(x) = \sum_{S \subseteq [n]} c_S x_S, \quad \text{where } x_S = \prod_{i \in S} x_i.$$

Degree of f is defined to be the degree of the corresponding polynomial, we denote it by $\text{deg}(f)$. *Sparsity* of f is the number of nonzero coefficients in this representation, we denote it by $\text{spar}(f)$.

Deterministic communication complexity and rank

For a function $f: X \times Y \rightarrow Z$, let's consider the following game for two players, Alice and Bob, that want to compute f . Alice and Bob are given $x \in X$ and $y \in Y$, respectively. Their goal is to compute $f(x, y)$. In order to do it, the players exchange information about their parts of the input using a simple communication channel that allows sending bit messages. Before the game the players come up with a *communication protocol* that determines their behavior on all possible inputs. The cost of a protocol is the maximum total number of bits sent by the players over all possible inputs $x \in X, y \in Y$. The *deterministic communication complexity* of f is the minimal cost of a deterministic communication protocol that computes f . We denote it by $D(f)$.

A *communication matrix* of the function f is a matrix $M_f \in Z^{X \times Y}$ such that $(M_f)_{x,y} := f(x, y)$ for all $x \in X, y \in Y$. We define $\text{rank}(f)$ to be the rank of matrix M_f .

Randomized communication complexity

In a *randomized communication game*, Alice and Bob have access to an unlimited amount of random bits and they can use it to decide which bit to send next. We say that a (*private coin*) *randomized communication protocol* Π computes a function $f: X \times Y \rightarrow Z$ with error ε if

$$\Pr_{r_A, r_B} [\Pi(x, y, r_A, r_B) = f(x, y)] \geq 1 - \varepsilon, \quad \forall x \in X, y \in Y,$$

where r_A and r_B are the strings of random bits used by Alice and Bob, respectively. The cost of a randomized protocol is the maximum number of bits that can be sent. We denote by $R_\varepsilon(f)$ the minimal cost of a private coin randomized communication protocol that computes f with error ε . A *randomized communication complexity* of f is defined as $R(f) := R_{1/3}(f)$.

3 Decision tree depth to size

We start by exploring perhaps one of the simplest scenarios, depth-to-size lifting in decision trees and state a lifting theorem for the class of *resistant* gadgets.

3.1 Resistant gadgets

Urquhart [21] proved that for any function $f: \{0, 1\}^n \rightarrow \{0, 1\}$,

$$\log \text{DTSize}(f \diamond \text{XOR}) = \Omega(\text{DT}(f)).$$

We generalize this result for the class of *resistant* gadgets.

► **Definition 1.** A gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$ is resistant if for every $i \in [m]$ and $b \in \{0, 1\}$, the function obtained by fixing the i th input to b is not constant. Equivalently, the minimum certificate complexity at inputs is larger than 1. E.g., XOR function is resistant while AND function is not.

► **Theorem 2.** For any $f: \{0, 1\}^n \rightarrow \{0, 1\}$ and a resistant gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$,

$$\log \text{DTSize}(f \diamond g) = \Omega(\text{DT}(f)).$$

3.2 Weakly resistant gadgets

There are gadgets, such as $x \vee (y \wedge z)$, which are clearly not resistant, but for which the lifting still holds as we will show below. To capture these cases, we define a more general class of *weakly resistant* gadgets.

► **Definition 3.** A gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$ is weakly resistant if for every certificate α (a partial assignment which sets the value of g) there is a partial assignment $y_j = b$ which conflicts with α and does not make g constant.

Every resistant gadget is trivially weakly resistant: we can take any variable mentioned in α and substitute the opposite value. The aforementioned gadget $h = x \vee (y \wedge z)$ is weakly resistant (as we show below) but not resistant, since $h|_{x=1} = 1$. The gadget $x \vee y$ is not even weakly resistant due to the certificate $x = y = 0$: if we substitute $x = 1$ or $y = 1$ then the gadget becomes constant.

Let us verify that h is weakly resistant, by considering all possible certificates:

- $x = 1$: take $x = 0$.
- $y = z = 1$: take $y = 0$ or $z = 0$.
- $x = y = 0$: take $y = 1$.
- $x = z = 0$: take $z = 1$.

We prove the following lifting theorem:

► **Theorem 4.** For any $f: \{0, 1\}^n \rightarrow \{0, 1\}$ and a weakly resistant gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$ the following holds:

$$\log \text{DTSize}(f \diamond g) = \Omega(\text{C}(f)).$$

Decision tree complexity is at most certificate complexity squared, thus we get the following corollary.

► **Corollary 5.** For any $f: \{0, 1\}^n \rightarrow \{0, 1\}$ and a weakly resistant gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$ the following holds:

$$\log \text{DTSize}(f \diamond g) = \Omega(\sqrt{\text{DT}(f)}).$$

We do not know whether the lower bound is tight, even for $h = x \vee (y \wedge z)$. At the moment we can't even rule out $\log \text{DTSize}(f \diamond h) = \Omega(\text{DT}(f))$. Sherstov [20, Theorem 6.4] proved that

$$\max(\log \text{rank}(f \diamond \text{AND}), \log \text{rank}(f \diamond \text{OR})) \geq \deg(f).$$

Since rank lower bounds decision tree size, this implies that

$$\log \text{DTSize}(f \diamond g) \geq \deg(f).$$

Note that we are not aware of any separation between $\text{DT}(f)$ and $\max(\text{C}(f), \deg(f))$.

3.3 Gadget classification

The dichotomy result of this section is based on the following classification of gadgets.

► **Lemma 6.** For every $g: \{0, 1\}^m \rightarrow \{0, 1\}$, one of the following cases holds:

1. Function g is a (possibly empty) conjunction or a disjunction of literals.
2. Function g is weakly resistant.

The two cases in following dichotomy theorem correspond to the two cases of Lemma 6.

- **Theorem 7.** *For every gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$, one of the following cases holds:*
1. *There is an infinite family of functions f_n with $\text{DT}(f_n) \rightarrow \infty$ and $\text{DTSize}(f \diamond g) = O(\text{DT}(f))$.*
 2. *For every function f , we have $\log \text{DTSize}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$.*

3.4 Generalization to search problems

A relation f is a subset of $\{0, 1\}^n \times V$. For a gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$, we define a composition $f \diamond g \subseteq (\{0, 1\}^m)^n \times V$ as a relation, such that

$$(z_1, z_2, \dots, z_n, r) \in (f \diamond g) \iff (g(z_1), g(z_2), \dots, g(z_n), r) \in f.$$

One can observe that the proofs of Theorems 2 and 4 work as well when we let f to be a relation. However, the Corollary 5 does not hold for the relations since $\text{DT}(f)$ can be exponentially greater than $C(f)$. As an example of this, one can take f to be a *falsified clause problem* corresponding to the Pigeonhole Principle Formula over an expander graph. It is known [4] that resolution width of any refutation for this formula at least $\Omega(n)$ (which is greater or equal than $\text{DT}(f)$), but the certificate complexity is constant (since each of the clauses of the formula is constant-sized).

However, we conjecture that lifting from decision tree depth to decision tree size can also be proved for weakly resistant gadgets. Equivalently, this will mean that such lifting holds for the gadget $g(x, y, z) := x \vee (y \wedge z)$.

4 Conjunction DAG width to size

4.1 Conjunction DAGs

Conjunction DAGs were first defined formally in [8], though they appear implicitly in previous work. A *conjunction DAG* over a set of variables is a single-rooted DAG with the following additional information:

- Each internal vertex is annotated with a variable, and it has two outgoing edges, one labeled 0 and the other one labeled 1.
- Each vertex v is annotated with a partial assignment $\rho(v)$ with the following constraint. Suppose that v queries x_i , and the answer b leads to the vertex v_b . Then the partial assignment $\rho(v_b)$ is a subset of the partial assignment $\rho(v) \cup \{x_i \leftarrow b\}$. (This is not identical to the definition in [8], but morally the same.)
- The partial assignment at the root is the empty assignment.
- Each leaf is annotated with some value.

A decision DAG *computes* f if for every leaf ℓ annotated with y_ℓ , $\rho(\ell)$ is a y_ℓ -certificate of f .

Every decision tree is a decision DAG. There are three parameters of interest: the (total) size (number of vertices), the leaf size (number of leaves), and the width (maximum number of variables in any $\rho(v)$).

4.2 Size vs width

A decision tree of depth d contains at most 2^d leaves, and this is tight for the parity function, in the sense that the bound $\text{DTSize}(f) \leq 2^{\text{DT}(f)}$ cannot be improved when f is the parity function.

Similarly, a conjunction DAG of width d contains at most $\binom{n}{\leq d} = O(n^d)$ vertices, and this is tight for the following function (also known as the *Tribes* function)

$$f(x) = \bigvee_{i=1}^{\sqrt{n}} \bigwedge_{j=1}^{\sqrt{n}} x_{ij}.$$

We can construct a conjunction DAG of width $O(\sqrt{n})$ for f as follows. We think of the input as a matrix, where i is the row number and j is the column number. We scan each row sequentially. If the current row consists only of 1s, we stop. Otherwise, we add the first 0 to ρ , and forget all the remaining entries of the row.

Conversely, consider the set of $\sqrt{n}^{\sqrt{n}}$ inputs having a single 0 per row. No two inputs share the same certificate, and so every conjunction DAG for f must contain at least $n^{\sqrt{n}/2}$ leaves.

4.3 Separation between decision tree size and conjunction DAG size

Let $d(f)$ denotes the conjunction DAG width of a function f , which is the smallest width of a conjunction DAG for f . Since $\text{DT}(f) \geq d(f) \geq C(f) = \Omega(\sqrt{\text{DT}(f)})$, in case of functions conjunction DAG width and decision tree depth are polynomially related. In this section, we show that in case of relations decision tree size and conjunction DAG size can be far apart.

Alekhovich et al. [1] constructed a family of CNF contradictions ϕ_n with $\text{poly}(n)$ many variables and clauses which have a refutation in Resolution of size $\text{poly}(n)$, but such that any refutation in tree-like Resolution (even in regular Resolution) is of size $2^{\Omega(n)}$.

We can think of a Resolution proof as a conjunction DAG which solves the *falsified clause problem*: given a truth assignment, find a falsified clause. This is a relation rather than function. Similarly, a tree-like Resolution proof is a decision tree solving the same problem.

Now we are going to show that there is a function $f: \{0, 1\}^n \rightarrow \{0, 1\}$ with a conjunction DAG of size $\text{poly}(n)$ such that $\text{DTSize}(f) = 2^{\Omega(n/\log n)}$. Consider the contradictions ϕ_n mentioned above. Index the clauses of ϕ_n using bitstrings of length $\ell = O(\log n)$ in some arbitrary way. Now consider a polynomial size Resolution proof of ϕ_n , and let f_i be the function mapping a truth assignment to the i -th bit of the bitstring indexing the falsified clause found by the proof. By construction, each f_i has a conjunction DAG of size $\text{poly}(n)$. Given decision trees for f_1, \dots, f_ℓ , we can construct a decision tree solving the falsified clause problem of size $\prod_i \text{DTSize}(f_i)$. Since this product must be at least $2^{\Omega(n)}$, we conclude that $\max_i \text{DTSize}(f_i) = 2^{\Omega(n/\log n)}$.

4.4 Gadget classification

For the case of conjunction DAGs, we can generalize Theorem 2:

► **Theorem 8.** *Let $f \subseteq \{0, 1\} \times V$ be a relation and $g: \{0, 1\}^m \rightarrow \{0, 1\}$ be a resistant gadget. If there is a conjunction DAG of size S computing $f \diamond g$, then there is a conjunction DAG of width $O(\log S)$ computing f .*

Moreover, for conjunction DAGs we can prove an analogue of Theorem 4:

► **Theorem 9.** *Let $f \subseteq \{0, 1\} \times V$ be a relation and $g: \{0, 1\}^m \rightarrow \{0, 1\}$ be a weakly resistant gadget. If there is a conjunction DAG for $f \diamond g$ with S leaves, then the certificate complexity of f is at most $O(\log S)$.*

Note that Theorem 9 gives us a lifting from certificate complexity to leaf size. Unfortunately, in the case of the falsified clause problem for CNF, this theorem cannot be effectively used to prove lower bounds since leaf size is usually small as well as certificate complexity. However, this theorem still implies a dichotomy result similar to one in Theorem 7.

- **Theorem 10.** *For every gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$, one of the following cases holds:*
1. *There exists an infinite family of functions f_n such that $f_n \diamond g$ can be computed with a conjunction DAG having $O(n)$ leaves, but $C(f_n) = \Omega(n)$.*
 2. *For every relation f , if we have a conjunction DAG for $f \diamond g$ with S leaves, then the certificate complexity of f is at most $O(\log S)$.*

5 Decision tree depth to parity decision tree depth and size

In this section, we are going to classify gadgets in the context of decision tree depth to parity decision tree depth and size lifting. We start by restating recent result of Chattopadhyay et al. [6]. After that we are going to state the “minimum weight lemma” (Lemma 13) that is a technical tool we will use multiple times throughout this section and the following one. We use this lemma to prove the lifting from certificate complexity to parity certificate complexity with OR gadget, which is the most challenging case in the classification. Finally, we will discuss an alternative and much simpler proof for the lifting from decision tree depth to parity decision tree depth using degree and sparsity. In some sense, this proof should give us a better exponent for the decision tree lifting. The main point of considering certificate complexity lifting is that there is a non-trivial upper bound for OR gadget showing that it is impossible to prove a linear lifting in this setting (see Theorem 14). As a byproduct, we get a proof of the log-rank conjecture for the class of functions that can be represented as $f \diamond \text{OR} \diamond \text{XOR}$.

5.1 Stifling gadgets

Chattopadhyay et al. [6] defined the following notion.

- **Definition 11.** *A function $g: \{0, 1\}^m \rightarrow \{0, 1\}$ is k -stifling if for every set of k coordinates and $b \in \{0, 1\}$ there is a way to set the remaining $m - k$ coordinates so that the output is b (regardless of the value of the chosen k coordinates). A function is stifling if it is 1-stifling.*

Chattopadhyay et al. [6] showed that if g is stifling then $\log \text{DTSize}_{\oplus}(f \diamond g) = \Theta(\text{DT}(f))$, where the hidden constant can depend on g . See the full version of this paper [2] for an exposition of their proof.

5.2 Minimum weight lemma

A parity certificate C is a system of affine equations, so it can be represented by a matrix M and a vector v that define a linear subspace.

- **Definition 12.** *A minimum weight of a parity certificate C is the minimum Hamming weight of non-zero vectors in the row space of M .*

For example, one can consider a certificate $\{x + y = 1, x + y + z = 0\}$. The minimum weight of this certificate is equal to 1 and this corresponds to the equation $z = 1$. On the other hand, the minimum weight of a certificate $\{x + y = 1, x + z = 0\}$ is equal to 2.

The following lemma is a key tool that we will use both in Section 5 and in Section 6. This lemma shows that if the certificate complexity of some function f is large enough in comparison to the parity certificate complexity of the lifted function $f \diamond g$ at some input,

then there is a substitution such that the minimal parity certificate of $f \diamond g$ at this input has large Hamming weight. Informally, this means that we can make a small enough substitution, such that the preimage of the parity certificate after the substitution will contain the points that we are interested in.

► **Lemma 13** (minimal weight lemma). *For functions $f: \{0, 1\}^n \rightarrow \{0, 1\}$ and $g: \{0, 1\}^m \rightarrow \{0, 1\}$, suppose that $f \diamond g$ has a parity certificate complexity at most k at some point x . Let K be a parameter. If certificate complexity of f at $g^n(x)$ is greater than $k \cdot K$ then we can find a blockwise partial assignment σ to the inputs of $f \diamond g$ consistent with x , such that for some $k' \leq k$:*

- $(f \diamond g)|_\sigma$ has a parity certificate of size k' at $x|_\sigma$ whose minimum weight is at least K ,
- $C(f|_{g^n(\sigma)}, g^n(x)|_{g^n(\sigma)}) > k'K$.

5.3 OR gadget

5.3.1 Separation

Chattopadhyay et al. [6] showed that $C_\oplus(f \diamond g) = \Theta(C(f))$ whenever g is stifling. This no longer holds when g is binary OR. The following theorem shows that there is at least a quadratic gap for OR gadget in the certificate to parity certificate complexity lifting.

► **Theorem 14.** *Let $n = m^2$, and let $f: \{0, 1\}^n \rightarrow \{0, 1\}$ be the function that accepts an $m \times m$ Boolean matrix iff it has no rows of Hamming weight 1.*

1. $C(f) = n$,
2. $C_\oplus(f \diamond \text{OR}) \leq 2m$.

Note that if we assume that the lifting result of [6] holds for the OR gadget as well then the quadratic separation is tight.

► **Proposition 15.** *Suppose that $\text{DT}_\oplus(f \diamond \text{OR}) = \Theta(\text{DT}(f))$ for all f . Then every function f satisfies $C(f) = O(C_\oplus(f \diamond \text{OR})^2)$.*

5.3.2 Lifting

Given Lemma 13 we prove the following lifting theorem for the certificate complexity.

► **Theorem 16.** *For every function $f: \{0, 1\}^n \rightarrow \{0, 1\}$, we have $C(f) = O(C_\oplus(f \diamond \text{OR})^3)$.*

This theorem leads us to the following natural question: can we improve this bound from cubic to quadratic, so the bound matches the separation provided by Proposition 15?

5.4 AND/OR gadgets

A function $g: \{0, 1\}^m \rightarrow \{0, 1\}$ affine projects to a function $h: \{0, 1\}^p \rightarrow \{0, 1\}$ if there are affine functions $\ell_1, \dots, \ell_m: \mathbb{Z}_2^p \rightarrow \mathbb{Z}_2$ such that

$$g(\ell_1(z), \dots, \ell_m(z)) = h(z).$$

A gadget $g: \{0, 1\}^m \rightarrow \{0, 1\}$ is AND/OR if it affine projects to both binary AND and binary OR. Here are some examples of AND/OR gadgets:

- $g(x, y, z) := x \vee (y \wedge z)$. The projections: $g(0, a, b) = a \wedge b$ and $g(a, b, 1) = a \vee b$.
- $g(x, y, z) := [x + y + z = 1]$. The projections: $g(1, \bar{a}, \bar{b}) = a \wedge b$ and $g(\bar{a}, \bar{b}, a \oplus \bar{b}) = a \vee b$.

► **Theorem 17.** *If g is an AND/OR gadget then for all functions $f: \{0, 1\}^n \rightarrow \{0, 1\}$*

$$\log \text{DTSize}_\oplus(f \diamond g) \geq C(f).$$

5.5 Gadget classification

We use the following lemma to classify gadgets which are not AND/OR.

► **Lemma 18.** *If g is not an AND/OR then g is a disjunction or a conjunction of affine forms in the inputs.*

Using these facts we prove the following classification of gadgets.

► **Theorem 19.** *For every gadget g , one of the following cases holds:*

1. *There is an infinite family of functions f_n with $\text{DT}(f_n) \rightarrow \infty$ and $\text{DTSize}_{\oplus}(f_n \diamond g) = O(1)$.*
2. *For every function f , $\text{DT}_{\oplus}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$. There is an infinite family of functions f_n with $\text{DT}(f_n) \rightarrow \infty$ and $\text{DTSize}(f_n \diamond g) = O(\text{DT}(f))$.*
3. *For every function f , $\log \text{DTSize}_{\oplus}(f \diamond g) = \Omega(\text{DT}(f)^{\Omega(1)})$ and $C_{\oplus}(f \diamond g) = \Omega(C(f)^{\Omega(1)})$.*

The first case of theorem corresponds to affine or constant gadgets, the second case corresponds to gadgets that affine project to either AND or OR, and the third case corresponds to AND/OR gadgets.

5.6 Lifting for OR gadget and the log-rank conjecture

In this section, we discuss a proof of the following inequality:

► **Theorem 20.** *For any function $f: \{0,1\}^n \rightarrow \{0,1\}$,*

$$\text{DT}(f) \leq O\left(\text{DT}_{\oplus}(f \diamond \text{OR})^{O(1)}\right).$$

This inequality is a corollary of Theorem 16 using the fact that DT is polynomially related to C, and DT_{\oplus} is polynomially related to C_{\oplus} . In [2], we present an alternative proof of this statement that gives as a byproduct a proof of the log-rank conjecture for a subclass of Boolean functions that can be represented as $f \diamond \text{OR} \diamond \text{XOR}$ for arbitrary function f .

In this section, we need a composition of a function and a gadget (XOR in this case) in the communication complexity context. For a function $f: \{0,1\}^n \rightarrow \{0,1\}$ we define a function $f_{\oplus}: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$, such that $f_{\oplus}(x, y) := f(x_1 \oplus y_1, \dots, x_n \oplus y_n)$, and associate it with the following communication problem: Alice and Bob are given x and y , respectively, and their goal is to compute $f(x, y)$. For a subclass of such XOR functions, we prove the following version of log-rank conjecture:

► **Theorem 21.** *For any function $f: \{0,1\}^n \rightarrow \{0,1\}$,*

$$D((f \diamond \text{OR})_{\oplus}) \leq \text{poly}(\log \text{rank}((f \diamond \text{OR})_{\oplus})).$$

The proof is due to the following chain of inequalities.

► **Lemma 22.** *For any function $f: \{0,1\}^n \rightarrow \{0,1\}$,*

$$\begin{aligned} \text{DT}(f) &\leq 2 \deg(f)^4 \leq O((\log \text{spar}(f \diamond \text{OR}))^4) = O((\log \text{rank}((f \diamond \text{OR})_{\oplus}))^4) \\ &\leq O(D((f \diamond \text{OR})_{\oplus})^4) \leq O(\text{DT}_{\oplus}(f \diamond \text{OR})^4). \end{aligned}$$

6 Block sensitivity to communication complexity

In this section, we provide a complete classification of gadgets for lifting from block sensitivity to deterministic and randomized communication complexities. Note that this classification will also give us a classification of gadgets for query-to-communication polynomial lifting since block sensitivity and query complexity are polynomially related to each other in the case of total functions. However, since the proof goes through block sensitivity, we will classify block sensitivity to communication complexity lifting.

6.1 Reductions in communication complexity

Let $f_i: \mathcal{X}_i \times \mathcal{Y}_i \rightarrow \{0, 1\}$ for $i = 1, 2$ be two-party functions. We say that f_1 reduces to f_2 , denoted $f_1 \leq f_2$, if the communication matrix of f_1 is a submatrix of the communication matrix of f_2 . Equivalently, $f_1 \leq f_2$ iff there exist one-to-one mappings π_A and π_B such that

$$f_1(x, y) = f_2(\pi_A(x), \pi_B(y)), \quad \forall (x, y) \in \mathcal{X}_1 \times \mathcal{Y}_1.$$

Zhang [22] proved the following theorem:

► **Theorem 23 (Zhang).** *If a two-party gadget $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ satisfies $\text{AND}, \text{OR} \leq g$, then for every function $f: \{0, 1\}^n \rightarrow Q$, the function $f \diamond g$ has (constant error) randomized communication complexity $\Omega(\text{bs}(f))$.*

For the classification of gadgets, we will need the following lemma that shows that if $\text{OR} \not\leq g$ then the communication matrix of g is block diagonal.

► **Lemma 24.** *If $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ satisfies that $\text{OR} \not\leq g$ then there are partitions of \mathcal{X} and \mathcal{Y} into disjoint sets*

$$\mathcal{X} = \mathcal{X}_0 \sqcup \mathcal{X}_1 \sqcup \dots \sqcup \mathcal{X}_k \quad \text{and} \quad \mathcal{Y} = \mathcal{Y}_0 \sqcup \mathcal{Y}_1 \sqcup \dots \sqcup \mathcal{Y}_k,$$

such that

- If $x \in \mathcal{X}_i, y \in \mathcal{Y}_j$, where $i \neq j$, or $i = 0$, or $j = 0$, then $g(x, y) = 0$.
- If $x \in \mathcal{X}_i, y \in \mathcal{Y}_i$, where $i \neq 0$, then $g(x, y) = 1$.

► **Definition 25.** *We say that a gadget $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ is a blow-up of a gadget $h: \mathcal{Z} \times \mathcal{W} \rightarrow \{0, 1\}$ if there are decompositions $\mathcal{X} = \bigsqcup_{z \in \mathcal{Z}} \mathcal{X}_z$ and $\mathcal{Y} = \bigsqcup_{w \in \mathcal{W}} \mathcal{Y}_w$, with $\mathcal{X}_z, \mathcal{Y}_w \neq \emptyset$, such that all $(x_z, y_w) \in \mathcal{X}_z \times \mathcal{Y}_w$ satisfy $g(x_z, y_w) = h(z, w)$.*

For example, g is a blow-up of XOR if (up to rearrangement) it has communication matrix of the following form

$$\begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & \dots & 1 \end{bmatrix}$$

If g is a blow-up of h then $f \diamond g$ and $f \diamond h$ have the same communication complexity in all models. Indeed, on the one hand, h is a restriction of g , and so a protocol for $f \diamond g$ can be used to solve $f \diamond h$; and on the other hand, by replacing $x \in \mathcal{X}_z$ by z and $y \in \mathcal{Y}_w$ by w , we can use a protocol for $f \diamond h$ to solve $f \diamond g$.

► **Corollary 26.** *If $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ satisfies both $\text{OR} \not\leq g$ and $\text{AND} \not\leq g$ then either g is constant, or it is a blow-up of XOR.*

► **Corollary 27.** *If $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ satisfies both $\text{OR} \not\leq g$ and $\text{XOR} \not\leq g$ then either g is constant, or it is a blow-up of AND. Similarly, if $\text{AND} \not\leq g$ and $\text{XOR} \not\leq g$ then either g is constant, or it is a blow-up of OR.*

6.2 Gadget classification for randomized communication complexity

In this section we state the following dichotomy result for the randomized case.

► **Theorem 28.** *For every gadget g , one of the following cases holds:*

1. *There is an infinite family of functions f_n with $\text{bs}(f_n) = \Omega(n)$ and $\text{R}(f_n \diamond g) = O(\log n)$.*
2. *For every function f , we have $\text{R}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$.*

The second case in the theorem corresponds to AND/OR gadgets, it holds due to Theorem 23. The first case is due to the following lemma.

► **Lemma 29.** *Let $g, h: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ be gadgets such that $\text{OR} \not\leq g$ and $\text{AND} \not\leq h$. Then*

$$\text{R}(\text{AND}_n \diamond g) = O(\log n), \quad \text{R}(\text{OR}_n \diamond h) = O(\log n).$$

6.3 Gadget classification for deterministic communication complexity

The following lifting theorem that implies the classification theorem.

► **Theorem 30.** *If a two-party gadget $g: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ satisfies $\text{AND}, \text{XOR} \leq g$, then for every function $f: \{0, 1\}^n \rightarrow \{0, 1\}$, the function $f \diamond g$ has communication complexity $\Omega(\text{bs}(f)^k)$ for some fixed constant $k > 0$. The same is true if g satisfies $\text{OR}, \text{XOR} \leq g$.*

Together with Theorem 23 this gives us a complete classification of gadgets.

► **Theorem 31.** *For every gadget g , one of the following cases holds:*

1. *There is an infinite family of functions f_n with $\text{bs}(f_n) = \Omega(n)$ and $\text{D}(f_n \diamond g) = O(1)$.*
2. *For every function f , we have $\text{D}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$.*

If $\text{AND}, \text{OR} \leq g$, or $\text{AND}, \text{XOR} \leq g$, or $\text{OR}, \text{XOR} \leq g$, then Theorem 23 and Theorem 30 show that $\text{D}(f \diamond g) = \Omega(\text{bs}(f)^{\Omega(1)})$, so the second case holds.

If none of these cases holds, then at least two of the functions AND, OR, XOR do not reduce to g . Corollaries 26 and 27 show that either g is constant (and so the first case trivially holds) or it is a blow-up of one of the functions XOR, OR, AND. By taking $f_n = \text{XOR}_n, \text{OR}_n, \text{AND}_n$ (respectively), we get the first case of the theorem.

7 Open problems

Matching lower and upper bounds for the certificate complexity lifting

Theorem 14 shows that one there is a function f such that $\text{C}(f) \geq \Omega(\text{C}_{\oplus}(f \diamond \text{OR})^2)$. However, Theorem 16 shows only that $\text{C}(f) \leq O(\text{C}_{\oplus}(f \diamond \text{OR})^3)$. Can we show that $\text{C}(f) \leq O(\text{C}_{\oplus}(f \diamond \text{OR})^2)$?

Generalization of current results to relations

Almost all the results discussed above were proved for Boolean functions. However, the question of proving lifting dichotomies for relations is still open.

One motivation for studying relations instead of functions is the following: any tree-like Resolution refutation of a CNF formula corresponds to a *decision tree*, solving the *falsified clause problem* for this CNF (which is usually a relation rather than Boolean function). Similarly, any tree-like $\text{Res}(\oplus)$ refutation of some CNF formula corresponds to a *parity decision tree*, solving the falsified clause problem for this CNF. So, any lifting theorem from decision trees to parity decision trees with constant size gadgets that holds for relations, should give us a new way of proving tree-like $\text{Res}(\oplus)$ lower bounds.

Conjunction DAG to parity conjunction DAG lifting

A parity conjunction DAG is defined similarly to a conjunction DAG, with the following two differences:

- Queries are linear forms rather than variables.
- Nodes are annotated by affine subspaces rather than partial assignments. The label $\rho(v_b)$ of a child node v_b , that is attached to the parent node v via an edge labeled b , is a subspace of $\rho(v) \cup \{\ell \leftarrow b\}$, where ℓ is the query in v .

Another possible direction of research is to prove a lifting theorem from conjunction DAG size to parity conjunction DAG size. The motivation for this kind of lifting also comes from the proof complexity. Dag-like Resolution refutation can be viewed as a conjunction DAG and dag-like $\text{Res}(\oplus)$ refutation can be viewed as a parity conjunction DAG. So, proving such lifting theorems for the relations with small enough gadgets can be used to prove lower bounds for $\text{Res}(\oplus)$ refutations, which is a long-standing open problem in proof complexity.

References


- 1 Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *Theory Comput.*, 3:81–102, 2007. doi:10.4086/toc.2007.v003a005.
- 2 Yaroslav Alekseev, Yuval Filmus, and Alexander Smal. Lifting dichotomies. *Electron. Colloquium Comput. Complex.*, pages TR24–037, 2024. URL: <https://eccc.weizmann.ac.il/report/2024/037>.
- 3 Paul Beame and Sajin Korothe. On disperser/lifting properties of the index and inner-product functions. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPICs*, pages 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ITCS.2023.14.
- 4 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, March 2001. doi:10.1145/375827.375835.
- 5 Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudo-random properties. *Comput. Complex.*, 28(4):617–659, December 2019. doi:10.1007/s00037-019-00190-7.
- 6 Arkadev Chattopadhyay, Nikhil S. Mande, Swagato Sanyal, and Suhail Sherif. Lifting to parity decision trees via stifling, 2022. doi:10.48550/arXiv.2211.17214.
- 7 Susanna de Rezende, Or Meir, Jakob Nordstrom, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 24–30, November 2020. doi:10.1109/FOCS46700.2020.00011.

- 8 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from Resolution. *Theory Comput.*, 16:Paper No. 13, 30, 2020. doi:10.4086/toc.2020.v016a013.
- 9 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:10.1137/15M103145X.
- 10 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018. doi:10.1137/16M1082007.
- 11 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM Journal on Computing*, 47(6):2435–2450, 2018. doi:10.1137/16M1059369.
- 12 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 132–143, 2017. doi:10.1109/FOCS.2017.21.
- 13 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. *SIAM J. Comput.*, 47(1):208–217, 2018. doi:10.1137/17M1136869.
- 14 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity [extended abstract]. In *STOC'12 – Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 233–247. ACM, New York, 2012. doi:10.1145/2213977.2214000.
- 15 Alexander Knop, Shachar Lovett, Sam McGuire, and Weiqiang Yuan. Log-rank and lifting for AND-functions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, pages 197–208, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3450999.
- 16 Shachar Lovett. Communication is bounded by root of rank. *J. ACM*, 63(1), February 2016. doi:10.1145/2724704.
- 17 Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with Sunflowers. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 104:1–104:24, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2022.104.
- 18 László Lovász and Michael Saks. Lattices, mobius functions and communications complexity. In *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science*, pages 81–90, 1988. doi:10.1109/SFCS.1988.21924.
- 19 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19, September 1999. doi:10.1007/s004930050062.
- 20 Alexander A. Sherstov. On quantum-classical equivalence for composed communication problems. *Quantum Inf. Comput.*, 10(5-6):435–455, 2010.
- 21 Alasdair Urquhart. The depth of resolution proofs. *Studia Logica: An International Journal for Symbolic Logic*, 99(1/3):349–364, 2011. URL: <http://www.jstor.org/stable/41475208>.
- 22 Shengyu Zhang. On the tightness of the Buhrman–Cleve–Wigderson simulation. In *Proceedings of the 20th International Symposium on Algorithms and Computation*, ISAAC '09, pages 434–440, Berlin, Heidelberg, 2009. Springer-Verlag. doi:10.1007/978-3-642-10631-6_45.
- 23 Zhiqiang Zhang and Yaoyun Shi. On the parity complexity measures of Boolean functions. *Theoretical Computer Science*, 411(26):2612–2618, 2010. doi:10.1016/j.tcs.2010.03.027.

Explicit Directional Affine Extractors and Improved Hardness for Linear Branching Programs

Xin Li  

Johns Hopkins University, Baltimore, MD, USA

Yan Zhong  

Johns Hopkins University, Baltimore, MD, USA

Abstract

Affine extractors give some of the best-known lower bounds for various computational models, such as AC^0 circuits, parity decision trees, and general Boolean circuits. However, they are not known to give strong lower bounds for read-once branching programs (ROBPs). In a recent work, Gryaznov, Pudlák, and Talebanfard (CCC' 22) introduced a stronger version of affine extractors known as directional affine extractors, together with a generalization of ROBPs where each node can make linear queries, and showed that the former implies strong lower bound for a certain type of the latter known as strongly read-once linear branching programs (SROLBPs). Their main result gives explicit constructions of directional affine extractors for entropy $k > 2n/3$, which implies average-case complexity $2^{n/3-o(n)}$ against SROLBPs with exponentially small correlation. A follow-up work by Chattopadhyay and Liao (CCC' 23) improves the hardness to $2^{n-o(n)}$ at the price of increasing the correlation to polynomially large, via a new connection to sumset extractors introduced by Chattopadhyay and Li (STOC' 16) and explicit constructions of such extractors by Chattopadhyay and Liao (STOC' 22). Both works left open the questions of better constructions of directional affine extractors and improved average-case complexity against SROLBPs in the regime of small correlation.

This paper provides a much more in-depth study of directional affine extractors, SROLBPs, and ROBPs. Our main results include:

- An explicit construction of directional affine extractors with $k = o(n)$ and exponentially small error, which gives average-case complexity $2^{n-o(n)}$ against SROLBPs with exponentially small correlation, thus answering the two open questions raised in previous works.
- An explicit function in AC^0 that gives average-case complexity $2^{(1-\delta)n}$ against ROBPs with negligible correlation, for any constant $\delta > 0$. Previously, no such average-case hardness is known, and the best size lower bound for any function in AC^0 against ROBPs is $2^{\Omega(n)}$.

One of the key ingredients in our constructions is a new linear somewhere condenser for affine sources, which is based on dimension expanders. The condenser also leads to an unconditional improvement of the entropy requirement of explicit affine extractors with negligible error. We further show that the condenser also works for general weak random sources, under the Polynomial Freiman-Ruzsa Theorem in F_2^n , recently proved by Gowers, Green, Manners, and Tao (arXiv' 23).

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors; Theory of computation → Circuit complexity; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Randomness Extractors, Affine, Read-once Linear Branching Programs, Low-degree polynomials, AC^0 circuits

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.10

Related Version *Full Version:* <https://eccc.weizmann.ac.il/report/2023/058/>

Funding *Xin Li:* Supported by NSF CAREER Award CCF-1845349 and NSF Award CCF-2127575.

Yan Zhong: Supported by NSF CAREER Award CCF-1845349.

Acknowledgements We thank anonymous reviewers for their helpful comments and a reviewer for pointing us to [20].



© Xin Li and Yan Zhong;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).
Editor: Rahul Santhanam; Article No. 10; pp. 10:1–10:14



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



1 Introduction

Randomness extractors are functions that extract almost uniform random bits from weak random sources that have poor quality. Although the original motivation of randomness extractors comes from bridging the gap between the quality of randomness required in typical applications and that available in practice, as pseudorandom objects, they turn out to have broad applications in computer science. For example, the kind of extractors known as *affine extractors* are shown to be closely connected to complexity theory. Indeed, they give strong size lower bounds for AC^0 circuits (constant depth circuits with NOT gates and unbounded fan-in AND, OR gates) by the standard switching lemma [23], and are shown to give exponential size lower bounds for DNF circuits with a bottom layer of parity gates, together with strong average-case hardness for parity decision trees [14]. Via sophisticated gate elimination techniques, they also give the best-known size lower bounds for general Boolean circuits [16, 18, 28]. We define affine extractors below.

► **Definition 1** (Affine extractor). *An (n, k) affine source is the uniform distribution over some affine subspace with dimension k , of the vector space F_2^n .¹ A function $\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ is an affine extractor for entropy k with error ε if for every (n, k) affine source X , we have*

$$\text{Ext}(X) \approx_\varepsilon U_m,$$

where U_m stands for the uniform distribution over $\{0, 1\}^m$, and \approx_ε means ε close in statistical distance. We say Ext is *explicit* if it is computable by a polynomial-time algorithm.

However, affine extractors are not known to imply strong lower bounds for computational models that measure space complexity. For example, a natural model in this context is a branching program, which is a directed acyclic graph with one source and two sinks, and each non-sink node has out-degree 2. To define the computation of the branching program, one marks each non-sink node with the index of an input bit, and labels the two outgoing edges by 0 and 1, respectively. Furthermore, one sink is labeled by 1 and the other is labeled by 0. The program now computes any input by following the natural path from the source to one sink, while reading the corresponding input bits and going through the corresponding edges. The program accepts the input if and only if the path ends in the sink with label 1, and the size of the branching program is defined as the number of its nodes, which roughly corresponds to $2^{O(s)}$ where s is the space complexity of the computation.

Proving non-trivial lower bounds of an explicit function for general branching programs turns out to be a challenging problem. The best known bound is $\Omega(\frac{n^2}{\log^2 n})$ [33] after decades of effort, which is not enough to separate P from LOGSPACE. Thus, most research on lower bounds for branching programs has focused on restricted models, and the most well-studied is the model of *read-once branching program*, where on any computational path, any input bit is read at most once. Exponential lower bounds are known in this model [41, 43, 17, 24, 27, 39, 36, 19, 5, 1, 26], however, it is not clear if affine extractors imply strong lower bounds here. For example, the inner product is a good affine extractor for any entropy $k > n/2$, but it can be computed by a read-once branching program of size $O(n)$.

In a recent work [22], Gryaznov, Pudlák, and Talebanfard introduced a generalization of affine extractors called *directional affine extractors* and a generalization of standard read-once branching programs called *read-once linear branching programs*, and show that explicit constructions of the former imply strong lower bounds for certain cases of the latter. We define the two generalizations below.

¹ More generally, affine sources and affine extractors can be defined over any finite field, but in this paper we focus on the binary field F_2 .

► **Definition 2** (Directional affine extractor). *A function $\text{DAExt} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ is a directional affine extractor for entropy k with error ε if for every (n, k) affine source X and every non-zero vector $a \in \mathbb{F}_2^n$, we have*

$$(\text{DAExt}(X), \text{DAExt}(X + a)) \approx_\varepsilon (U_m, \text{DAExt}(X + a)).$$

We say the function is a (zero-error) directional affine disperser if there exists some $b \in \{0, 1\}^m$ such that

$$\left| \text{Supp}(\text{DAExt}(X) \mid \text{DAExt}(X + a) = b) \right| = 2^m.$$

► **Remark 3.** Our definition is slightly more general than the definition in [22], since we allow the extractor to output more than one bits. In the special case of $m = 1$, our definition implies that in [22], the reverse is also true up to a small loss in parameters as shown in [12].

► **Definition 4** (Linear branching program [22]). *A linear branching program on \mathbb{F}_2^n is a directed acyclic graph P with the following properties:*

- *There is only one source s in P .*
- *There are two sinks in P , labeled with 0 and 1 respectively.*
- *Every non-sink node v is labeled with a linear function $\ell_v : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$. Moreover, there are exactly two outgoing edges from v , one is labeled with 1 and the other is labeled with 0.*

The size of P is the number of non-sink nodes in P . P computes a Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ in the following way. For every input $x \in \mathbb{F}_2^n$, P follows the computation path by starting from s , and when on a non-sink node v , moves to the next node following the edge with label $\ell_v(x) \in \{0, 1\}$. The computation ends when the path ends at a sink, and $f(x)$ is defined to be the label on this sink.

[22] defines two kinds of read-once linear branching programs (ROLBP for short). Specifically, given any linear branching program P and any node v in P , let Pre_v denote the span of all linear queries that appear on any path from the source to v , excluding the query ℓ_v . Let Post_v denote the span of all linear queries in the subprogram starting at v .

► **Definition 5** (Weakly read-once linear branching program). *A linear branching program P is weakly read-once if for every inner node v of P , it holds that $\ell_v \notin \text{Pre}_v$.*

► **Definition 6** (Strongly read-once linear branching program). *A linear branching program P is strongly read-once if for every inner node v of P , it holds that $\text{Pre}_v \cap \text{Post}_v = \{0\}$.*

In this paper, we will focus on strongly read-once linear branching programs, and use SROLBP as a shorthand. As observed in [22] and [12], even the more restricted SROLBPs generalize several important and well-studied computational models, for example, decision trees, parity decision trees, and standard read-once branching programs. These models have applications in diverse areas, such as learning theory, streaming algorithms, communication complexity and query complexity. Thus, just as the natural generalizations from AC^0 circuits to $\text{AC}^0[\oplus]$ circuits (AC^0 with parity gates), and from decision trees to parity decision trees, studying the generalization from ROBPs to ROLBPs is also a natural direction. In addition, as observed in [22], parity decision trees are the only case in $\text{AC}^0[\oplus]$ for which we have strong average-case lower bounds, and they are closely related to tree-like resolution refutation proof systems. Thus studying ROLBPs as a generalization of parity decision trees is of particular interest (in fact, this is the original motivation in [22]). We now define two complexity measures of SROLBPs below.

► **Definition 7.** For a Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, let $\text{SROLBP}(f)$ denote the smallest possible size of a strongly read-once linear branching program that computes f , and $\text{SROLBP}_\varepsilon(f)$ denote the smallest possible size of a strongly read-once linear branching program P such that

$$\Pr_{x \leftarrow_U \mathbb{F}_2^n} [P(x) = f(X)] \geq \frac{1}{2} + \varepsilon.$$

The definition can be adapted to ROBPs naturally.

The main contribution of [22] is to show that directional affine extractors give strong average-case hardness for SROLBPs. Specifically, they show that for any directional affine extractor DAExt for entropy k with error ε , we have $\text{SROLBP}_{\sqrt{\varepsilon/2}}(\text{DAExt}) \geq \varepsilon 2^{n-k-1}$. In addition, they give an explicit construction of directional affine extractor for $k \geq \frac{2n}{3} + c$ with $\varepsilon \leq 2^{-c}$, which also implies exponential average-case hardness for SROLBPs of size up to $2^{\frac{2}{3}n - o(n)}$. Thus, directional affine extractors are indeed stronger than standard affine extractors and give strong lower bounds in more computational models. [22] left open the question of explicit constructions of directional affine extractors for $k = o(n)$.

In a follow-up work, Chattopadhyay and Liao [12] showed that another kind of extractors, known as *sumset extractors*, also give strong average-case hardness for SROLBPs. These extractors were introduced by Chattopadhyay and Li [9], which are extractors that work for the sum of two (or more) independent weak random sources. By using existing constructions of such extractors in [11], they give an explicit function Ext such that $\text{SROLBP}_{n - \Omega(1)}(\text{Ext}) \geq 2^{n - \log^{O(1)} n}$, i.e., the branching program size lower bound becomes close to optimal, but the correlation increases from exponentially small to polynomially large. Similarly, [12] left open the question of obtaining improved average-case hardness against SROLBPs in the small correlation regime.

We remark that directional affine extractors are a special case of *affine non-malleable extractors*, which are defined by Chattopadhyay and Li [10]. Roughly, an affine non-malleable extractor is an affine extractor such that the output is still close to uniform, even conditioned on the output of the extractor where the input affine source is modified by any affine function with no fixed points.

In this context, directional affine extractors just correspond to the case where the tampering function adds a non-zero affine shift to the source. Previously, the best affine non-malleable extractor due to Li [32] works for entropy $k \geq (1 - \gamma)n$ for some small constant $\gamma < 1/3$ with error $2^{-\Omega(n)}$. Thus this does not give a better construction of directional affine extractors. However, [32] does give an improved sumset extractor, which yields an explicit function Ext such that $\text{SROLBP}_\varepsilon(\text{Ext}) \geq 2^{n - O(\log n)}$ for any constant $\varepsilon > 0$, i.e., the branching program size lower bound becomes optimal up to the constant in $O(\cdot)$, but the correlation increases to any constant.

1.1 Our Results

In this paper, we present a much more in-depth study of directional affine extractors, affine non-malleable extractors, SROLBPs, and standard ROBPs. To begin with, we observe that it is not a priori clear that SROLBPs are more powerful than standard ROBPs. Indeed, it is easy to see that $\text{AC}^0[\oplus]$ and parity decision trees are exponentially more powerful than AC^0 circuits and standard decision trees, respectively, since parity requires exponential size AC^0 circuits and decision trees. However, any parity function can be computed by an RBP of size $O(n)$. Nevertheless, there are previous works [34, 25, 20] which showed that computing

explicit characteristic functions of certain affine subspaces require ROBPs of size $2^{\Omega(n)}$ (e.g., the satisfiable Tseitin formulas in [20]). Since such functions are easily computable by an SROLBP of size $O(n)$, this provides a separation between SROLBP and RBP and shows that indeed SROLBPs are exponentially more powerful than ROBPs.

In turn, this further demonstrates that directional affine extractors have stronger properties than standard affine extractors, as they imply strong lower bounds for SROLBPs. Next, we give explicit constructions of directional affine extractors with much better parameters than that in [22]. Our construction works for any linear entropy with exponentially small error.

► **Theorem 8.** *For any constant $0 < \delta \leq 1$, there exists a family of explicit directional affine extractors $\text{DAExt} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ for entropy $k \geq \delta n$ with error $\varepsilon = 2^{-\Omega(n)}$ and output length $m = \Omega(n)$.*

In fact, our construction can work for slightly sub-linear entropy.

► **Theorem 9.** *There exists a constant $c > 1$ and an explicit family of directional affine extractors $\text{DAExt} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ for entropy $k \geq cn(\log \log \log n)^2 / \log \log n$ with error $\varepsilon = 2^{-n^{\Omega(1)}}$ and output length $m = n^{\Omega(1)}$, as well as an explicit family of directional affine dispersers for entropy $k \geq cn(\log \log n)^2 / \log n$ with $m = n^{\Omega(1)}$.*

This theorem immediately gives much improved average-case hardness for SROLBPs.

► **Theorem 10.** *There is an explicit function DAExt such that $\text{SROLBP}_{2^{-n, \Omega(1)}}(\text{DAExt}) \geq 2^{n - \tilde{O}(\frac{n}{\log \log n})}$, where $\tilde{O}(\cdot)$ hides $(\log \log \log n)^2$ factors.*

In particular, we can achieve exponentially small correlation while obtaining a $2^{n-o(n)}$ size lower bound for SROLBPs, which is almost optimal. This significantly improves the $2^{n/3-o(n)}$ size lower bound in [22] and the polynomially large correlation in [12]. Thus, Theorem 9 and 10 provide positive answers to the two open questions in [22] and [12] mentioned before.

We remark that under our new definition, a directional affine extractor is strictly stronger than a standard affine extractor. Thus Theorem 9 also improves the entropy requirement of negligible error affine extractors, from the previously best-known result of $\frac{n}{\sqrt{\log \log n}}$ [42, 29] to $\frac{cn(\log \log \log n)^2}{\log \log n}$.

We also revisit the hardness results for standard ROBPs. As mentioned before, exponential and even close to optimal size lower bounds are known for explicit functions in this model, where the current best result is an explicit function that requires ROBPs (in fact, SROLBPs) of size $2^{n-O(\log n)}$ [32]. However, there has also been a lot of interest in finding functions in lower complexity classes that give strong lower bounds for ROBPs. It is clear that the class NC^0 is not sufficient. Thus the next possible class is AC^0 . Indeed there are previous works giving explicit AC^0 functions that require ROBPs of size $2^{\Omega(\sqrt{n})}$ [24, 27, 19, 5] and even $2^{\Omega(n)}$ [20], yet there is no average-case hardness as far as we know. Here, we improve both the size lower bound and the average-case hardness by giving an explicit AC^0 function that has negligible correlation with ROBPs of size $2^{(1-\delta)n}$ for any constant $\delta > 0$.

► **Theorem 11.** *For any constant $\delta > 0$ there is an explicit function $\text{AC}^0\text{-Ext}$ in AC^0 such that $\text{ROBP}_{2^{-\text{poly} \log n}}(\text{AC}^0\text{-Ext}) \geq 2^{(1-\delta)n}$.*

One of the key ingredients in our constructions is a new linear somewhere condenser for affine sources. Specifically, we have

► **Definition 12.** *For any $0 < \delta < \gamma < 1$, a function $\text{SCond} : \mathbb{F}_2^n \rightarrow (\mathbb{F}_2^m)^\ell$ is a (δ, γ) affine somewhere condenser, if it satisfies the following property: for any affine source X over \mathbb{F}_2^n with entropy δn , let $(Y_1, \dots, Y_\ell) = \text{SCond}(X) \in (\mathbb{F}_2^m)^\ell$, then there exists at least one $i \in [\ell]$ such that Y_i is an affine source over \mathbb{F}_2^m with entropy at least γm .*

► **Theorem 13.** *There exists a constant $\beta > 0$ such that for any $0 < \delta \leq 1/2$, there is an explicit $(\delta, 1/2 + \beta)$ affine somewhere condenser $\text{SCond} : \mathbb{F}_2^n \rightarrow (\mathbb{F}_2^m)^t$, where $t = \text{poly}(1/\delta)$ and $m = n/\text{poly}(1/\delta)$. Moreover, SCond is a linear function.*

We further show that (a slight modification of) this condenser works for general weak random sources, under the well-known Polynomial Freiman-Ruzsa Theorem in \mathbb{F}_2^n , once one of the most important conjectures in additive combinatorics and very recently proved by Gowers, Green, Manners, and Tao [21].

Previously, all condensers of this kind are based on sum-product theorems, and the function is a polynomial with degree $\text{poly}(1/\delta)$ [3, 38, 44]. In contrast, there exist constructions of linear *seeded* extractors, where if one lists the outputs of the extractor for all possible seeds, then we get a somewhere random source such that at least one output is close to uniform, and the function is a linear function. However, in many applications such as ours, one needs to use a somewhere condenser instead of simply listing all outputs of an extractor, since the former only gives a small number (e.g., a constant) of outputs as opposed to $\text{poly}(n)$ outputs from the extractor. Hence, our linear somewhere condenser complements the existing sum-product theorem based somewhere condensers. Moreover, our construction of the condenser is based on *dimension expanders*, which are algebraic pseudorandom objects previously studied based on their own interests, with no clear applications in computer science as far as we know. Thus, our construction can be viewed as one of the first applications of dimension expanders in computer science.

Finally, we study the question of whether directional affine extractors can give strong lower bounds for the class of $\text{AC}^0[\oplus]$ in a black box way. Cohen and Tal [15] showed via probabilistic methods that standard affine extractors do not suffice since depth-3 $\text{AC}^0[\oplus]$ circuits can compute optimal affine extractors. Using a slightly modified argument as that in [15], we show that even the stronger version of directional affine extractors does not suffice. Specifically, depth-3 $\text{AC}^0[\oplus]$ circuits can also compute optimal directional affine extractors. This in turn provides a strong separation of $\text{AC}^0[\oplus]$ from SROLBP .

► **Theorem 14.** *There exists a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ which is a directional affine extractor for entropy k with error ε , where $k = \log \frac{n}{\varepsilon^2} + \log \log \frac{n}{\varepsilon^2} + O(1)$ such that the following properties hold.*

1. f is a polynomial of degree $\log \frac{n}{\varepsilon^2} + \log \log \frac{n}{\varepsilon^2} + O(1)$.
2. f can be realized by a XOR-AND-XOR circuit of size $O((n/\varepsilon)^2 \cdot \log^3(n/\varepsilon))$.
3. f can be realized by a De Morgan formula of size $O((n^5/\varepsilon^2) \cdot \log^3(n/\varepsilon))$.

2 Overview of the Techniques

Here we give a sketch of the main ideas used in this paper. For clarity, we shall be informal at places and ignore some technical details.

2.1 Directional affine extractors

Our starting point is the construction of affine extractors by Li [29], which works for sub-linear entropy with exponentially small error. We first briefly recall the construction there. Divide an affine source X of entropy rate δ into $O(1/\delta)$ blocks. By choosing the size of the blocks appropriately, one can show that there exists a “good” block X_g of entropy rate $\Omega(\delta)$, and the source X still has a lot of entropy conditioned on X_g (i.e., we get an affine *block source*). If we know the position of X_g , randomness extraction is easy: we apply a somewhere condenser (e.g., those in [3, 38, 44]) to condense X_g into a matrix with a constant number of rows,

such that at least one row has entropy rate $1 - \delta/2$. At this point, we can apply a linear two-source extractor (e.g., the inner product function) to each row of the matrix and the source X to get an affine *somewhere random* source, conditioned on the fixing of X_g . This is another matrix with a constant number of rows, such that at least one row is uniform, and one can apply existing techniques to deterministically extract random bits from this source [37].

However, when δ is small, we don't know which block X_g is good. Thus in [29], the construction tries all blocks, and then combines them together. To make this process work, the construction crucially maintains the following property: (*) for each block X_i , the output bits produced from this block are constant degree polynomials of the input bits, and the degrees decrease geometrically from the first block to the last block. With this property, the analysis goes by focusing on the first good block X_g . Notice that we can fix all the outputs produced from blocks before X_g , while all outputs produced from blocks after X_g have degrees less than those from X_g . Thus if we take the XOR of all these outputs, an XOR lemma of polynomials [40, 4] guarantees the final output is still close to uniform. We note that the XOR lemma of polynomials only works for degree up to $\log n$. Hence it is important to keep the degree c of the outputs from each block to be as small as possible. Roughly, we will need $c^{O(1/\delta)} < \log n$.

Our strategy now is to adapt this construction to directional affine extractors. Towards this, we use techniques from constructions of non-malleable extractors since, as we remark before, directional affine extractors are a special case of affine non-malleable extractors. Recent constructions of non-malleable extractors usually consist of two steps: first, generate a small advice that is different from the tampered version with high probability, and then use the advice together with other tools (e.g., correlation breakers) to achieve non-malleability. Thus, our goal is to adapt these two steps to directional affine extractors while, at the same time, still maintaining property (*), which is crucial to achieving any linear entropy or slightly sub-linear entropy. We now explain both steps.

As before, for each block X_i we will get an output U_i , which is close to uniform if X_i is a good block. Divide U_i into two parts $U_i = U_{i1} \circ U_{i2}$. We will use U_{i1} to generate the advice and U_{i2} for the rest of the construction. Notice that from the tampered input $X' = X + a$ we also have a tampered version $U'_i = U'_{i1} \circ U'_{i2}$. In the following, we will always use letters with prime to denote the corresponding random variables produced from the tampered input. If $U_{i1} \neq U'_{i1}$ then we are done, otherwise we use $U_{i1} = U'_{i1}$ to sample some $\Omega(\delta^2 n)$ bits H_i from an encoding of X , using an asymptotically good binary linear code. Since $X' = X + a$, we have that $H_i + H'_i$ basically corresponds to the sampled bits from the encoding of a . Thus $H_i \neq H'_i$ with high probability by the distance of the linear code. However, we cannot just do sampling naively since we need to keep the degree to be a constant. Therefore, we also divide both U_{i1} and the encoding of X into $\Omega(\delta^2 n)$ blocks where each block contains a constant number of bits, and use each block of U_{i1} to sample one bit from the corresponding block of the encoding of X . By the distance property of the code, there are $\Omega(\delta^2 n)$ blocks of the encoding of X and X' that are different. Thus we still have $H_i \neq H'_i$ with high probability, and now each bit of H_i is a constant degree polynomial of the bits of U_{i1} and X . The advice string is now $U_{i1} \circ H_i$.

Once we have the advice, we can append it to another string extracted from X by using a linear seeded extractor and U_{i2} as the seed. Now notice that the string produced from X is different from the string produced from X' with high probability, and they are linearly correlated conditioned on the fixing of (U_i, U'_i) . Thus we can apply, for example, a known affine non-malleable extractor (the state-of-the-art affine non-malleable extractor

with negligible error only works for high entropy). However, the known construction of affine non-malleable extractor in [10] has super constant degree. Indeed, even one application of this extractor results in a polynomial of degree larger than $\log n$, which already defeats our purpose to get a directional affine extractor (we can still get a directional affine disperser, though).

To solve this problem, we develop new ideas that make use of the special structure of $X' = X + a$. Recall that in our construction, for every block X_i we get a U_{i2} , which is close to uniform if X_i is good, and X still has enough entropy conditioned on X_i . Our idea now is to use a *seeded non-malleable extractor* snmExt instead, which is an extractor with a uniform random seed, such that if an adversary tampers with the seed but not the source, then the output of the extractor on the original inputs is close to uniform given the output on the tampered inputs. By appending the advice string to U_{i2} and getting $\tilde{U}_i = U_i \circ H_i$, we have $\tilde{U}_i \neq \tilde{U}'_i$ with high probability, and the seed \tilde{U}_i has high entropy if H_i has small size, which suffices for the seeded non-malleable extractor as long as the extractor is strong. Now, if the seeded non-malleable extractor is also *linear* conditioned on any fixing of the seed, then we have $\text{snmExt}(X', \tilde{U}'_i) = \text{snmExt}(X, \tilde{U}'_i) + \text{snmExt}(a, \tilde{U}'_i)$. Since $\text{snmExt}(X, \tilde{U}_i)$ is close to uniform given $\text{snmExt}(X, \tilde{U}'_i)$ (because it is a non-malleable extractor), and the extractor is strong (we can fix the seeds $(\tilde{U}_i, \tilde{U}'_i)$), this implies that $\text{snmExt}(X, \tilde{U}_i)$ is close to uniform given $\text{snmExt}(X', \tilde{U}'_i)$.²

Luckily, there are previous constructions of linear seeded non-malleable extractors due to Li [30], which are based on the inner product function. Moreover, this extractor also has the property that each output bit is a constant degree polynomial of the input bits. Thus everything seems to work out, except for one problem: the non-malleable extractor in [30] only works when the source has entropy rate $> 1/2$, but here our goal is to work for any linear (or slightly sub-linear) entropy. A natural idea would be to use the somewhere condenser (e.g., in [3, 38, 44]) to boost the entropy rate of X . However, all known condensers of this kind are based on sum-product theorems, which are non-linear functions, and applying them changes the structure of $X' = X + a$, which is important for our construction. Another idea is to apply a linear seeded extractor to X and try all possible seeds. This indeed keeps the structure of $X' = X + a$, but will result in a $\text{poly}(n)$ number of outputs, and combining them together will result in a polynomial of large, super constant degree.

This motivates another key ingredient in our construction, a new linear somewhere condenser for affine sources. In short, we construct a linear function which, given any affine source on n bits with entropy rate $0 < \delta \leq 1/2$, outputs $\text{poly}(1/\delta)$ rows such that each row has $n/\text{poly}(1/\delta)$ bits, and at least one row has entropy rate $1/2 + \beta$ for some absolute constant $\beta > 0$. This complements the sum-product based somewhere condensers, and can be viewed as a separate contribution of our work. We will explain the construction of this condenser later, but finish the description of our directional affine extractor here, assuming that we have the linear somewhere condenser.

The rest of the construction roughly goes as follows. We apply the linear somewhere condenser to the source X to get a constant number of rows, then apply snmExt to each row using \tilde{U}_i as the seed. Thus we get a constant number of outputs such that at least one of them is close to uniform conditioned on the corresponding tampered output. Now we apply an *affine correlation breaker* such as those in [31, 8, 11] to further break the correlations between different outputs, and combine these outputs together by taking the XOR. The

² The actual analysis involves more details since here X is not independent of $(\tilde{U}_i, \tilde{U}'_i)$, but the property still holds due to the affine structure. We omit the details here.

correlation breaker guarantees that the final output is close to uniform conditioned on the tampered output. To keep the degree small, we need to replace all seeded extractors used in the correlation breaker with a constant degree linear seeded extractor in [29]. This keeps the output bits to be constant degree polynomials of the input bits, and the remaining construction is essentially the same as that in [29].

2.2 Linear somewhere condenser

We now describe our construction of the linear somewhere condenser. This is based on another pseudorandom object known as *dimension expander*. Informally, a dimension expander is a set of linear mappings from a vector space F^n to itself, such that for any linear subspace $V \subset F^n$ with small dimension $k \leq n/2$, the span of the union of all the images of V under the set of linear mappings has dimension at least $(1 + \alpha)k$ for some absolute constant $\alpha > 0$. Readers familiar with expander graphs can see that this is a linear algebraic analog of expander graphs. Thus, it is desirable to give explicit constructions of the set of linear mappings which has as few number of mappings as possible, where this number d is called the degree. Dimension expanders were first introduced by Barak, Impagliazzo, Shpilka, and Wigderson [2], who also showed the existence of such objects. Later, Bourgain and Yehudayoff [6, 7] gave explicit constructions of dimension expanders with degree $d = O(1)$ over any field. Interestingly, as far as we know, there are no previous applications of dimension expanders in computer science, and they are mainly studied based on their own interests and connections to other algebraic pseudorandom objects. Thus our construction can be viewed as one of the first applications of dimension expanders in computer science.

Given an explicit dimension expander $\{T_i\}_{i \in [d]}$ where each T_i is a linear mapping, and any affine source X with entropy rate $\delta \leq 1/2$, we first construct a basic somewhere condenser as follows. Divide X equally into $X = X_1 \circ X_2$, and our condenser produces $2d + 2$ outputs: $(X_1, X_2, \{X_1 + T_i(X_2)\}_{i \in [d]}, \{T_i(X_1) + X_2\}_{i \in [d]})$. We show that at least one output has entropy rate $(1 + \gamma)\delta$ for some constant $\gamma > 0$, and we give some intuition below. By the structure of affine sources, one can show that there exists another affine source X_3 independent of X_1 such that $X_2 = X_3 + L(X_1)$ for some linear function L . Let $H(X_1) = s$, $H(X_3) = r$ and $H(L(X_1)) = t$, then we have $s + r = \delta n$. If either s or r is small, e.g., $s \ll \delta n/2$, then we must have $r \gg \delta n/2$ and thus $H(X_2) = r + t \geq (1 + \gamma)\delta n/2$. Therefore the entropy rate of X_2 is at least $(1 + \gamma)\delta$. The case of $r \ll \delta n/2$ is similar. Hence, we only need to consider the case where $s \approx \delta n/2$ and $r \approx \delta n/2$, and notice that we must have either $s \leq \delta n/2$ or $r \leq \delta n/2$. Furthermore, in this case, t must be small, since otherwise, we would again have $H(X_2) = r + t \geq (1 + \gamma)\delta n/2$.

For simplicity, assume that $s = r = \delta n/2$, and $t = 0$. Hence both X_1 and X_2 have entropy rate $\delta \leq 1/2$, and they are independent. Without loss of generality, assume the supports of both X_1 and X_2 are linear subspaces. By the property of the dimension expander, $\text{Span}(\cup_{i \in [d]} T_i(X_1))$ has dimension at least $(1 + \alpha)\delta n/2$. We now argue that there exists an $i \in [d]$ such that the support of $T_i(X_1) + X_2$ has dimension at least $(1 + \alpha/d)\delta n/2$, which implies that $T_i(X_1) + X_2$ has entropy rate at least $(1 + \alpha/d)\delta$. To see this, assume otherwise, then for any $i \in [d]$, any vector in the support of $T_i(X_1) + X_2$ can be expressed as a linear combination of the $r = \delta n/2$ basis vectors in the support of X_2 and $< (\alpha/d)\delta n/2$ other vectors. This implies that $\text{Span}(\cup_{i \in [d]} T_i(X_1))$ has dimension $< \delta n/2 + d \cdot (\alpha/d)\delta n/2 = (1 + \alpha)\delta n/2$, since any vector in $\text{Span}(\cup_{i \in [d]} T_i(X_1))$ can be expressed as a linear combination of the $r = \delta n/2$ basis vectors in the support of X_2 and $< d \cdot (\alpha/d)\delta n/2$ other vectors. This contradicts the property of the dimension expander.

10:10 Directional Affine Extractors and Linear Branching Programs

Thus, in all cases, we get the desired entropy rate boost. Our final somewhere condenser involves repeated uses of the basic condenser, as in previous works. It is easy to see that the entropy rate of at least one output will increase to $1/2 + \beta$ for some absolute constant $\beta > 0$ after $O(\log(1/\delta))$ uses of the basic condenser. The number of outputs is, therefore, $\text{poly}(1/\delta)$ and each output has $n/\text{poly}(1/\delta)$ bits. Finally, it is clear that the condenser is a linear function.

Once we have this linear condenser, we can even replace the somewhere condensers used in [29] by the new condenser. This further reduces the degree of the polynomials of the output bits (since previous somewhere condensers are polynomials instead of linear functions). Therefore we can push the entropy requirement of our directional affine extractor to be even better than that in [29], from $\frac{n}{\sqrt{\log \log n}}$ to $\frac{cn(\log \log \log n)^2}{\log \log n}$.

We show that a slight modification of our linear condenser also works for general weak random sources, under the Polynomial Freiman-Ruzsa Theorem. Roughly, the idea is to use a careful analysis of subsources and collision probability. Specifically, it is known that if the collision probability of a distribution is small, then the distribution is close to having high min-entropy. On the other hand, if the collision probability is large, then (without loss of generality) assuming the distribution is the uniform distribution over some unknown subset, existing results in additive combinatorics imply that there is a large subset A in the support of the distribution such that the size of $A + A$ is not much larger than A . The Polynomial Freiman-Ruzsa Theorem then implies that there is another large subset $A' \subset A$ which is “close” to an affine subspace, which roughly reduces the analysis to the case of affine sources.

2.3 AC^0 average-case hardness for ROBPs

To show AC^0 average-case hardness for ROBPs, we use a standard observation that if one conditions on an inner node, then the input bits prior to this node and the input bits after this node are still independent. We then construct an appropriate extractor in AC^0 , which we call $\text{AC}^0\text{-Ext}$, for sources with such a structure. Specifically, given any ROBP of size s and any constant $\delta > 0$, we can find a cut or anti-chain (a maximal subset of vertices such that none of which is an ancestor of any other vertex) of size $O(s)$ at roughly depth δn above the sinks, so that conditioned on the fixing of any vertex in the cut, the input uniform random string X now becomes two independent weak sources A and B , where A corresponds to the first part of the program and B corresponds to the second part. Since we don't know the order of bits queried by the ROBP, the bits of the two sources are interleaved, and we view $X = A + B$. Using a standard averaging argument, one can show that with high probability, the following properties are satisfied: (1) A and B are supported on disjoint subsets of input bits; (2) A has min-entropy roughly $(1 - \delta)n - \log s$ and B has min-entropy δn ; and (3) B is an oblivious bit-fixing source, which is obtained by fixing some unknown bits in a uniform random string. If $s \leq 2^{(1-2\delta)n}$ then both A and B have entropy rate roughly δ . Now, our goal is to construct an extractor in AC^0 for sources with this structure, that is also *strong* in B . This means that even if we condition on the fixing of the vertex in the cut and B , the output of the extractor is still close to uniform. On the other hand, the output of the ROBP is completely determined by the vertex and B . Thus our extractor is average-case hard for ROBPs of size up to $2^{(1-2\delta)n}$.

As usual, the function $\text{AC}^0\text{-Ext}$ will be compositions of different, more basic extractors as building blocks. Thus we need all these building blocks to be computable in AC^0 . Here, we leverage the constructions from two previous works on extractors in AC^0 : (1) the AC^0 -computable extractors $\text{AC}^0\text{-BExt}$ for bit-fixing source by Cheng and Li [13], and (2) the AC^0 -computable strong linear seeded extractors $\text{AC}^0\text{-LExt}$ by Papakonstantinou, Woodruff, and Yang [35].

Now we can describe our main idea of construction. Divide X into $t = O(1/\delta)$ blocks, and by an averaging argument, there exists a block B_g of B with entropy rate $\Omega(\delta)$. Now for the block $X_g = A_g + B_g$, we can fix A_g so that X_g is an oblivious bit-fixing source of entropy rate $\Omega(\delta)$ and is a deterministic function of B . We next fix the bits from B outside of the g -th block so that the source X outside of X_g is a deterministic function of A and thus independent of X_g . Moreover, A and X still have enough entropy left.

Applying the above-mentioned extractor $\text{AC}^0\text{-BFExt}$ for bit-fixing sources to each block X_i , we convert X into a *somewhere random source* $Y = Y_1 \circ \dots \circ Y_t$ where the row Y_g is a deterministic function of B_g and close to uniform, while all the other rows are deterministic functions of A . At this point, we can simply take the XOR of the Y_i 's to obtain a close-to-uniform output. However, as mentioned before, we need the extractor to be strong in B and this simple approach is not sufficient. Instead, we fix all the outputs produced by $\text{AC}^0\text{-BFExt}$ for X_i where $i \neq g$. Note that these are all deterministic functions of A . Thus conditioned on this fixing, Y becomes a deterministic function of B , which is independent of A . Moreover, as long as the output size of $\text{AC}^0\text{-BFExt}$ is not too large, A still has enough entropy left. Since $X = A + B$, we can now apply a *strong t -affine correlation breaker* as in [31, 11] with each Y_i as the seed to extract from X a random string, and take the XOR of them. The property of the correlation breaker guarantees that the string produced from Y_g and X is close to uniform conditioned on all the other outputs and Y . Hence the XOR is also close to uniform conditioned on B . To ensure the correlation breaker is computable in AC^0 , we replace all the strong (linear) seeded extractors in the known constructions of t -affine correlation breakers with the above-mentioned $\text{AC}^0\text{-LExt}$. Since $t = O(1/\delta)$ is a constant, the correlation breaker involves a constant number of compositions of $\text{AC}^0\text{-LExt}$, which is still in AC^0 .

3 Open Problems

Our work leaves several natural open problems. The most obvious is to further improve the constructions of directional affine extractors and the average-case hardness for SROLBPs. It would also be quite interesting to show any hardness of explicit functions for WROLBPs, which appears to require new ideas. Finally, it is an interesting question to see if there exist functions in AC^0 that achieve optimal hardness for ROBPs, or strong hardness for SROLBPs.

References

- 1 Alexander E. Andreev, Juri L. Baskakov, Andrea E. F. Clementi, and José D. P. Rolim. Small pseudo-random sets yield hard functions: New tight explicit lower bounds for branching programs. In Jiri Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings*, volume 1644 of *Lecture Notes in Computer Science*, pages 179–189. Springer, 1999. doi:10.1007/3-540-48523-6_15.
- 2 Boaz Barak, Russel Impagliazzo, Amir Shpilka, and Avi Wigderson. Definition and existence of dimension expanders. Discussion (no written record), 2004.
- 3 Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2005.
- 4 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, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 488–497. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.54.

10:12 Directional Affine Extractors and Linear Branching Programs

- 5 Beate Bollig and Ingo Wegener. A very simple function that requires exponential size read-once branching programs. *Inf. Process. Lett.*, 66(2):53–57, 1998. doi:10.1016/S0020-0190(98)00042-8.
- 6 Jean Bourgain. Expanders and dimensional expansion. *Comptes Rendus Mathematique*, 347(7):357–362, 2009.
- 7 Jean Bourgain and Amir Yehudayoff. Expansion in $SL_2(\mathbb{R})$ and monotone expanders. *Geometric and Functional Analysis*, 23(1):1–41, 2013.
- 8 Eshan Chattopadhyay, Jesse Goodman, and Jyun-Jie Liao. Affine extractors for almost logarithmic entropy. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 622–633, 2022. doi:10.1109/FOCS52979.2021.00067.
- 9 Eshan Chattopadhyay and Xin Li. Extractors for sunset sources. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC, Cambridge, MA, USA, June 18-21, 2016*, pages 299–311. ACM, 2016. doi:10.1145/2897518.2897643.
- 10 Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1171–1184. ACM, 2017. doi:10.1145/3055399.3055483.
- 11 Eshan Chattopadhyay and Jyun-Jie Liao. Extractors for sum of two sources. 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 1584–1597. ACM, 2022. doi:10.1145/3519935.3519963.
- 12 Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In *Proceedings of the Conference on Proceedings of the 38th Computational Complexity Conference, CCC '23, Dagstuhl, DEU, 2023*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
- 13 Kuan Cheng and Xin Li. Randomness extraction in AC0 and with small locality. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA*, volume 116 of *LIPICs*, pages 37:1–37:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.APPROX-RANDOM.2018.37.
- 14 Gil Cohen and Igor Shinkar. The complexity of DNF of parities. In Madhu Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 47–58. ACM, 2016. doi:10.1145/2840728.2840734.
- 15 Gil Cohen and Avishay Tal. Two structural results for low degree polynomials and applications. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015*, 2015.
- 16 Evgeny Demenkov and Alexander Kulikov. An elementary proof of $3n-o(n)$ lower bound on the circuit complexity of affine dispersers. In *Proceedings of the 36th international conference on Mathematical foundations of computer science*, pages 256–265, 2011.
- 17 Paul E. Dunne. Lower bounds on the complexity of 1-time only branching programs. In Lothar Budach, editor, *Fundamentals of Computation Theory, FCT '85, Cottbus, GDR, September 9-13, 1985*, volume 199 of *Lecture Notes in Computer Science*, pages 90–99. Springer, 1985. doi:10.1007/BFb0028795.
- 18 Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than- $3n$ lower bound for the circuit complexity of an explicit function. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 89–98, 2016. doi:10.1109/FOCS.2016.19.
- 19 Anna Gál. A simple function that requires exponential size read-once branching programs. *Inf. Process. Lett.*, 62(1):13–16, 1997. doi:10.1016/S0020-0190(97)00041-0.

- 20 Ludmila Glinskikh and Dmitry Itsykson. Satisfiable Tseitin Formulas Are Hard for Non-deterministic Read-Once Branching Programs. In Kim G. Larsen, Hans L. Bodlaender, and Jean-Francois Raskin, editors, *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017)*, volume 83 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:12, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2017.26.
- 21 W. T. Gowers, Ben Green, Freddie Manners, and Terence Tao. On a conjecture of marton, 2023. arXiv:2311.05762.
- 22 Svyatoslav Gryaznov, Pavel Pudlák, and Navid Talebanfard. Linear Branching Programs and Directional Affine Extractors. In *37th Computational Complexity Conference (CCC 2022)*, volume 234, pages 4:1–4:16, 2022.
- 23 Johan Håstad. Almost optimal lower bounds for small depth circuits. In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20. ACM, 1986. doi:10.1145/12130.12132.
- 24 Stasys Jukna. Entropy of contact circuits and lower bounds on their complexity. *Theor. Comput. Sci.*, 57:113–129, 1988. doi:10.1016/0304-3975(88)90166-1.
- 25 Stasys Jukna. A note on read-k times branching programs. *RAIRO - Theoretical Informatics and Applications*, 28:75–83, January 1995.
- 26 Valentine Kabanets. Almost k-wise independence and hard boolean functions. *Theor. Comput. Sci.*, 297(1-3):281–295, 2003. doi:10.1016/S0304-3975(02)00643-6.
- 27 Matthias Krause, Christoph Meinel, and Stephan Waack. Separating the eraser turing machine classes L_e , NL_e , $co-NL_e$ and P_e . *Theor. Comput. Sci.*, 86(2):267–275, 1991. doi:10.1016/0304-3975(91)90021-S.
- 28 Jiayu Li and Tianqi Yang. $3 \ln - o(n)$ circuit lower bounds for explicit functions. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 1180–1193, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3519976.
- 29 Xin Li. A new approach to affine extractors and dispersers. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC*, 2011.
- 30 Xin Li. Non-malleable extractors, two-source extractors and privacy amplification. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, 2012.
- 31 Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, STOC 2017, pages 1144–1156, New York, NY, USA, 2017. Association for Computing Machinery.
- 32 Xin Li. Two source extractors for asymptotically optimal entropy, and (many) more. Technical report, Arxiv, 2023. arXiv:2303.06802.
- 33 E. I. Nechiporuk. On a boolean function. *Doklady of the Academy of Sciences of the USSR*, 164(4):765–766, 1966.
- 34 EA Okolniskhnikova. On lower bounds for branching programs. *Siberian Advances in Mathematics*, 3:152–156, January 1993.
- 35 Periklis A Papakonstantinou, David P Woodruff, and Guang Yang. True randomness from big data. *Scientific reports*, 6:33740, 2016.
- 36 Stephen Ponzio. A lower bound for integer multiplication with read-once branching programs. *SIAM Journal on Computing*, 28(3):798–815, 1998. doi:10.1137/S0097539795290349.
- 37 Anup Rao. Extractors for low-weight affine sources. In *Proceedings of the 2009 24th Annual IEEE Conference on Computational Complexity, CCC '09*, pages 95–101. IEEE Computer Society, 2009.
- 38 Ran Raz. Extractors with weak random seeds. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 11–20, 2005.
- 39 Janos Simon and Mario Szegedy. A new lower bound theorem for read-only-once branching programs and its applications. In *Advances In Computational Complexity Theory*, 1992.

10:14 Directional Affine Extractors and Linear Branching Programs

- 40 Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(7):137–168, 2008. doi:10.4086/toc.2008.v004a007.
- 41 Ingo Wegener. On the complexity of branching programs and decision trees for clique functions. *J. ACM*, 35(2):461–471, 1988. doi:10.1145/42282.46161.
- 42 Amir Yehudayoff. Affine extractors over prime fields. *Combinatorica*, 31(2):245–256, 2011.
- 43 Stanislav Zák. An exponential lower bound for one-time-only branching programs. In Michal Chytil and Václav Koubek, editors, *Mathematical Foundations of Computer Science 1984, Praha, Czechoslovakia, September 3-7, 1984, Proceedings*, volume 176 of *Lecture Notes in Computer Science*, pages 562–566. Springer, 1984. doi:10.1007/BFb0030340.
- 44 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Theory of Computing*, 2007.

Linear-Size Boolean Circuits for Multiselection

Justin Holmgren  

NTT Research, Sunnyvale, CA, USA

Ron Rothblum  

Technion, Haifa, Israel

Abstract

We study the circuit complexity of the *multiselection problem*: given an input string $x \in \{0, 1\}^n$ along with indices $i_1, \dots, i_q \in [n]$, output $(x_{i_1}, \dots, x_{i_q})$. A trivial lower bound for the circuit size is the input length $n + q \cdot \log(n)$, but the straightforward construction has size $\Theta(q \cdot n)$.

Our main result is an $O(n + q \cdot \log^3(n))$ -size and $O(\log(n + q))$ -depth circuit for multiselection. In particular, for any $q \leq n / \log^3(n)$ the circuit has linear size and logarithmic depth. Prior to our work no linear-size circuit for multiselection was known for *any* $q = \omega(1)$ and regardless of depth.

2012 ACM Subject Classification Theory of computation \rightarrow Design and analysis of algorithms

Keywords and phrases Private Information Retrieval, Batch Selection, Boolean Circuits

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.11

Related Version *Full Version*: <https://eccc.weizmann.ac.il/report/2023/113/>

Funding *Ron Rothblum*: Ron Rothblum is funded by the European Union (ERC, FASTPROOF, 101041208). 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. Neither the European Union nor the granting authority can be held responsible for them.

Acknowledgements We thank Yuval Ishai for useful discussions and his encouragement and an anonymous reviewer for useful comments.

1 Introduction

In the *selection problem*, also commonly called multiplexing, one is given a long string $x \in \{0, 1\}^n$ and an index $i \in [n]$, and the goal is to output x_i . Selection is one of the basic operations in the word RAM model (and indeed costs unit time in that model). Implementing it by a Boolean circuit however is more involved and requires a circuit of size $\Theta(n)$.¹

In this work we investigate a natural generalization of selection, which we call *multiselection*; now rather than a single index, one is given indices $i_1, \dots, i_q \in [n]$, and the goal is to compute $(x_{i_1}, \dots, x_{i_q})$. We denote this function by $\text{Sel}^{n \rightarrow q}$. Multiselection is trivial in the word RAM model of computation (it costs $\Theta(q)$ operations with any standard instruction set). We focus on implementing multiselection by small *Boolean circuits*, where here and throughout this work all circuits have bounded fan-in. In this setting, we do not have matching upper and lower bounds. The best circuit size lower bound for multiselection is the input length $n + q \cdot \log n$. On the other hand, the straightforward construction (which selects each index separately) is much larger, with size $\Theta(n \cdot q)$. A slightly more sophisticated approach reduces to sorting, and results in a circuit of size $\Theta(n \log^2 n)$ for any $q = O(n)$. Somewhat surprisingly, it appears that multiselection has not been systematically studied and in particular we are not aware of essentially any other upper bounds (see further discussion in Section 1.1).

¹ The $\Omega(n)$ lower bound simply comes from the input size (which is a circuit size lower bound for any function in which each input has non-zero influence). The upper bound follows from a simple divide and conquer approach (cf. [27, Lemma 2.5.5] or Proposition 2).

Our main result is an $O(n)$ -size, and $O(\log n)$ -depth, Boolean circuit for multiselection for any $q \leq O(n/\log^3(n))$. More generally: [Main Theorem] For all $n, q \in \mathbb{Z}^+$ there is a Boolean circuit computing $\text{Sel}^{n \rightarrow q}$ of size $O(n + q \cdot \log^3(n))$ and depth $O(\log(n + q))$. To the best of our knowledge, no linear-size circuit for multiselection was previously known for any $q = \omega(1)$, even without any restriction on the depth.

The main technical tools used in our construction are self-routing superconcentrators [25] and low-depth quasi-linear size sorting networks [1]. We also discuss applications of our linear-size multiselection circuit to problems in cryptography in Appendix A.

► **Remark 1 (Uniformity).** The circuits that we construct to prove Section 1 are *log-space uniform*, meaning that they can be generated in logarithmic space (and polynomial-time) by a uniform algorithm, given 1^n as input. Motivated by some of our applications, in Appendix B we also show an extension of the result with a *linear-time* uniform generation algorithm (on a word RAM machine), but only for $q < n^\varepsilon$, where $\varepsilon > 0$ is a universal constant.

1.1 Related Work

1.1.1 Tight Compaction

Several recent works [5, 22, 4, 3] study the related and central problem of tight compaction, in which the task is to retrieve all “marked” symbols of a string \mathbf{x} (in any order). Although this has a similar flavor to multiselection, there are several important differences. First, compaction circuits assume that each symbol x_i of \mathbf{x} is marked with a bit b_i indicating whether or not x_i is to be selected. In multiselection, we are instead given a *list* of indices i_1, \dots, i_q that should be selected. Given such a list, we know how to compute $(b_{i_1}, \dots, b_{i_q})$ with circuits only of size $\Omega(n \log^2 n)$. Second, the ordering of a compaction circuit’s output is unconstrained, while the ordering of a multiselection circuit’s output is determined by the ordering of its input indices i_1, \dots, i_q .

We address similar issues in our approach to constructing multiselection circuits. Interestingly, our construction implicitly utilizes a variant of compaction that can be instantiated in linear size using the [5] construction. We discuss this point further in our technical overview.

1.1.2 Uniselection Lower Bounds.

As noted above, it is easy to construct an $O(n)$ -size circuit for the *uniselection* problem (i.e., the case of $q = 1$). Interestingly though, the uniselection problem has been a source for some of the best known circuit and formula lower bounds that are known (for any explicit function).

In particular, Paul [24] gave a $2n - o(n)$ lower bound on the circuit complexity of uniselection (over any 2-bit gate basis). Blum [7] gave a $3n - o(n)$ lower bound for computing a closely related function (as a matter of fact this function involves computing a multiselection with $q = 3$ and applying a simple gadget function to the result). Nechiporuk [23] proved a quadratic lower bound on the formula size for a modified selection function and Andreev [2] proved an $n^{2.5 - o(1)}$ lower bound on the de Morgan formula size of roughly the same function.

2 Technical Overview

In this section we give an overview of the proof techniques underlying the proof of Section 1. In this overview we focus on achieving a *linear-size* multiselection circuit and defer the additional complications needed to simultaneously achieve *logarithmic-depth* to the technical sections.

The main technical tool that we use in our construction is a *superconcentrator*. Recall that a superconcentrator [28] is a constant-degree directed acyclic graph with $O(n)$ vertices, n of which are sources and n of which are sinks, with the property that for any $q \leq n$, any set S of sources, and any set T of sinks with $|S| = |T| = q$, there exist q vertex-disjoint paths from S to T .

2.1 Weak multiselection from superconcentrators

We first use superconcentrators to construct linear-size circuits for a *weak* form of multiselection (to be described shortly) and then show how to upgrade this construction to a circuit for full-fledged multiselection.

By replacing each vertex of a superconcentrator with a constant-sized routing gadget, we obtain for every $q \leq n$ a circuit \hat{C} of size $O(n)$ that implements the following weak form of multiselection: For any list of distinct indices $\mathbf{i} = (i_1, \dots, i_q) \in [n]^q$, there exists an “advice string” $\hat{\mathbf{i}} \in \{0, 1\}^{\Theta(n)}$ and a reordering j_1, \dots, j_q of i_1, \dots, i_q such that $\hat{C}(\mathbf{x}, \hat{\mathbf{i}}) = (x_{j_1}, \dots, x_{j_q})$, for all $\mathbf{x} \in \{0, 1\}^n$. The string $\hat{\mathbf{i}}$ describes the q vertex-disjoint paths from the source vertices i_1, i_2, \dots, i_q , to the sinks $1, 2, \dots, q$, by specifying how each vertex’s routing gadget should be configured.

This weak form of multiselection suffers from three drawbacks, which we will repair one by one:

1. The cost of computing $\hat{\mathbf{i}}$ might be large, and in particular super-linear in the length of \mathbf{x} ,
2. The output of \hat{C} is misordered – it is $(x_{j_1}, \dots, x_{j_q})$ instead of $(x_{i_1}, \dots, x_{i_q})$, and
3. The input \mathbf{i} is required to consist of *distinct* indices in $[n]$.

2.2 Amortizing the computation of the advice string

We address issue 1 by (temporarily) switching to a larger alphabet, which allows to amortize the cost of computing the advice string $\hat{\mathbf{i}}$. We refer to the latter as a “block multiselector”. Later we will show how to use a block multiselector to construct our desired (binary alphabet) multiselector.

In more detail, consider a generalization of multiselection, over an alphabet $\Sigma = \{0, 1\}^s$. The input is now a string $\mathbf{x} \in \Sigma^n$ and indices $i_1, \dots, i_q \in [n]$ and the goal, as before, is to output $(x_{i_1}, \dots, x_{i_q}) \in \Sigma^q$. Jumping ahead, it will suffice for us to set the block size s to be poly-logarithmic in n .

To construct the block multiselector, denoted \hat{C}_s , we simply take s copies of \hat{C} so that the j^{th} copy gets as input the j^{th} bit of each input block and produces the j^{th} bit of each output block; all copies of \hat{C} share the same advice string input. This circuit \hat{C}_s has the property that for all $\mathbf{i} \in [n]^q$ (consisting of distinct elements), there exists $\hat{\mathbf{i}} \in \{0, 1\}^{\Theta(n)}$ and a reordering $\mathbf{j} = (j_1, \dots, j_q)$ of \mathbf{i} such that for all $x \in (\{0, 1\}^s)^n$, it holds that $\hat{C}_s(x, \hat{\mathbf{i}}) = (x_{j_1}, \dots, x_{j_q})$.

The size of \hat{C}_s is $O(n \cdot s)$, but the computation of $\hat{\mathbf{i}}$ from i_1, \dots, i_q is entirely independent of s . Thus, by choosing s to be sufficiently large, we can hope to amortize the computation of $\hat{\mathbf{i}}$ relative to our input length, which is $\Omega(n \cdot s)$. There is an $O(n \log^2(n))$ circuit for computing $\hat{\mathbf{i}}$ for Pippenger’s self-routing superconcentrators [25], so by using these in the prior step it suffices now to set $s = \log^2(n)$.

Overall, we obtain a linear-size block multiselector C_s (i.e., of size $O(n \cdot s)$), for block size $s = \log^2(n)$ and number of queries $q \leq n$, with the following property: for all distinct $i_1, \dots, i_q \in [n]^q$, there exists a reordering j_1, \dots, j_q of i_1, \dots, i_q such that $C_s(\mathbf{x}, i_1, \dots, i_q) = (x_{j_1}, \dots, x_{j_q})$, for all $\mathbf{x} \in (\{0, 1\}^s)^n$. We note that such a circuit could also be constructed using linear-size circuits for tight compaction [5], in conjunction with Lemma 21.

2.3 Reordering the outputs and handling non-unique inputs

We next simultaneously resolve issues 2 (misordered output) and 3 (the restriction of unique indices). The high level idea is to append to each of the blocks its (original) block index. This means that after applying the block-multiselector from the previous step, we still keep track of the original location of this block, which we can combine with i_1, \dots, i_q to rearrange the output blocks in the desired order (and handle multiplicities appropriately).

In more detail, on input $\mathbf{x} \in (\{0, 1\}^s)^n$ and $\mathbf{i} \in [n]^q$, we construct the string $\mathbf{x}' := ((1, x_1), \dots, (n, x_n)) \in (\{0, 1\}^{\log(n)+s})^n$ as well as an index string $\mathbf{i}' \in [n]^q$ that consists of a sequence of *distinct* elements that contains all elements of \mathbf{i} (the string \mathbf{i}' can be constructed by sorting i_1, \dots, i_q , removing duplicates, and adding suitable padding). We then invoke $C_{s+\log n}$ on input $(\mathbf{x}', \mathbf{i}')$ to obtain the set

$$\{(i'_1, x_{i'_1}), \dots, (i'_q, x_{i'_q})\}, \quad (1)$$

represented as a list of elements (in some order).

We next combine (1) with

$$\{(1, i_1), \dots, (q, i_q)\} \quad (2)$$

to obtain

$$\{(1, i_1, x_{i_1}), \dots, (q, i_q, x_{i_q})\} \quad (3)$$

using the same representations of sets. This step, which can be viewed as an *inner join* (cf the database literature), combines the two lists based on the common keys i_1, \dots, i_q . It can be implemented in quasilinear size using sorting circuits. Once we have constructed the set in Equation (3) in some arbitrary order, we can sort its elements by their first coordinate to obtain the ordered list $((1, i_1, x_{i_1}), \dots, (q, i_q, x_{i_q}))$, from which we can read off $(x_{i_1}, \dots, x_{i_q})$.

The dominant costs in this construction arise from the usage of sorting circuits. In each instance, standard sorting circuits are applicable, with size $\tilde{O}(q) \cdot (s + \log n)$. If we use the sorting circuits of [1], then the $\tilde{O}(q)$ factor is in fact $O(q \log q)$, so (with $s = \log^2(n)$) it suffices to set $q \leq n / \log^3(n)$.

To summarize, we have now built a linear-size block-multiselection circuit for querying $q = n / \log^3(n)$ blocks of size $s = \log^2(n)$.

2.4 From block-multiselection back to bit-multiselection

Finally, we return to our original goal of multiselection over $\{0, 1\}$. We do so as follows: given queries i_1, \dots, i_q to *bits* of a string x , we group the bits of x into s -bit blocks $B_1, \dots, B_{n/s}$. We view each index i_j as consisting of a prefix $p_j \in [n/s]$ (specifying the block index) and a suffix $r_j \in [s]$ (specifying the internal index within the block). We apply the multiselection circuit of the previous step to obtain blocks B_{p_1}, \dots, B_{p_q} indexed by the $\log(n/s)$ -bit prefixes p_1, \dots, p_q of i_1, \dots, i_q . We then use q copies of a (uni-)selection circuit to obtain and output the r_j^{th} bit from each block B_{p_j} . The circuit implements the desired multiselection functionality and has size $O((n/s) \cdot s + q \cdot s) = O(n)$.

2.5 Achieving low-depth

A straightforward implementation of our construction has depth $\text{polylog}(n)$. Improving this to $O(\log n)$ requires significant care. For example, while we can use a logarithmic depth sorting network [1], that network uses abstract comparator gates. When working over

a large alphabet, the implementation of each comparator gate has super-constant depth (when implemented as a Boolean circuit), and so the resulting overall depth is ostensibly super-logarithmic. Nevertheless, we are able to provide suitable implementations of all of the gadgets so that the overall construction achieves logarithmic depth.

3 Preliminaries

We assume that the reader is familiar with the notion of a Boolean circuit and the associated function that it computes. We emphasize that, unless otherwise stated, throughout this work by “circuit” we refer to constant fan-in circuits over the standard De Morgan base.

3.1 Uniformity

We say that a family $\mathcal{C} = \{C_n\}_{n \in \mathbb{Z}^+}$ of Boolean circuits is *log-space uniform* if there exists a Turing machine that on input 1^n uses $O(\log n)$ space and prints the description of the circuit C_n , on a write-only output tape. The description is a listing of the gates in an (arbitrary) topological ordering along with a specification of the gate operation and pointers to each of its inputs. In Appendix B we also discuss an extension of our main result for *linear-time* uniformity.

3.2 Boolean Circuits for Uniselection

The following standard proposition gives a linear-size circuit, that given a string $\mathbf{x} \in \Sigma^n$, over an alphabet Σ , and an index $i \in [n]$, outputs the symbol x_i (in other words, a multiplexer).

► **Proposition 2.** *Let $s = s(n) \in \mathbb{Z}^+$ and let $\Sigma = \{0, 1\}^s$. There exists a log-space uniform Boolean circuit for $\text{Sel}_{\Sigma}^{n \rightarrow 1}$ with size $O(n \cdot s)$ and depth $O(\log n)$.*

Proof. Without loss of generality assume that $s = 1$ (for $s > 1$ we can use s parallel copies of the circuit for $s = 1$). The circuit follows a divide and conquer strategy. Assume for simplicity that n is a power of 2. Let $\mathbf{x} \in \{0, 1\}^n$ and $i \in [n]$ be the inputs. Let i_1 denote the most significant bit of i and let i' denote the remaining bits (i.e., $i' \in [n/2]$). To select the i^{th} bit of \mathbf{x} , the circuit recursively selects the $(i')^{\text{th}}$ bit of both the lower and upper halves of \mathbf{x} . Then, based on i_1 , it decides which of the two bits to output. Overall the circuit size satisfies the recursion: $S(n) = 2S(n/2) + O(1)$ and the depth satisfies $D(n) = D(n/2) + O(1)$. The proposition follows. ◀

3.3 Small-Size Sorting Circuits

Our circuits for multiselection heavily rely on circuits for sorting integers.

► **Lemma 3.** *For every $k = k(n)$ and $m = m(n)$, there is a log-space uniform Boolean circuit of size $O(n \cdot m \cdot \log n)$ for sorting n integers of m bits.*

This lemma follows immediately from the sorting networks of Ajtai, Komlós, and Szemerédi [1]. In order to make our multiselection circuit have logarithmic depth, we need a refined version of Lemma 3, which we present in Section 4.

4 Low-Depth Sorting of Logarithmic-Length Keys

We first recall what it means to sort with respect to a partial ordering.

► **Definition 4.** *Let Σ be a finite set. We say that $\mathbf{x}, \mathbf{y} \in \Sigma^n$ are reorderings of each other if for some permutation $\pi : [n] \rightarrow [n]$, it holds that $x_i = y_{\pi(i)}$, for all $i \in [n]$.*

► **Definition 5.** Let Σ be a finite set with a strict partial ordering \prec . A Boolean circuit is said to sort Σ^n with respect to \prec if on any input $\mathbf{x} \in \Sigma^n$, it outputs a reordering \mathbf{y} of \mathbf{x} such that for any $i, j \in [n]$, $y_i \prec y_j \implies i < j$.

In particular, we will focus on a partial ordering that compares integers by their k most significant bits. That is, for any $m \in \mathbb{Z}^+$ and any $\mathbf{x}, \mathbf{y} \in \{0, 1\}^m$, we write $\mathbf{x} \prec_k \mathbf{y}$ to denote that (x_1, \dots, x_k) lexicographically precedes (y_1, \dots, y_k) .

► **Lemma 6.** For every $m = m(n)$ and $k \in [m]$, there is a log-space uniform Boolean circuit of size $O(n \cdot m \cdot \log n)$ and depth $O(k + \log n)$ for sorting $(\{0, 1\}^m)^n$ with respect to \prec_k .

We are mainly interested in the setting $k = \Theta(\log n)$, in which case the constructed Boolean circuit has size $O(n \cdot m \cdot \log n)$ and depth $O(\log n)$.

To the best of our knowledge Lemma 6 was not previously known. The straight-forward circuit based on AKS networks yields circuits with matching size $O(n \cdot m \cdot \log n)$ but depth $O(\log n \cdot \log k)$. Kospanov [17] obtained circuits with better depth $O(\log n + \log k)$ but much larger size $O(n^2 \cdot k)$. A recent work of [18] obtains size $O(n \cdot k^2)$ and depth $O(\log n + k \log k)$, which is better for some values of $k = o(\log n)$, but worse in our regime of $k = \Theta(\log n)$. The work of Lin and Shi [22, Theorem 1.1] similarly constructs circuits with size $O(n \cdot m \cdot \log n)$ and depth $O(\log n)$ when $n > 2^{4k+7}$, but not for all values of $k = \Theta(\log n)$.

4.1 Sorting Networks

Our sorting circuits rely on sorting *networks*, which are generic constructions of n -input sorting circuits from 2-input sorting circuits.

► **Definition 7 (Sorting Networks).** An n -input sorting network is a circuit C with n inputs and n outputs, and with all gates having fan-in and fan-out two, such that for any set Σ with strict partial ordering \prec , if each gate of C is replaced by a circuit that sorts Σ^2 with respect to \prec , then the resulting circuit sorts Σ^n with respect to \prec .

For the best asymptotic size and depth, we use the celebrated sorting network of Ajtai, Komlós, and Szemerédi [1].

► **Lemma 8 ([1]).** There is a (log-space uniform) n -input sorting network with size $O(n \log n)$ and depth $O(\log n)$.

4.2 From Sorting Networks to Boolean Circuits

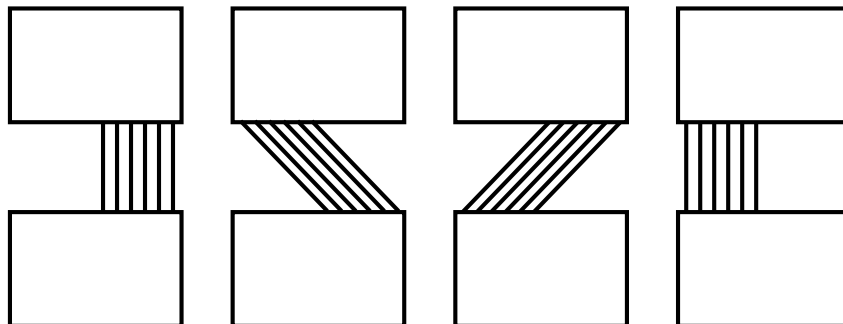
We prove Lemma 6 by combining the following lemma with Lemma 8.

► **Lemma 9.** Suppose that there is a log-space uniform sorting network on n elements with size $S = S(n)$ and depth $D = D(n)$. Then, for all $m = m(n)$ and $k \in [m]$, there is a log-space uniform Boolean circuit with size $O(S \cdot m)$ and depth $O(D + k)$ for sorting $(\{0, 1\}^m)^n$ with respect to \prec_k .

Proof. Starting with an n -input sorting network \mathcal{A} with size S and depth D , we obtain a Boolean circuit \mathcal{C} from \mathcal{A} by replacing each gate with a Boolean circuit $\text{Sort}_{k,m}^{(2)}$ that sorts $(\{0, 1\}^m)^2$ with respect to \prec_k . With this approach it is easy to make \mathcal{C} have size $O(S \cdot m)$ – since $\text{Sort}_{k,m}^{(2)}$ merely needs to have size $O(m)$.

Depth seems to pose more of a difficulty. Locality considerations imply that $\text{Sort}_{k,m}^{(2)}$ must have depth at least $\Omega(\log k)$ (e.g. the least significant bit of either output of $\text{Sort}_{k,m}^{(2)}(\mathbf{x}, \mathbf{y})$ depends on all of the k most significant bits of \mathbf{x} and \mathbf{y}). Thus, it might seem that this approach dooms \mathcal{C} to have depth $\Omega(D \cdot \log k)$.

We circumvent this difficulty by noting that although input-to-output paths in \mathcal{C} are concatenations of D input-to-output paths in $\text{Sort}_{k,m}^{(2)}$, these D paths cannot all be worst-case. In particular, the i^{th} output bit of one copy of $\text{Sort}_{k,m}^{(2)}$ is only connected to the j^{th} input bit of another copy if $i \equiv j \pmod{m}$ (see Figure 1).



■ **Figure 1** The four different ways that we can connect one copy of $\text{Sort}_{k,n}^{(2)}$ to another. In this depiction, a copy of $\text{Sort}_{k,n}^{(2)}$ is depicted by a rectangle, with inputs incident to the bottom edge and output incident to the top. The left half corresponds to the first input/output, and the right half to the second.

We leverage this with the following lemma, whose proof we defer momentarily.

► **Lemma 10.** *For any $m = m(n)$ and $k \in [m]$, there is a log-space uniform Boolean circuit $\text{Sort}_{k,m}^{(2)}$ with size $O(m)$ that sorts $(\{0,1\}^m)^2$ with respect to \prec_k and has the following additional property:*

For all $\hat{i}, \hat{j} \in [m]$ and $i, j \in [2m]$ with $i \equiv \hat{i} \pmod{m}$ and $j \equiv \hat{j} \pmod{m}$, every path in $\text{Sort}_{k,m}^{(2)}$ from the i^{th} input vertex to the j^{th} output vertex has length $O(\min(j, k) - \min(i, k) + 1)$. In particular there is no such path if $k > i > j$.

Assuming Lemma 10, consider any input-to-output path \mathbf{p} in \mathcal{C} . We write \mathbf{p} as a composition of paths $\mathbf{p}_1 \circ \dots \circ \mathbf{p}_D$, where each \mathbf{p}_ℓ is an input-to-output path of some copy of $\text{Sort}_{k,m}^{(2)}$. Define $i_\ell, j_\ell \in [2m]$ so that \mathbf{p}_ℓ starts at the i_ℓ^{th} input and ends at the j_ℓ^{th} output of that copy, and define $\hat{i}_\ell, \hat{j}_\ell \in [m]$ such that $i_\ell \equiv \hat{i}_\ell \pmod{m}$ and $j_\ell \equiv \hat{j}_\ell \pmod{m}$. By the observation above, we have $\hat{j}_\ell = \hat{i}_{\ell+1}$ for all $\ell \in [D-1]$, and by Lemma 10, we have for some constant L that $|\mathbf{p}_\ell| \leq L \cdot (\min(\hat{j}_\ell, k) - \min(\hat{i}_\ell, k) + 1)$.

Putting this together, we bound

$$\begin{aligned}
 |\mathbf{p}| &= \sum_{\ell=1}^D |\mathbf{p}_\ell| \\
 &\leq \sum_{\ell=1}^D L \cdot (\min(\hat{j}_\ell, k) - \min(\hat{i}_\ell, k) + 1) \\
 &= L \cdot (\min(\hat{j}_D, k) - \min(\hat{i}_1, k) + D) \quad (\text{telescoping sum because } \hat{j}_\ell = \hat{i}_{\ell+1} \text{ for } \ell \in [D-1]) \\
 &\leq L \cdot (k + D) \\
 &= O(k + D). \quad \blacktriangleleft
 \end{aligned}$$

It remains to prove Lemma 10.

11:8 Linear-Size Boolean Circuits for Multiselection

Proof of Lemma 10. We use the straightforward construction that compares the elements bit-by-bit starting with their most significant bits. In more detail, we will construct a circuit $\widetilde{\text{Sort}}_{k,m}^{(2)} : \{\prec, ?, \succ\} \times \{0, 1\}^m \times \{0, 1\}^m$ that has the functionality

$$\begin{aligned}\widetilde{\text{Sort}}_{k,m}^{(2)}(? , \mathbf{x}, \mathbf{y}) &= \begin{cases} (\mathbf{x}, \mathbf{y}) & \text{if } \mathbf{x} \prec \mathbf{y} \\ (\mathbf{y}, \mathbf{x}) & \text{otherwise.} \end{cases} \\ \widetilde{\text{Sort}}_{k,m}^{(2)}(\prec , \mathbf{x}, \mathbf{y}) &= (\mathbf{x}, \mathbf{y}) \\ \widetilde{\text{Sort}}_{k,m}^{(2)}(\succ , \mathbf{x}, \mathbf{y}) &= (\mathbf{y}, \mathbf{x})\end{aligned}$$

and we define $\text{Sort}_{k,m}^{(2)} = \widetilde{\text{Sort}}_{k,m}^{(2)}(? , \cdot, \cdot)$. This clearly gives the desired functionality. The first input symbol aids in recursively constructing the circuit for $\widetilde{\text{Sort}}_{k,m}^{(2)}$, and is meant to indicate whether the global decision on the comparison between x and y is (1) still undecided (?), (2) decided toward x or (3) decided toward y .

Our construction of $\widetilde{\text{Sort}}_{k,m}^{(2)}$ is recursive. If $k = 0$, then we define $\widetilde{\text{Sort}}_{k,m}^{(2)}$ to be the identity function. For $k > 0$, $\widetilde{\text{Sort}}_{k,m}^{(2)}$ takes input $(c, \mathbf{x}, \mathbf{y})$, it first computes

$$(c', a_1, b_1) = \begin{cases} (\prec, x_1, y_1) & \text{if } c = \prec \text{ or } (c = ? \text{ and } x_1 < y_1) \\ (? , x_1, y_1) & \text{if } c = ? \text{ and } x_1 = y_1 \\ (\succ, y_1, x_1) & \text{if } c = \succ \text{ or } (c = ? \text{ and } x_1 > y_1). \end{cases}$$

Then, with \mathbf{x}' and \mathbf{y}' denoting (x_2, \dots, x_m) and (y_2, \dots, y_m) , it computes

$$(\mathbf{a}', \mathbf{b}') = \widetilde{\text{Sort}}_{k-1, m-1}^{(2)}(c', \mathbf{x}', \mathbf{y}').$$

Finally it outputs $(a_1 \circ \mathbf{a}', b_1 \circ \mathbf{b}')$, where \circ denotes string concatenation.

It is easy to check the correctness of this construction.

Circuit Size. Let $S(k, m)$ denote the size of $\widetilde{\text{Sort}}_{k,m}^{(2)}$. For $k = 0$ we clearly have $S(k, m) = O(m)$. For larger k , the recursive construction gives $S(k, m) = S(k-1, m-1) + O(1)$, so by induction $S(k, m) = O(m)$ for all $k \leq m$.

Input-to-Output Path Lengths. We now bound the length of different input-to-output paths in $\widetilde{\text{Sort}}_{k,m}^{(2)}$. As in the construction above, denote the input to $\widetilde{\text{Sort}}_{k,m}^{(2)}$ by $(c, \mathbf{x}, \mathbf{y})$ and denote the output by (\mathbf{a}, \mathbf{b}) .

▷ **Claim 11.** There exists a constant L such that any path in $\widetilde{\text{Sort}}_{k,m}^{(2)}$ from an input $\{x_i, y_i\}$ to an output $\{a_j, b_j\}$ has length at most $L \cdot (j - i + 1)$, and any path from the input c to an output $\{a_j, b_j\}$ has length at most $L \cdot j$.

In particular any path to an output $\{a_j, b_j\}$ has length at most $L \cdot j$.

Proof. Let $(\mathbf{x}', \mathbf{y}')$ denote the input to the recursive call to $\widetilde{\text{Sort}}_{k-1, m-1}^{(2)}$ and let $(\mathbf{a}', \mathbf{b}')$ denote the output of this call.

We consider several cases separately:

- Paths to $\{a_1, b_1\}$ have length bounded by an absolute constant because a_1 and b_1 are just a (constant-sized) function of the inputs (c, x_1, y_1) . If L is sufficiently large then this constant is at most L .
- For $j > 1$, paths from $\{c, x_1, y_1\}$ to $a_j (= a'_{j-1})$ consist of a path through a constant-sized circuit (the computation of c') concatenated with a path to a'_{j-1} in $\widetilde{\text{Sort}}_{k-1, m-1}^{(2)}$. By induction the latter path has length at most $L \cdot (j-1)$, so if L is sufficiently large then the total path length is at most $L \cdot j$.
- For $i > 1$ and $j > 1$, paths from $\{x_i, y_i\}$ to $\{a_j, b_j\}$ are just paths from $\{x'_{i-1}, y'_{i-1}\}$ to $\{a'_{j-1}, b'_{j-1}\}$ and thus by induction they have length at most $L \cdot (j-i+1)$. ◁

Returning to $\text{Sort}_{k,m}^{(2)} = \widetilde{\text{Sort}}_{k,m}^{(2)}(\cdot, \cdot, \cdot)$ (without the tilde), we can see that any input-to-output path in $\text{Sort}_{k,m}^{(2)}$ directly corresponds to an input-to-output path in $\widetilde{\text{Sort}}_{k,m}^{(2)}$ whose length is appropriately bounded by Claim 11. That concludes the proof of Lemma 10. ◀

5 Unordered Multiselection Over Large Alphabets

In this section we construct a circuit that implements a relaxation of multiselection. Informally, this relaxation allows the output elements to appear in any order (and possibly with repetitions), rather than the same order in which they were queried. Additionally, rather than selecting individual bits (which is our eventual goal) - we consider a generalization to a larger alphabet size.

► **Definition 12.** Unordered multiselection (with q queries, over alphabet Σ) is the search problem defined by the relation

$$\widetilde{\text{Sel}}_{\Sigma}^{n \rightarrow q} \stackrel{\text{def}}{=} \left\{ ((\mathbf{x}, \mathbf{i}), \mathbf{y}) \in (\Sigma^n \times [n]^q) \times (\Sigma \cup \{\perp\})^q : \{y_k : y_k \neq \perp\}_{k \in [q]} = \{x_{i_k}\}_{k \in [q]} \right\}.$$

5.1 Superconcentrators and Routing

We recall the notion of a superconcentrator, which is the main technical tool involved in our construction of Boolean circuits for unordered multiselection.

► **Definition 13 (Networks).** A network \mathcal{N} is a tuple $\mathcal{N} = (V, E, A, B)$, where $G = (V, E)$ is a directed acyclic graph and $A, B \subseteq V$ are respectively sets of sources and sinks in G .

If $|A| = |B| = n$, then \mathcal{N} is said to be an n -network. The size of \mathcal{N} , denoted by $|\mathcal{N}|$, is defined to be $|E|$; the degree of \mathcal{N} is the degree of G ; and the depth of \mathcal{N} is the longest path length in G .

A family of n -networks $\{\mathcal{N}_n\}_{n \in \mathbb{Z}^+}$ is said to be log-space uniform if there an algorithm that on input 1^n runs in $O(\log n)$ space and outputs G (say with an adjacency list representation) and A, B (say represented by indicator strings).

► **Definition 14 (Superconcentrators).** An n -network $\mathcal{N} = (V, E, A, B)$ is said to be a superconcentrator if for all $q \in [n]$ and all subsets $X \subseteq A$ and $Y \subseteq B$ with $|X| = |Y| = q$, there exist q vertex-disjoint paths from X to Y .

► **Definition 15.** Let $\mathcal{S} = (V, E, A, B)$ be an n -superconcentrator. The routing problem for \mathcal{S} is a search problem $\text{Route}_{\mathcal{S}}$ whose input is a pair of (indicator strings for) sets $X \subseteq A$, $Y \subseteq B$ with $|X| = |Y|$. A valid corresponding output is any (indicator string for a) set $R \subseteq E$ of edges such that R is a union of q vertex-disjoint paths from X to Y , where $q = |X| = |Y|$.

11:10 Linear-Size Boolean Circuits for Multiselection

Note that whenever \mathcal{S} is an n -superconcentrator, the search problem $\text{Route}_{\mathcal{S}}$ is total. Naturally, it is desirable for a superconcentrator \mathcal{S} to admit efficient algorithms for $\text{Route}_{\mathcal{S}}$. The state of the art in this respect is Pippenger's construction of a "self-routing" superconcentrator.

Informally, a delay- d self-routing protocol for a superconcentrator (V, E, A, B) associates each vertex $v \in V$ with a finite automaton that can communicate synchronously with the automata on neighboring vertices (where u is said to neighbor v if (u, v) or (v, u) is in E). For any sets $X \subseteq A$ and $Y \subseteq B$ with $|X| = |Y| = q$, if the automata in X and Y are initialized with state 1, and all other automata are initialized with state 0, then in d steps the automata jointly compute q vertex-disjoint paths from X to Y by on the d^{th} step transmitting 1 on all edges in those paths and 0 on other edges.

► **Definition 16.** A self-routing protocol with delay d for an n -superconcentrator $\mathcal{S} = (V, E, A, B)$ is a tuple (Σ, δ) , where:

- Σ is a finite "alphabet" set that contains $\{0, 1\}$; and
- $\delta : \Sigma^{V \cup E} \rightarrow \Sigma^{V \cup E}$ is a function such that:
 - for every vertex $v \in V$ with incident edges E_v , $\delta(\mathbf{q})_v$ depends as a function of \mathbf{q} only on $(q_e)_{e \in E_v}$.
 - for every edge $e = (u, v) \in E$, $\delta(\mathbf{q})_e$ depends as a function of \mathbf{q} only on q_u and q_v .
- For any sets $X \subseteq A$ and $Y \subseteq B$ with $|X| = |Y|$, if we define $\mathbf{q}^{(0)} \in \Sigma^{V \cup E}$ by

$$q_v^{(0)} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } v \in X \cup Y \\ 0 & \text{if } v \in V \setminus (X \cup Y) \end{cases}$$

and $q_e^{(0)} \stackrel{\text{def}}{=} 0$ for all $e \in E$, then the set

$$R \stackrel{\text{def}}{=} \{e \in E : \delta^d(\mathbf{q}^{(0)})_e = 1\}$$

satisfies $((1_X, 1_Y), 1_R) \in \text{Route}_{\mathcal{S}}$.

Such a self-routing protocol for an n -superconcentrator is said to be log-space uniform if there is a log-space Turing machine that on input 1^n outputs:

- a description of Σ ;
- for each $v \in V$ with incident edges E_v , the truth table for $\delta(\mathbf{q})_v$ as a function of $(q_e)_{e \in E_v}$.
- for each $e = (u, v) \in E$, the truth table for $\delta(\mathbf{q})_e$ as a function of (q_u, q_v) .

► **Definition 17.** A self-routing superconcentrator is a superconcentrator \mathcal{S} with an associated self-routing protocol (Σ, δ) for \mathcal{S} . A self-routing superconcentrator is said to be log-space uniform if both the superconcentrator and the associated self-routing protocol are log-space uniform.

► **Theorem 18** ([25]). There exists a log-space uniform self-routing n -superconcentrator \mathcal{S} with size $O(n)$, degree $O(1)$, and depth $O(\log n)$, where the self-routing protocol has alphabet size $O(1)$ and delay $O(\log n)$.

► **Corollary 19.** There exists a log-space uniform n -superconcentrator \mathcal{S} with size $O(n)$, degree $O(1)$, and depth $O(\log n)$, such that $\text{Route}_{\mathcal{S}}$ is solvable by a log-space uniform Boolean circuit of size $O(n \log n)$ and depth $O(\log n)$.

Proof. Let $\mathcal{S} = (V, E, A, B)$ be the log-space uniform self-routing superconcentrator given by Theorem 18, let the self-routing protocol be given by (Σ, δ) , and let $d = O(\log n)$ denote the delay of this self-routing protocol. To prove the corollary, we need to construct a log-space uniform Boolean circuit for solving $\text{Route}_{\mathcal{S}}$, with size $O(n \log n)$ and depth $O(\log n)$.

We are given 1_X and 1_Y . The circuit starts by computing $\mathbf{q}^{(0)} \in \Sigma^{V \cup E}$ as defined in Definition 16. By construction this can be done with a circuitry of size $O(n)$ and depth $O(1)$. The circuit next computes $\delta^d(\mathbf{q}^{(0)})$. Recall that each output symbol of $\delta(\mathbf{q})$ depends on at most $\deg(\mathcal{S}) = O(1)$ symbols of \mathbf{q} . Since the alphabet Σ is also constant-sized, this means that δ is implementable by a (log-space uniform) circuit of size $O(|V \cup E|) = O(n)$ and depth $O(1)$. Iterating this circuit d times, we compute $\delta^d(\mathbf{q}^{(0)})$ with log-space uniform circuitry of size $O(d \cdot n)$ and depth $O(d)$. Finally, we extract from $\delta^d(\mathbf{q}^{(0)})$ the indicator string 1_R for the set

$$R \stackrel{\text{def}}{=} \{e \in E : \delta^d(\mathbf{q}^{(0)})_e = 1\}.$$

This takes (log-space uniform) circuitry of size $O(|E|) = O(n)$ and depth $O(1)$.

In total, this circuit for computing 1_R is log-space uniform and has size $O(n \cdot d) = O(n \log n)$ and depth $O(d) = O(\log n)$. By the definition of a self-routing protocol, R satisfies $((1_X, 1_Y), 1_R) \in \text{Routes}_{\mathcal{S}}$, i.e. the circuit solves $\text{Routes}_{\mathcal{S}}$. ◀

5.2 From Superconcentrators To Unordered Multiselection

► **Proposition 20.** *Let $n \in \mathbb{Z}^+$, let $s = s(n)$ satisfy $s = \Omega(\log^2 n)$, and let $\Sigma = \{0, 1\}^s$. For any $q = q(n) \in [n]$ there exists a log-space uniform Boolean circuit for $\widetilde{\text{Sel}}_{\Sigma}^{n \rightarrow q}$ with size $O(n \cdot s)$ and depth $O(\log n)$.*

Proof. Let $\mathcal{S} = (V, E, A, B)$ be a log-space uniform n -superconcentrator with size $O(n)$, depth $O(\log n)$, degree $O(1)$ such that $\text{Routes}_{\mathcal{S}}$ is solvable by a log-space uniform Boolean circuit of size $O(n \log n)$ and depth $O(\log n)$. (Such an \mathcal{S} is guaranteed to exist by Corollary 19). Order the elements of A and B arbitrarily so that $A = \{a_1, \dots, a_n\}$ and $B = \{b_1, \dots, b_n\}$.

The Circuit. Our circuit, taking as input $\mathbf{x} \in \Sigma^n$ and $(i_1, \dots, i_q) \in [n]^q$, is constructed in two parts.

The first part of our circuit computes indicator strings $1_X \in \{0, 1\}^A$, $1_Y \in \{0, 1\}^B$, and $1_R \in \{0, 1\}^E$ for sets $X \subseteq A$, $Y \subseteq B$, and $R \subseteq E$ defined as follows. The set X is defined as $\{a_{i_1}, \dots, a_{i_q}\}$, Y is defined as $\{b_1, \dots, b_{q'}\}$, where q' is the cardinality of X (which may be smaller than q due to repetitions), and R is the edges of q' vertex-disjoint paths from X to Y .

The second part of our circuit consists of a gadget g_v for each vertex $v \in V$. For a vertex $v \in A$, say $v = a_j$, the gadget g_v simply outputs x_j if $v \in X$ and \perp otherwise. For other v , i.e. with positive in-degree d , the gadget g_v is a circuit for $\text{Sel}_{\Sigma \cup \{\perp\}}^{d \rightarrow 1}$ (e.g. the circuit from Proposition 2). To construct the inputs for g_v , first order the in-neighbors of v arbitrarily as u_1, \dots, u_d . The data input for g_v is then constructed by taking the i^{th} symbol, for any $i \in [d]$, to be the output of g_{u_i} . The selector input for g_v is constructed to be (the unique) $i^* \in [d]$ such that $(u_{i^*}, v) \in R$ if such an i^* exists, and arbitrary otherwise.

Finally, the outputs of our circuit are the outputs of g_{b_1}, \dots, g_{b_q} .

Size and Depth. In the first part of our circuit, the computation of 1_X uses log-space uniform circuitry of size $O(n \log^2 n) = O(n \cdot s)$ and depth $O(\log n)$ by Lemma 21 below. Next, 1_Y is obtained by sorting 1_X with log-space uniform circuitry of size $O(n \log n)$ and depth $O(\log n)$ (this is possible by Lemma 8). Finally, the computation of 1_R uses log-space uniform circuitry of size $O(n \log n)$ and depth $O(\log n)$ because of our choice of superconcentrator \mathcal{S} . In total the circuitry of this part has size $O(n \cdot s)$ and depth $O(\log n)$.

11:12 Linear-Size Boolean Circuits for Multiselection

In the second part of our circuit, there are $O(n)$ gadgets (because \mathcal{S} has size $O(n)$), and each gadget has size $O(s)$ (because \mathcal{S} has constant degree). Additionally for each non-source gadget there is constant-sized circuitry to compute the selector input as a function of 1_R (again because \mathcal{S} has constant degree). In total the circuitry size is $O(n \cdot s)$. As for depth, each gadget has constant depth. Since the output of gadget u is an input to gadget v iff (u, v) is an element of E , and \mathcal{S} has depth $O(\log n)$, the circuitry here also has depth $O(\log n)$.

Correctness. The main claim that we use to establish correctness is that for every $a_i \in X$ and every vertex $v \in V$, if v lies on a path in R from a_i to B , then the output of g_v is x_i , and otherwise the output of g_v is \perp . Here a_i is well-defined as a partial function of v because R consists only of vertex-disjoint paths. This claim suffices for correctness because there are paths in R from each a_{i_k} to B , but not from any $a \in A \setminus \{i_1, \dots, i_q\}$, so the set of non- \perp outputs is exactly $\{x_{i_1}, \dots, x_{i_q}\}$.

We prove the claim by induction on the depth of v (i.e. the maximum length of a path ending in v). The base case is when v has depth 0, i.e. v is a source vertex a_i in (V, E) . In this case we *defined* g_v to output x_i .

For the inductive step, consider a vertex v with positive depth, and suppose that the claim holds for all u of lesser depth, and in particular for the in-neighbors u_1, \dots, u_d of v . Now if v lies on a path p from some a_i to Y , this path must go through u_j for some $j \in [d]$, and then the inductive hypothesis implies that the output of g_{u_j} is x_i . The construction of the selector input for g_v ensures that g_v also outputs x_i , which completes the proof of the claim. \blacktriangleleft

5.3 Computing Set Indicator Strings

► **Lemma 21.** *For $q = q(n) \in \mathbb{Z}^+$, there is a log-space uniform Boolean circuit with size $O((n+q) \cdot \log^2(n+q))$ and depth $O(\log(n+q))$ that maps $(i_1, \dots, i_q) \in [n]^q$ to the indicator string $1_{\{i_1, \dots, i_q\}} \in \{0, 1\}^n$.*

Proof. On input (i_1, \dots, i_q) , the circuit performs the following steps:

1. Construct the list $(1, \dots, n, i_1, \dots, i_q) \in [n]^{n+q}$, and sort it to obtain a non-decreasing list $(j_i)_{i \in [n+q]}$ with the property that j_i appears more than once in the list iff $j_i \in \{i_1, \dots, i_q\}$. This can be done with circuitry of size $O((n+q) \cdot \log^2(n+q))$ and depth $O(\log(n+q))$ by Lemma 6.
2. Label j_i with 0 if the value j_i appears once in the list. If j_i appears more than once, we label the first occurrence with 1 and all other occurrences with \perp . This labels j_i with 1 only if j_i is in $\{i_1, \dots, i_q\}$, and labels j_i with 0 only if it isn't.

More explicitly, we compute labels

$$b_i = \begin{cases} \perp & \text{if } i > 1 \text{ and } j_{i-1} = j_i \\ 1 & \text{otherwise, if } i < n+q \text{ and } j_i = j_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

for $i \in [n+q]$. This can be done with circuitry of size $O((n+q) \log n)$ and depth $O(\log \log n)$: first compute in parallel for each $i \in [n+q-1]$ whether or not $j_i = j_{i+1}$ (this circuit has size $O(\log n)$ and depth $O(\log \log n)$), and then compute each b_i with a constant-sized circuit.

3. Sort the list $((j_i, b_i))_{i \in [n+q]}$ in order of increasing j_i , except that if $b_i = \perp$ we treat j_i as $+\infty$. That is, we prepend each j_i with 1 if $b_i = \perp$ and with 0 otherwise. We then take the first n elements of the result to obtain a list $((i, b'_i))_{i \in [n]}$ such that $b'_i \in \{0, 1\}$ satisfies

$$b'_i = \begin{cases} 1 & \text{if } i \in \{i_1, \dots, i_q\} \\ 0 & \text{otherwise.} \end{cases}$$

In other words, (b'_1, \dots, b'_n) is $1_{\{i_1, \dots, i_q\}}$, which is our desired output. This step can be done with circuitry of size $O((n+q) \cdot \log^2(n+q))$ and depth $O(\log(n+q))$ by Lemma 6. ◀

6 From Unordered to Ordered Multiselection

In this section we construct a circuit for *ordered* multiselection over a large alphabet from a circuit for *unordered* multiselection over a slightly larger alphabet.

► **Lemma 22.** *Let $q = q(n) \in [n]$ and $s = s(n) = \Omega(\log n)$ be integer functions of n , let $\Sigma = \{0, 1\}^s$, let $\tilde{\Sigma} = [n] \times \Sigma$, and suppose there is a (log-space uniform) Boolean circuit for $\widetilde{\text{Sel}}_{\tilde{\Sigma}}^{n/s \rightarrow q}$ with size S and depth D .*

Then there is a (log-space uniform) Boolean circuit for $\text{Sel}_{\Sigma}^{n \rightarrow q}$ with size $S + O(q \cdot s \cdot \log q)$ and depth $D + O(\log n)$.

Combining Lemma 22 with Proposition 20 gives the following corollary.

► **Corollary 23.** *Let $s = s(n)$ satisfy $s = \Omega(\log^2 n)$, let $\Sigma = \{0, 1\}^s$, and let $q = q(n) \in [n]$. Then there is a log-space uniform Boolean circuit for $\text{Sel}_{\Sigma}^{n \rightarrow q}$ with size $O(n \cdot s + q \cdot s \cdot \log n)$ and depth $O(\log n)$.*

6.1 Boolean Circuits for Inner Joins

The main tool that we use in the proof of Lemma 22 is Boolean circuitry for computing an *inner join* of two relations (cf. relational databases [12]).

► **Definition 24.** *If $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{X}$ and $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y}$ are relations, the inner join of \mathcal{R} and \mathcal{S} (which we denote by $\mathcal{R} \bowtie \mathcal{S}$) is*

$$\mathcal{R} \bowtie \mathcal{S} \stackrel{\text{def}}{=} \{(w, x, y) : (w, x) \in \mathcal{R} \wedge (x, y) \in \mathcal{S}\}.$$

To our knowledge, prior work on the computational complexity of computing joins has focused on algorithms in the RAM or PRAM models of computation, as opposed to Boolean circuits.

We focus for simplicity on a special case. First, we require that the relation \mathcal{S} is a partial function. That is, for every $x \in \mathcal{X}$ there is at most one $y \in \mathcal{Y}$ such that $(x, y) \in \mathcal{S}$. The partial function requirement prevents $|\mathcal{R} \bowtie \mathcal{S}|$ from being larger than $|\mathcal{R}|$. We also require that for every $(w, x) \in \mathcal{R}$ there exists some $(x, y) \in \mathcal{S}$.

The following proposition says that inner joins in this special case are computable by (uniform) Boolean circuits of logarithmic depth and nearly linear size.

► **Proposition 25.** *Let n, m be positive integers, let \mathcal{W} and \mathcal{Y} be sets whose elements are represented by s -bit strings, and let \mathcal{X} be a finite set whose elements are represented by k -bit strings.*

There exists a log-space uniform Boolean circuit of size $O(q \cdot (k + s) \cdot \log q)$ and depth $O(k + \log q)$ that takes as input a relation $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{X}$ and a partial function $f \subseteq \mathcal{X} \times \mathcal{Y}$ with $|\mathcal{R}| \leq q$, $|f| \leq q$, and $\{x : \exists w, (w, x) \in \mathcal{R}\} \subseteq \{x : \exists y, (x, y) \in f\}$, and outputs $\mathcal{R} \bowtie f$.

Here sets are represented by an arbitrarily ordered listing of their elements, padded with \perp elements as needed to have length q .

11:14 Linear-Size Boolean Circuits for Multiselection

Proof. Denote the elements of $\{x : \exists y, (x, y) \in f\}$ by x_1, \dots, x_k , where $x_1 < \dots < x_k$. For $i \in [k]$, let W_i denote the set $\{w : (w, x_i) \in \mathcal{R}\}$, and let n_i denote $|W_i|$. With this notation, our desired output is a list of length q whose non- \perp elements in some order are precisely $(w, x_i, f(x_i))_{i \in [k], w \in W_i}$.

We first construct a list with at most $2q$ elements in $(\mathcal{W} \cup \{\star\}) \times \mathcal{X} \times (\mathcal{Y} \cup \{\star\})$, namely $\{(x_i, w, \star)\}_{i \in [k], w \in W_i}$ and $\{(x_i, \star, f(x_i))\}_{i \in [k]}$. Such a list is readily obtained by concatenating the listings of \mathcal{R} and f (with the natural embeddings of $\mathcal{W} \times \mathcal{X}$ and $\mathcal{X} \times \mathcal{Y}$ in $(\mathcal{W} \cup \star) \times \mathcal{X} \times (\mathcal{Y} \cup \star)$).

We sort this list with respect to the partial ordering that defines $(w, x, y) \prec (w', x', y')$ iff $x < x'$ or $x = x'$ and $w = \star$ and $w' \neq \star$. By Lemma 6 this can be done with circuitry of size $O(q \cdot (k + s) \cdot \log q)$ and depth $O(\log q)$. This yields a list \mathcal{L} composed of k blocks, the i^{th} of which has length $n_i + 1$ and is

$$\left((\star, x_i, f(x_i)), (w_{i,1}, x_i, \star), \dots, (w_{i,n_i}, x_i, \star) \right),$$

where $\{w_{i,1}, \dots, w_{i,n_i}\} = W_i$.

With local processing, we obtain two lists \mathcal{L}_f and \mathcal{L}_S where the i^{th} block of \mathcal{L}_f is

$$\left(\underbrace{f(x_i), \perp, \dots, \perp}_{n_i \text{ times}} \right) \quad (4)$$

and the i^{th} block of \mathcal{L}_S is

$$\left(\perp, (w_{i,1}, x_i), \dots, (w_{i,n_i}, x_i) \right). \quad (5)$$

Using Lemma 26 below, we map \mathcal{L}_f to a list whose i^{th} block is

$$\left(\underbrace{f(x_i), \dots, f(x_i)}_{n_i + 1 \text{ times}} \right). \quad (6)$$

We then locally combine (6) with (5) to obtain a list whose i^{th} block is

$$\left(\perp, ((w_{i,1}, x_i, f(x_i))), \dots, (w_{i,n_i}, x_i, f(x_i)) \right). \quad (7)$$

A final sorting step sends the \perp elements to the back of the list, which allows us to conclude by truncating the list to length n . This step can also be done with circuitry of size $O(q \cdot (k + s) \cdot \log q)$ and depth $O(\log q)$ by Lemma 6. \blacktriangleleft

► Lemma 26. For $n, s \in \mathbb{Z}^+$, there is a (logspace-uniform) constant fan-in Boolean circuit of size $O(n \cdot s)$ and depth $O(\log n)$ that takes as input $\mathbf{x} \in (\{0, 1\}^s \cup \{\perp\})^n$ with $x_1 \neq \perp$, and outputs $\mathbf{y} \in (\{0, 1\}^s)^n$ such that

$$y_i = \begin{cases} x_i & \text{if } x_i \neq \perp \\ y_{i-1} & \text{otherwise.} \end{cases}$$

Equivalently, $y_i = x_{j_i}$, where $j_i = \max\{j : 1 \leq j \leq i \wedge x_j \neq \perp\}$ (this maximum is guaranteed to be over a non-empty set because $x_1 \neq \perp$).

Proof. Without loss of generality we can assume that $s = 1$ because for $s > 1$ we can use s copies of the circuit for the $s = 1$ case.

Consider the binary operation $\star : \{0, 1, \perp\} \times \{0, 1, \perp\} \rightarrow \{0, 1, \perp\}$ defined by

$$\tau \star v = \begin{cases} \tau & \text{if } v = \perp \\ v & \text{if } v \neq \perp. \end{cases}$$

It is clear that our desired \mathbf{y} has $y_1 = x_1$ and $y_i = y_{i-1} \star x_i$ for $i > 1$.

We now observe that \star is an associative operation. To see this, consider any $\sigma, \tau, v \in \{0, 1, \perp\}$ and consider separately the cases $v = \perp$ and $v \neq \perp$. If $v = \perp$ then $(\sigma \star \tau) \star v = \sigma \star \tau = \sigma \star (\tau \star v)$. If $v \neq \perp$ then $(\sigma \star \tau) \star v = v = \sigma \star v = \sigma \star (\tau \star v)$.

Now we use a classic result of Ladner and Fischer [20] that for *any* associative operation $\star : \Sigma \times \Sigma \rightarrow \Sigma$ and $n \in \mathbb{Z}^+$, there exists a (log-space uniform) circuit of size $O(n)$ and depth $O(\log n)$ (with only \star -gates) that computes all \star -prefix products. That is, the circuit takes as input $\mathbf{x} \in \Sigma^n$ and outputs \mathbf{y} such that $y_i = x_1 \star \dots \star x_i$.

In our case, \star is computable by a Boolean circuit of constant size, so replacing \star -gates by such a circuit finishes the proof. \blacktriangleleft

6.2 A Circuit for Ordered Multiselection

We are now ready to prove Lemma 22.

Proof of Lemma 22. Let \tilde{C} be a circuit for $\widetilde{\text{Sel}}_{\Sigma}^{n \rightarrow q}$.

The Circuit. Our circuit for $\text{Sel}_{\Sigma}^{n \rightarrow q}$ takes as input $(\mathbf{x}, \mathbf{i}) \in \Sigma^n \times [n]^q$ and consists of the following stages:

1. Compute $\tilde{\mathbf{x}} \in \tilde{\Sigma}^n$, defined so that $\tilde{x}_i = (i, x_i)$ for $i \in [n]$, and compute a list representation of

$$\mathcal{I} = \{(1, i_1), \dots, (q, i_q)\}.$$

2. Compute $\tilde{\mathbf{y}} \leftarrow \tilde{C}(\tilde{\mathbf{x}}, \mathbf{i})$ to obtain $\tilde{\mathbf{y}} \in (\tilde{\Sigma} \cup \{\perp\})^q$ such that

$$\{\tilde{y}_k : \tilde{y}_k \neq \perp\}_{k \in [q]} = \{\tilde{x}_{i_k}\}_{k \in [q]} = \{(i_k, x_{i_k})\}_{k \in [q]}.$$

In particular, each \tilde{y}_k that is not equal to \perp has the form (i, x_i) for some $i \in [n]$.

3. Deduplicate $\tilde{\mathbf{y}}$ to obtain a list representation of the partial function

$$\mathcal{Y} = \{(i_k, x_{i_k})\}_{k \in [q]}.$$

This involves sorting the non- \perp symbols $\{(i_k, x_{i_k})\}$ of $\tilde{\mathbf{y}}$ in order of increasing i_k (this can be done with circuitry of size $O(n \cdot s \cdot \log n)$ and depth $O(\log n)$ by Lemma 6).

4. Apply the circuit given by Proposition 25 to \mathcal{I} and \mathcal{Y} to compute a list representation of

$$\mathcal{I} \bowtie \mathcal{Y} = \{(k, i_k, x_{i_k})\}_{k \in [q]}.$$

5. Sort the list representation of $\mathcal{I} \bowtie \mathcal{Y}$ in order of increasing k , and read off the desired output $(x_{i_1}, \dots, x_{i_q})$.

Size and Depth. The “computation” of $\tilde{\mathbf{x}}$ from \mathbf{x} and of \mathcal{I} from \mathbf{i} is just adding constant values, so it can certainly be done by a (log-space uniform) Boolean circuit of size $O(n \cdot (s + \log n))$ and depth $O(1)$.

By assumption, \tilde{C} is a circuit of size S and depth D .

Via sorting (Lemma 6), one can deduplicate $\tilde{\mathbf{y}}$ with circuitry of size $O(q \cdot (s + \log n) \cdot \log q) = O(q \cdot s \cdot \log q)$ and depth $O(\log n)$.

By Proposition 25, the computation of $\mathcal{I} \bowtie \mathcal{Y}$ can also be done with circuitry of size $O(q \cdot (s + \log n) \cdot \log q) = O(q \cdot s \cdot \log n)$ and depth $O(\log n)$.

Sorting the elements of $\mathcal{I} \bowtie \mathcal{Y}$ again takes log-space uniform circuitry of size $O(q \cdot s \cdot \log q)$ and depth $O(\log n)$ by Lemma 6.

In total our constructed circuit has size $S + O(q \cdot s \cdot \log q)$ and depth $D + O(\log n)$. ◀

7 Binary Multiselection and Proof of Main Theorem

Next, we use the multiselection circuit over large alphabets to obtain a *binary* multiselection circuit.

► **Lemma 27.** *Let $q = q(n)$ and $s = s(n)$ be integer functions of n , let $\Sigma = \{0, 1\}^s$, and let C_n be a (log-space uniform) Boolean circuit for $\text{Sel}_{\Sigma}^{n/s \rightarrow q}$ with size S and depth D .*

Then there is a (log-space uniform) Boolean circuit C'_n for $\text{Sel}_{\{0,1\}}^{n \rightarrow q}$ with size $S + O(q \cdot s)$ and depth $D + O(\log s)$.

Proof. On data input $\mathbf{x} \in \{0, 1\}^n$ and selector input $(i_1, \dots, i_q) \in [n]^q$, C'_n performs the following steps:

1. View $\mathbf{x} \in \{0, 1\}^n$ as $\mathbf{X} \in \Sigma^{n/s}$ by setting $X_i = (x_{(i-1) \cdot s + 1}, \dots, x_{i \cdot s})$. View each i_j as a $\log n$ -bit string with prefix $p_j \in \{0, 1\}^{\log n - \log s}$ and a suffix $s_j \in \{0, 1\}^{\log s}$. This is just relabeling wires, so it requires no circuitry.
2. Compute $(X_{p_1}, \dots, X_{p_q}) \leftarrow C_n(\mathbf{X}, (p_1, \dots, p_q))$. This requires circuitry of size S and depth D .
3. For each $j \in [q]$ in parallel, compute $x_{i_j} = \text{Sel}_{\{0,1\}}^{s \rightarrow 1}(X_{p_j}, (s_j))$. By Proposition 2, for each $j \in [q]$ this requires circuitry of size $O(s)$ and depth $O(\log s)$, for a total circuitry size of $O(q \cdot s)$ and depth $O(\log s)$.
4. Output $(x_{i_1}, \dots, x_{i_q})$. ◀

We can now prove our main theorem, which we recall here for convenience.

[Main Theorem] For all $n, q \in \mathbb{Z}^+$ there is a Boolean circuit computing $\text{Sel}^{n \rightarrow q}$ of size $O(n + q \cdot \log^3(n))$ and depth $O(\log(n + q))$.

Proof. Set $s(n) = \log^2 n$ and let $\Sigma = \{0, 1\}^s$. By Corollary 23, there exists a log-space uniform circuit for $\text{Sel}_{\Sigma}^{n/s \rightarrow q}$ with size $O(s \cdot n/s + q \cdot s \cdot \log n) = O(n + q \log^3 n)$ and depth $O(\log(n/s)) = O(\log n)$. Applying Lemma 27 to this circuit yields a log-space uniform circuit for $\text{Sel}_{\{0,1\}}^{n \rightarrow q}$ with size $O(n + q \cdot (\log^3 n + s)) = O(n + q \cdot \log^3 n)$ and depth $O(\log n + \log s) = O(\log n)$. ◀

References

- 1 M. Ajtai, J. Komlós, and E. Szemerédi. An $O(n \log n)$ sorting network. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, STOC '83, pages 1–9, New York, NY, USA, 1983. Association for Computing Machinery. doi:10.1145/800061.808726.
- 2 Alexander Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of π -scheme. *Moscow University Mathematics Bulletin*, 42(1):63–66, 1987.

- 3 Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Kartik Nayak, Enoch Peserico, and Elaine Shi. Optorama: Optimal oblivious RAM. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 403–432. Springer, 2020. doi:10.1007/978-3-030-45724-2_14.
- 4 Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Enoch Peserico, and Elaine Shi. Oblivious parallel tight compaction. In *ITC*, volume 163 of *LIPICs*, pages 11:1–11:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
- 5 Gilad Asharov, Wei-Kai Lin, and Elaine Shi. Sorting short keys in circuits of size $o(n \log n)$. *SIAM J. Comput.*, 51(3):424–466, 2022. doi:10.1137/20m1380983.
- 6 Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the servers computation in private information retrieval: PIR with preprocessing. In *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 55–73. Springer, 2000.
- 7 Norbert Blum. A boolean function requiring $3n$ network size. *Theor. Comput. Sci.*, 28:337–345, 1984. doi:10.1016/0304-3975(83)90029-4.
- 8 Elette Boyle, Yuval Ishai, Rafael Pass, and Mary Wootters. Can we access a database both locally and privately? In *TCC (2)*, volume 10678 of *Lecture Notes in Computer Science*, pages 662–693. Springer, 2017.
- 9 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *TCC (2)*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437. Springer, 2019.
- 10 Ran Canetti, Justin Holmgren, and Silas Richelson. Towards doubly efficient private information retrieval. In *TCC (2)*, volume 10678 of *Lecture Notes in Computer Science*, pages 694–726. Springer, 2017.
- 11 Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998. doi:10.1145/293347.293350.
- 12 Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.
- 13 Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2012.
- 14 Justin Holmgren and Ron D. Rothblum. Faster sounder succinct arguments and IOPs. In *CRYPTO (1)*, volume 13507 of *Lecture Notes in Computer Science*, pages 474–503. Springer, 2022.
- 15 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *STOC*, pages 262–271. ACM, 2004.
- 16 Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, pages 723–732. ACM, 1992.
- 17 É Sh Kospanov. Scheme realization of the sorting problem. *Diskretnyi Analiz i Issledovanie Operatsii*, 1(1):13–19, 1994.
- 18 Michal Koucký and Karel Král. Sorting short integers. In *ICALP*, volume 198 of *LIPICs*, pages 88:1–88:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.
- 19 Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 364–373. IEEE Computer Society, 1997. doi:10.1109/SFCS.1997.646125.
- 20 Richard E. Ladner and Michael J. Fischer. Parallel prefix computation. *J. ACM*, 27(4):831–838, October 1980. doi:10.1145/322217.322232.
- 21 Wei-Kai Lin, Ethan Mook, and Daniel Wichs. Doubly efficient private information retrieval and fully homomorphic ram computation from ring lwe. Cryptology ePrint Archive, Paper 2022/1703, 2022. URL: <https://eprint.iacr.org/2022/1703>.
- 22 Wei-Kai Lin and Elaine Shi. Optimal sorting circuits for short keys. *CoRR*, abs/2102.11489, 2021. arXiv:2102.11489.

- 23 È. I. Nechiporuk. A Boolean function. *Sov. Math., Dokl.*, 7:999–1000, 1966.
- 24 Wolfgang J. Paul. A $2.5n$ -lower bound on the combinational complexity of boolean functions. *SIAM J. Comput.*, 6(3):427–443, 1977. doi:10.1137/0206030.
- 25 Nicholas Pippenger. Self-routing superconcentrators. *J. Comput. Syst. Sci.*, 52(1):53–60, 1996. doi:10.1006/jcss.1996.0005.
- 26 Noga Ron-Zewi and Ron D. Rothblum. Proving as fast as computing: succinct arguments with constant prover overhead. In *STOC*, pages 1353–1363. ACM, 2022.
- 27 John E. Savage. *Models of computation - exploring the power of computing*. Addison-Wesley, 1998.
- 28 Leslie G. Valiant. On non-linear lower bounds in computational complexity. In William C. Rounds, Nancy Martin, Jack W. Carlyle, and Michael A. Harrison, editors, *Proceedings of the 7th Annual ACM Symposium on Theory of Computing, May 5-7, 1975, Albuquerque, New Mexico, USA*, pages 45–53. ACM, 1975. doi:10.1145/800116.803752.

A Applications

While we find the multiselection problem to be basic and natural, we additionally mention two concrete applications of the linear-sized multiselection circuit of Section 1 to problems in cryptography.

A.1 Application 1: Simplifying Efficient Arguments

Recently there has been a great deal of interest, both in theory and in practice, in developing efficient argument-systems (aka computationally sound proofs), proving for example that a given formula is satisfiable with a proof that is much shorter than the satisfying assignment (and is also very efficiently verifiable). A key bottleneck in such proof-systems is the efficiency of *proving* correctness, relative to the cost of merely computing the function.

Recent works [26, 14] consider the setting of Boolean circuits and construct provers whose size is *linear* in the size of the original computation. These works, following Kilian’s pioneering work [16], use a (generalization of a) PCP which is compiled into a succinct argument by sending a short hash of the PCP and then decommitting to desired locations that are sampled by the verifier. This naturally requires multiselection: the prover needs to restrict the PCP proof string to the selected indices. Thus, to get a linear-size prover, these works need a linear-size multiselection gadget.

Both [26] and [14] relied on ad-hoc solutions leveraging application-specific structure of the queries (i_1, \dots, i_q) and expended considerable effort to guarantee this structure. For example in the case of [14], a new local testing procedure was presented for tensor codes, with the novel property that the local testing queries were efficiently “multiselectable”.

Section 1 makes this work unnecessary by removing the burden of worrying about a particular query structure. This significantly simplifies these works (especially [14]) and is likely to simplify similar future works.

A.2 Application 2: More Efficient Batch PIR

Private information retrieval (PIR) [11, 19] is a process by which a client with an index $i \in [n]$ obtains an element x_i from a server with a database $\mathbf{x} \in \Sigma^n$ while (computationally) hiding all information about i from the server. *Batch PIR* has just one modification: the client has multiple indices i_1, \dots, i_q , and correspondingly obtains x_{i_1}, \dots, x_{i_q} . We call q the *batch size*. Thus the standard notion of PIR, which we also refer to as *non-batch PIR*, corresponds to the case $q = 1$. The *raison d’être* of batch PIR is that the server’s running time can be much less than q times the cost of non-batch PIR.

Indeed, our Section 1 implies the following corollary: if the ring learning with errors² (ring LWE) assumption holds, then for every batch size $q \leq n/\log^3 n$ and every constant $\epsilon > 0$, there is a batch PIR protocol in which:

- the server's running time is $n \cdot \text{polylog}(\lambda)$, where λ is a computational security parameter,
- the client-to-server communication is $(1 + \epsilon) \cdot q \cdot \log n + \text{poly}(\lambda)$,
- the server-to-client communication is $(1 + \epsilon) \cdot q + \text{poly}(\lambda)$, and
- the client's running time is $q \cdot \log n \cdot \text{polylog}(\lambda) + \text{poly}(\lambda)$.

In a nutshell, the idea is for the client to send an encryption ct_i of the indices $\mathbf{i} = (i_1, \dots, i_q)$ under a fully homomorphic encryption (FHE) scheme with the appropriate efficiency properties. The server, holding database $x \in \{0, 1\}^n$, homomorphically evaluates $\text{Sel}^{n \rightarrow q}(x, \cdot)$ (represented by the circuit of Section 1) on ct_i and sends the result to the client, which decrypts to obtain its output.

As for efficiency, homomorphic evaluation should satisfy two properties. First, the cost of homomorphically evaluating a circuit C should be $|C| \cdot \text{polylog}(\lambda)$ (at least when C is the circuit for $\text{Sel}^{n \rightarrow q}$ constructed in Section 1). Second, the ciphertexts resulting from homomorphic evaluation should have rate 1. If the ring LWE assumption holds, then one way to obtain such an FHE scheme is by combining the work of [13], which constructs FHE with $\text{polylog}(\lambda)$ evaluation overhead³, with the work of [9], which for any constant $\epsilon > 0$ constructs FHE in which evaluated ciphertexts have rate $1 - \epsilon$.

A.3 Prior Work

A.3.1 Batch PIR via Batch Codes

One major alternative approach to batch PIR is a reduction to non-batch PIR using *batch codes* [15], which encode a database $\mathbf{x} \in \{0, 1\}^n$ as a string $\mathbf{X} \in (\{0, 1\}^{N/m})^m$ such that to recover any q bits of \mathbf{x} , it suffices to read one bit from each N/m -bit block of \mathbf{X} . Here N , m , and q are parameters of the batch code. By retrieving each of these m bits with a non-batch PIR protocol, we obtain a batch PIR protocol with batch size q , server running time $\approx N$, and communication $\approx m$.

To replicate our batch PIR result via this approach, one would need batch codes with $m \approx k$ and $N \approx n$ for q at least $\omega(1)$. However, no such codes are known (see the table in Section 1.2 of [15]).

A.3.2 Doubly Efficient PIR.

A recent breakthrough result of Lin, Mook, and Wichs [21] shows how to achieve *doubly efficient PIR (DEPIR)* [6, 10, 8], i.e. PIR where the server's running time is sublinear in the database length (after a one-time deterministic preprocessing step), under the ring LWE assumption. Among other benefits, this allows the server's work to increase sublinearly with the number of queries, similarly to batch PIR. Amazingly, unlike in batch PIR, this is true even if the queries come from independent clients.

One can generically construct batch PIR protocols from DEPIR protocols, although as we now explain, our construction of batch PIR is more efficient than what one would obtain from [21]. Their DEPIR protocol exhibits a tunable trade-off between the server's online time and preprocessing time. They present two specific parameter settings:

² If we instead assume only standard LWE, we obtain a similar result but with all $\text{polylog}(\lambda)$ factors replaced by $\text{poly}(\lambda)$.

³ The work of [13] obtains this overhead only for circuits of width at least λ .

11:20 Linear-Size Boolean Circuits for Multiselection

- **(Online-Optimized)** The online time is $\text{polylog}(n) \cdot \text{poly}(\lambda)$, but the preprocessing time is $n^{1+\epsilon} \cdot \text{poly}(\lambda)$ for any constant $\epsilon > 0$.
- **(Preprocessing-Optimized)** The preprocessing time is $n \cdot 2^{\tilde{O}(\sqrt{\log n})} \cdot \text{poly}(\lambda)$, but the online time is $2^{\tilde{O}(\sqrt{\log n})} \cdot \text{poly}(\lambda)$.

In contrast to our batch PIR construction, their doubly efficient PIR server's *total* running time with either parameter setting never depends only linearly on n , and the server's per-query running time (including the amortized cost of pre-processing) is not polylogarithmic in n unless the number of queries is larger than the database.

B Linear-time Uniformity

In this section we briefly discuss an extension of our multiselection circuit that can be generated by a *linear-time* algorithm (rather than log-space uniformity as in Section 1). The specific notion of uniformity that we consider here is constructability by an algorithm in the standard word RAM model, that on input 1^n , runs in $O(n)$ time, using words of size $O(\log n)$ and a standard instruction set.




► **Theorem 28** (Linear-time Uniformity). *There exists a constant $\epsilon > 0$ such that for every $q = q(n) \leq n^\epsilon$, there exists a linear-time uniform Boolean circuit computing $\text{Sel}^{n \rightarrow q}$ of linear-size and logarithmic-depth.*

Proof Sketch. As a matter of fact, we show how one can generically take any n^c -time-uniform multiselection circuit for $c \geq 1$ (e.g., those of Section 1) and transform it into a *linear-time uniform* multiselection circuit (for a somewhat smaller number of queries), while preserving the linear-size and logarithmic depth.

The idea is to first generate a multiselection circuit C for input strings of length $n' = n^{1/c}$. By assumption, this can be done in time $O((n')^c) = O(n)$. The multiselection is then constructed as follows: we partition the input string $x \in \{0, 1\}^n$ into n' blocks of size $n/n' = n^{1-1/c}$ bits each. We then run $n^{1-1/c}$ copies of C , where the i -th copy is given the i -th bit of each of the blocks. The output of these circuit copies of C can be interpreted as q blocks in which our desired indices lie. We then apply a direct (uni-)selector to each block to obtain the desired bit. This step can be implemented by q circuits, each of size $O(n^{1-1/c})$. As long as $q = O(n^{1/c})$, the constructed circuit has linear-size and logarithmic depth.

To generate the circuit, our algorithm first generates the base multi-selector circuit which as noted above, takes time $O(n)$. Once that circuit is generated, creating the $n^{1-1/c}$ copies (with suitable input wiring) can be done in time $O(n^{1/c} \cdot n^{1-1/c}) = O(n)$. The additional circuitry can also be constructed in $O(q \cdot n^{1-1/c}) = O(n)$ time. ◀

A Subquadratic Upper Bound on Sum-Of-Squares Composition Formulas

Pavel Hrubeš   

Institute of Mathematics of ASCR, Prague, Czech Republic

Abstract

For every n , we construct a sum-of-squares identity

$$\left(\sum_{i=1}^n x_i^2\right)\left(\sum_{j=1}^n y_j^2\right) = \sum_{k=1}^s f_k^2,$$

where f_k are bilinear forms with complex coefficients and $s = O(n^{1.62})$. Previously, such a construction was known with $s = O(n^2/\log n)$. The same bound holds over any field of positive characteristic.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Sum-of-squares composition formulas, Hurwitz's problem, non-commutative arithmetic circuit

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.12

Funding *Pavel Hrubeš*: This work was supported by Czech Science Foundation GAČR grant 19-27871X.

1 Introduction

The problem of Hurwitz [8] asks for which integers n, m, s does there exist a sum-of-squares identity

$$(x_1^2 + \dots + x_n^2)(y_1^2 + \dots + y_m^2) = f_1^2 + \dots + f_s^2, \quad (1)$$

where f_1, \dots, f_s are bilinear forms in x and y with complex coefficients. Historically, the problem was motivated by existence of non-trivial identities with $n = m = s$. Starting with the obvious $x_1^2 y_1^2 = (x_1 y_1)^2$, the first remarkable identity is

$$(x_1^2 + x_2^2)(y_1^2 + y_2^2) = (x_1 y_1 - x_2 y_2)^2 + (x_1 y_2 + x_2 y_1)^2.$$

It can be interpreted as asserting multiplicativity of the norm on complex numbers. Euler's 4-square identity is an example with $n, m, s = 4$ which has later been interpreted as multiplicativity of the norm on quaternions. The final one is an 8-square identity which arises in connection to the algebra of octonions.

A classical result of Hurwitz [8] shows that these are the only cases: an identity (1) exists with $m, s = n$ iff $n \in \{1, 2, 4, 8\}$. An extension of this result is given by Hurwitz-Radon theorem [11]: an identity (1) exists with $s = n$ iff $m \leq \rho(n)$, where $\rho(n)$ is the Hurwitz-Radon number. The value of $\rho(n)$ is known exactly. For every n , $\rho(n) \leq n$ and equality is achieved only in the cases $n \in \{1, 2, 4, 8\}$. Asymptotically, $\rho(n)$ lies between $2 \log_2 n$ and $2 \log_2 n + 2$ if n is a power of 2. As shown in [12], Hurwitz-Radon theorem remains valid over any field of characteristic different from two. Hurwitz's problem is an intriguing question with connections to several branches of mathematics. We recommend D. Shapiro's monograph [13] on this subject.



© Pavel Hrubeš;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 12; pp. 12:1–12:11

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Let $\sigma(n)$ denote the smallest s such that an identity (1) with $m = n$ exists. While Hurwitz-Radon theorem solves the case $s = n$ *exactly*, even the *asymptotic* behavior of $\sigma(n)$ is not known. Elementary bounds¹ are $n \leq \sigma(n) \leq n^2$. Hurwitz's theorem implies that the first inequality is strict if n is sufficiently large. Using Hurwitz-Radon theorem, the upper bound can be improved to

$$\sigma(n) \leq O(n^2 / \log n).$$

As far as we are aware, this was the best asymptotic upper bound previously known. In this paper, we will improve it to a truly subquadratic bound

$$\sigma(n) \leq O(n^{1.62}). \tag{2}$$

A specific motivation for this problem comes from arithmetic circuit complexity. In [6], Wigderson, Yehudayoff and the current author related the sum-of-squares problem with complexity of non-commutative computations. Non-commutative arithmetic circuit is a model for computing polynomials whose variables do not multiplicatively commute. Since the seminal paper of Nisan [10], it has been an open problem to give a superpolynomial lower bound on circuit size in this model. In [6], it has been shown that a superlinear lower bound of $\Omega(n^{1+\epsilon})$ on $\sigma(n)$ translates to an exponential lower bound in the non-commutative setting. Hence, providing asymptotic lower bounds on Hurwitz's problem can be seen as a concrete approach towards answering Nisan's question. A more general, and hence less concrete, result of this flavor was given by Carmosino et al. in [1]. In an attempt to implement the sum-of-squares approach, the authors from [6] gave an $\Omega(n^{6/5})$ lower bound under the assumption that the identity (1) involves *integer* coefficients only [7]. However, the upper bound (2) goes in the opposite direction. Since it is superlinear, it does not immediately frustrate the approach from [6], it merely dampens its optimism.

2 The main result

Let \mathbb{F} be a field. Define $\sigma_{\mathbb{F}}(n, m)$ as the smallest s such that there exist bilinear² $f_1, \dots, f_s \in \mathbb{F}[x_1, \dots, x_n, y_1, \dots, y_m]$ satisfying (1). Furthermore, let $\sigma_{\mathbb{F}}(n) := \sigma_{\mathbb{F}}(n, n)$.

► **Theorem 1.** *Let \mathbb{F} be either \mathbb{C} or a field of positive characteristic. Then $\sigma_{\mathbb{F}}(n) \leq O(n^c)$ where $c < 1.62$.*

This will be proved in Section 4. In Section 5.1, we will give a modification of Theorem 1 that applies to any field.

► Remark 2.

- (i) If the field has characteristic two, Theorem 1 is trivial. Since $(\sum_i x_i^2)(\sum_j y_j^2) = (\sum_{i,j} x_i y_j)^2$, we have $\sigma_{\mathbb{F}}(n, m) = 1$.
- (ii) Instead of \mathbb{C} , the result holds also over Gaussian rationals $\mathbb{Q}(i)$.

Notation

Given vectors $u, v \in \mathbb{F}^n$, $\langle u, v \rangle := \sum_{i=1}^n u_i v_i$ is their inner product. For a set S , $\binom{S}{k}$ denotes the set of k -element subsets of S and $\binom{S}{\leq k}$ the set of subsets with at most k elements. $\binom{n}{\leq k} := \sum_{i=0}^k \binom{n}{i}$. $[n]$ is the set $\{1, \dots, n\}$.

¹ The former is obtained by substituting $(1, 0, \dots, 0)$ for the y variables, the latter by writing $(\sum x_i^2)(\sum y_j^2) = \sum_{i,j} (x_i y_j)^2$.

² Namely, of the form $\sum_{i,j} a_{i,j} x_i y_j$.

3 Hurwitz-Radon conditions

In this section, we give some well-known properties of σ that we will need later.

The definition immediately implies that $\sigma_{\mathbb{F}}(n, m)$ is symmetric, subadditive, and monotone:

$$\begin{aligned}\sigma_{\mathbb{F}}(n, m) &= \sigma_{\mathbb{F}}(m, n), \\ \sigma_{\mathbb{F}}(n, m_1 + m_2) &\leq \sigma_{\mathbb{F}}(n, m_1) + \sigma_{\mathbb{F}}(n, m_2), \\ \sigma_{\mathbb{F}}(n, m) &\leq \sigma_{\mathbb{F}}(n, m'), \quad m \leq m'.\end{aligned}\tag{3}$$

The following lemma gives a characterization of σ in terms of Hurwitz-Radon conditions (4). A proof can be found, e.g., in [13], but we present it for completeness.

► **Lemma 3.** *Let \mathbb{F} be a field of characteristic different from two. Then $\sigma_{\mathbb{F}}(n, m)$ equals the smallest s such that there exist matrices $H_1, \dots, H_m \in \mathbb{F}^{n \times s}$ satisfying*

$$\begin{aligned}H_i H_i^t &= I_n, \\ H_i H_j^t + H_j H_i^t &= 0, \quad i \neq j,\end{aligned}\tag{4}$$

for every $i, j \in [m]$.

Proof. Let f_1, \dots, f_s be bilinear polynomials in variables x_1, \dots, x_n and y_1, \dots, y_m . Then the vector $\bar{f} = (f_1, \dots, f_s)$ can be written as

$$\bar{f} = \sum_{i=1}^n \bar{x} H_i y_i,$$

where $\bar{x} = (x_1, \dots, x_n)$ and $H_i \in \mathbb{F}^{n \times s}$. Hence

$$\sum_{k=1}^s f_k^2 = \bar{f} \bar{f}^t = \sum_i y_i^2 \bar{x} H_i H_i^t \bar{x}^t + \sum_{i < j} y_i y_j \bar{x} (H_i H_j^t + H_j H_i^t) \bar{x}^t.$$

If the matrices satisfy (4), this equals $\sum_i y_i^2 \bar{x} I_n \bar{x}^t = (y_1^2 + \dots + y_m^2)(x_1^2 + \dots + x_n^2)$, which gives a sum-of-squares identity with s squares. Conversely, if $(y_1^2 + \dots + y_m^2)(x_1^2 + \dots + x_n^2) = \sum f_k^2$, we must have $\bar{x} H_i H_i^t \bar{x}^t = x_1^2 + \dots + x_n^2$ and $\bar{x} (H_i H_j^t + H_j H_i^t) \bar{x}^t = 0$. In characteristic different from 2, this is possible only if the conditions (4) are satisfied. ◀

Given a natural number of the form $n = 2^k a$ where a is odd, the Hurwitz-Radon number is defined as

$$\rho(n) = \begin{cases} 2k + 1, & \text{if } k = 0 \\ 2k, & \text{if } k = 1 \\ 2k, & \text{if } k = 2 \\ 2k + 2, & \text{if } k = 3 \end{cases} \pmod{4}$$

Observe that

$$2 \log_2 n \leq \rho(n) \leq 2 \log_2(n) + 2,$$

whenever n is a power of two.

Square matrices A_1, A_2 *anticommute* if $A_1 A_2 = -A_2 A_1$. A family of square matrices A_1, \dots, A_t will be called *anticommuting* if A_i, A_j anticommute for every $i \neq j$.

The following lemma is a key ingredient in the proof of Hurwitz-Radon theorem. A self-contained construction can be found in [2].

12:4 A Subquadratic Upper Bound on Sum-Of-Squares Composition Formulas

► **Lemma 4.** *For every n , there exists an anticommuting family of $t = \rho(n) - 1$ integer matrices $e_1, \dots, e_t \in \mathbb{Z}^{n \times n}$ which are orthonormal and antisymmetric (i.e., $e_i e_i^t = I_n$ and $e_i = -e_i^t$).*

► **Remark 5.** A straightforward construction (see, e.g., [5]) gives an anticommuting family of $t = 2 \log_2 n + 1$ integer matrices $e_1, \dots, e_t \in \mathbb{Z}^{n \times n}$ with $e_i^2 = \pm I_n$ whenever n is a power of two. With minor modifications, these matrices could be used in the subsequent construction instead.

4 The construction

Let e_1, \dots, e_t be a set of square matrices. Given $A = \{i_1, \dots, i_k\} \subseteq [t]$ with $i_1 < \dots < i_k$, let $e_A := \prod_{j=1}^k e_{i_j}$.

► **Lemma 6.** *Let e_1, \dots, e_t be a set of anticommuting matrices. If $A, B \subseteq [t]$ have even size (resp. odd size) then e_A, e_B anticommute assuming $|A \cap B|$ is odd (resp. even).*

Proof. Since e_i anticommutes with every e_j , $j \neq i$, but commutes with itself, we obtain

$$e_A e_i = (-1)^{|A \setminus \{i\}|} e_i e_A.$$

This implies that

$$e_A e_B = (-1)^q e_B e_A,$$

where $q = |A| \cdot |B| - |A \cap B|$. Hence if A, B are even (resp. odd) and their intersection is odd (resp. even), q is odd and e_A, e_B anticommute. ◀

Given integers $0 \leq k \leq t$, a (k, t) -parity representation of dimension s over a field \mathbb{F} is a map $\xi : \binom{[t]}{k} \rightarrow \mathbb{F}^s$ such that for every $A, B \in \binom{[t]}{k}$

$$\begin{aligned} \langle \xi(A), \xi(A) \rangle &= 1, \\ \langle \xi(A), \xi(B) \rangle &= 0, \text{ if } A \neq B \text{ and } (|A \cap B| = k \pmod{2}). \end{aligned} \quad (5)$$

► **Lemma 7.** *Let $0 \leq k \leq t$. Over \mathbb{C} , there exists a (k, t) -parity representation of dimension $\binom{t}{\leq \lfloor k/2 \rfloor}$. If \mathbb{F} is a field of odd characteristic p , there exists a (k, t) -parity representation of dimension $(p-1) \binom{t}{\leq \lfloor k/2 \rfloor}$.*

The case of odd characteristic will be proved in the Appendix.

Proof of Lemma 7 over \mathbb{C} . Let $0 \leq k \leq t$ be given and $d := \lfloor k/2 \rfloor$.

For $a \in \{0, 1\}^t$, let $|a|$ be the number of ones in a . Recall that a polynomial is multilinear, if every variable in it has individual degree at most one. We first observe:

▷ **Claim 8.** There exists a multilinear polynomial $f \in \mathbb{Q}(x_1, \dots, x_t)$ of degree at most d such that for every $a \in \{0, 1\}^t$

$$f(a) = \begin{cases} 1, & \text{if } |a| = k \\ 0, & \text{if } |a| < k \text{ and } (|a| = k \pmod{2}). \end{cases} \quad (6)$$

Proof of Claim. Consider the polynomial

$$g(x_1, \dots, x_t) := c \prod_{0 \leq i < k, i = k \pmod{2}} \left(\sum_{j=1}^t x_j - i \right).$$

Then g has degree d and we can choose $c \in \mathbb{Q}$ so that g satisfies (6). Since we care about inputs from $\{0, 1\}^t$, g can be rewritten as a multilinear polynomial f of degree at most d . \triangleleft

Since f is multilinear, we can write it as

$$f(x_1, \dots, x_t) = \sum_{C \in \binom{[t]}{\leq d}} \alpha_C \prod_{i \in C} x_i,$$

where α_C are rational coefficients. Identifying a subset A of $[t]$ with its characteristic vector in $\{0, 1\}^t$, we have

$$f(A) = \sum_{C \subseteq A} \alpha_C.$$

Let $s := \binom{t}{\leq d}$. Given $A \in \binom{[t]}{k}$, let $\xi(A) \in \mathbb{C}^s$ be the vector whose coordinates are indexed by subsets $C \in \binom{[t]}{\leq d}$ such that

$$\xi(A)_C = \begin{cases} (\alpha_C)^{1/2}, & \text{if } C \subseteq A \\ 0, & \text{if } C \not\subseteq A. \end{cases}$$

This guarantees

$$\langle \xi(A), \xi(B) \rangle = \sum_C \xi(A)_C \xi(B)_C = \sum_{C \subseteq A \cap B} \alpha_C = f(A \cap B).$$

Hence conditions (6) translate to the desired properties of the map ξ . \blacktriangleleft

Combining Lemma 6 and 7, we obtain the following bound on σ :

► Theorem 9. *Let n be a non-negative integer. Let $0 \leq k \leq \rho(n) - 1$ and $m := \binom{\rho(n)-1}{k}$. Then*

$$\sigma_{\mathbb{C}}(n, m) \leq n \cdot \binom{\rho(n) - 1}{\leq \lfloor k/2 \rfloor}.$$

If \mathbb{F} is a field of odd characteristic p then

$$\sigma_{\mathbb{F}}(n, m) \leq (p-1)n \cdot \binom{\rho(n) - 1}{\leq \lfloor k/2 \rfloor}.$$

Proof. Let n, k, m be as in the assumption. Let e_1, \dots, e_t be the matrices from Lemma 4 with $t = \rho(n) - 1$. Let ξ be the (k, t) -parity representation given by the previous lemma. For $A \in \binom{[t]}{k}$, let

$$H_A := e_A \otimes \xi(A),$$

where e_A is defined as in Lemma 6, $\xi(A)$ is viewed as a row vector, and \otimes is the Kronecker (tensor) product.

Note that each H_A has dimension $n \times (ns)$ where s is the dimension of the parity representation, and there are $m = \binom{t}{k}$ such matrices H_A . By Lemma 3, it is sufficient to show that the system of matrices $H_A, A \in \binom{[t]}{k}$, satisfies Hurwitz-Radon conditions (4).

We have

$$H_A H_B^t = (e_A e_B^t) \otimes (\xi(A) \xi(B)^t) = \langle \xi(A), \xi(B) \rangle \cdot e_A e_B^t.$$

12:6 A Subquadratic Upper Bound on Sum-Of-Squares Composition Formulas

Since every e_i is orthonormal, we have $e_A e_A^t = I_n$. From (5), we have $\langle \xi(A), \xi(A) \rangle = 1$ and hence

$$H_A H_A^t = I_n .$$

If $A \neq B$ then

$$H_A H_B^t + H_B H_A^t = \langle \xi(A), \xi(B) \rangle \cdot (e_A e_B^t + e_B e_A^t) . \quad (7)$$

If $|A \cap B| = k \bmod 2$ then $\langle \xi(A), \xi(B) \rangle = 0$ by (5) and hence (7) equals zero. If $|A \cap B| \neq k \bmod 2$ then $e_A e_B^t + e_B e_A^t = 0$. This is because $e_A e_B = -e_B e_A$ by Lemma 6 and that, since e_i are antisymmetric, e_A, e_B are either both symmetric or both antisymmetric. Therefore (7) equals zero for every $A \neq B \in \binom{[t]}{k}$. ◀

Theorem 1 is an application of Theorem 9.

Proof of Theorem 1. Assume first that n is a power of 16. This gives $\rho(n) = 2 \log_2(n) + 1$. Let k be the smallest integer with $n \leq \binom{2 \log_2 n}{k} =: m$. From the previous theorem and monotonicity of σ (cf. (3)), we obtain

$$\sigma_{\mathbb{F}}(n) \leq \sigma_{\mathbb{F}}(n, m) \leq cns ,$$

where the constant c depends on the field only and $s := \binom{2 \log_2 n}{\leq \lfloor k/2 \rfloor}$.

We have $k = 2(\alpha + \epsilon_n) \log_2 n$ where $\alpha \in (0, \frac{1}{2})$ is such that $H(\alpha) = 1/2$ (H is the binary entropy function) and $\epsilon_n \rightarrow 0$ as n approaches infinity. We also have

$$s \leq 2^{2H(\frac{\alpha + \epsilon_n}{2}) \log_2 n} = n^{2H(\frac{\alpha}{2}) + \epsilon'_n} ,$$

where $\epsilon'_n \rightarrow 0$. Hence

$$\sigma_{\mathbb{F}}(n) \leq cn^{1+2H(\frac{\alpha}{2}) + \epsilon'_n} .$$

The numerical value of α is $0.11 \dots$ which leads to $\sigma_{\mathbb{F}}(n) \leq cn^{1.615 + \epsilon'_n} \leq O(n^{1.616})$.

If n is not a power of 16, take n' with $n < n' < 16n$ which is. By monotonicity of σ , we have $\sigma_{\mathbb{F}}(n) \leq \sigma_{\mathbb{F}}(n')$. ◀

4.1 Comments

► Remark 10.

- (i) Instead of \mathbb{C} , the proof of Theorem 9 applies to any field where all rationals have a square root. However, Theorem 1 holds also over Gaussian rationals $\mathbb{Q}(i)$ (cf. Section 5.1).
- (ii) In positive characteristic, the bounds in Lemma 7 and Theorem 9 can sometimes be improved: if $\mathbb{F} \supseteq \mathbb{F}_{p^2}$, the factor $(p-1)$ can be dropped. For certain values of k , $\binom{t}{\leq \lfloor k/2 \rfloor}$ can be replaced with $\binom{t}{\lfloor k/2 \rfloor}$ (cf. Remark 19).

An improvement on the dimension of parity representation in Lemma 7, if possible, will lead to an improvement in Theorem 1. However, this dimension cannot be too small:

► Remark 11. If k is even, every (k, t) -parity representation must have dimension at least $s = \binom{\lfloor t/2 \rfloor}{\lfloor k/2 \rfloor}$ over any field. This is because there exists a family \mathcal{A} of k -element subsets of $[t]$ whose pairwise intersection is even, and $|\mathcal{A}| = s$. The map ξ must assign linearly independent vectors to elements of \mathcal{A} . Similarly for k odd.

On the other hand, $\binom{t}{\lfloor k/2 \rfloor}$ in Lemma 7 can be replaced with $\binom{t}{\lfloor t-k/2 \rfloor}$ which gives a smaller bound if $k > t/2$. This is because we can instead work with complements of $A \in \binom{[t]}{k}$.

The notion of (k, t) -parity representation can be restated in the language of *orthonormal representations* of graphs of Lovász [9]. Given a graph G with vertex set V , its orthonormal representation is a map $\xi(V) : \rightarrow \mathbb{F}^s$ such that for every $u, v \in V$

$$\begin{aligned} \langle \xi(u), \xi(u) \rangle &= 1, \\ \langle \xi(u), \xi(v) \rangle &= 0, \text{ if } u \neq v \text{ are not adjacent in } G. \end{aligned}$$

In this language, (k, t) -parity representation is an orthonormal representation of the following combinatorial Knesner-type graph $G_{k,t}$: vertices of $G_{k,t}$ are k -element subsets of $[t]$. There is an edge between u and v iff $|u \cap v| \neq k \bmod 2$. Orthogonal representations of related graphs have been studied by Haviv in [4, 3].

5 Modifications and extensions

5.1 A sum of bilinear products

Define $\beta_{\mathbb{F}}(n)$ as the smallest s such there exists an identity

$$(x_1^2 + \dots + x_n^2)(y_1^2 + \dots + y_n^2) = f_1 f'_1 + \dots + f_s f'_s, \tag{8}$$

where f_1, \dots, f_s and f'_1, \dots, f'_s are bilinear forms with coefficients from \mathbb{F} .

We have $\beta_{\mathbb{F}}(n) \leq \sigma_{\mathbb{F}}(n)$. In some contexts, β is a more natural quantity than σ . In this section, we give a modification of Theorem 1 in terms of β :

► **Theorem 12.** *Over any field, $\beta_{\mathbb{F}}(n) \leq O(n^c)$ where $c < 1.62$.*

► **Remark 13.** In characteristic different from two, we have $ff' = \left(\frac{f+f'}{2}\right)^2 - \left(\frac{f-f'}{2}\right)^2$, which allows to rewrite (8) as

$$(x_1^2 + \dots + x_n^2)(y_1^2 + \dots + y_n^2) = g_1^2 + \dots + g_s^2 - h_1^2 - \dots - h_s^2.$$

It follows that

$$\begin{aligned} \sigma_{\mathbb{F}}(n) &\leq 2\beta_{\mathbb{F}}(n), \text{ if } \mathbb{F} \text{ contains a square root of } -1, \\ \sigma_{\mathbb{F}}(n) &\leq p\beta_{\mathbb{F}}(n), \text{ if } \mathbb{F} \text{ has characteristic } p > 0. \end{aligned}$$

We conclude that, first, Theorem 1 is a consequence of Theorem 12 and, second, Theorem 1 holds also over Gaussian rationals $\mathbb{Q}(i)$.

The proof of Theorem 12 is a straightforward modification of that of Theorem 1 and we only highlight the main points.

The following is an analogy of Lemma 3:

► **Lemma 14.** *Assume that there are matrices $H_1, \dots, H_m, \tilde{H}_1, \dots, \tilde{H}_m \in \mathbb{F}^{n \times s}$ satisfying*

$$H_i \tilde{H}_i^t = I_n, \quad H_i \tilde{H}_j^t + H_j \tilde{H}_i^t = 0, \quad i \neq j,$$

for every $i, j \in [m]$. Then $\beta_{\mathbb{F}}(n, m) \leq s$.

Proof. Define

$$(f_1, \dots, f_s) = \sum_{i=1}^n \bar{x} H_i y_i, \quad (f'_1, \dots, f'_s) = \sum_{i=1}^n \bar{x} \tilde{H}_i y_i.$$

Hence

$$\sum_{k=1}^s f_k f'_k = (f_1, \dots, f_s)(f'_1, \dots, f'_s)^t = \sum_i y_i^2 \bar{x} H_i \tilde{H}_i^t \bar{x}^t + \sum_{i < j} y_i y_j \bar{x} (H_i \tilde{H}_j^t + H_j \tilde{H}_i^t) \bar{x}^t.$$

This equals $\sum_i y_i^2 \bar{x} I_n \bar{x}^t = (y_1^2 + \dots + y_n^2)(x_1^2 + \dots + x_n^2)$ as required. \blacktriangleleft

► Lemma 15. For $0 \leq k \leq t$ and any field \mathbb{F} of characteristic different from two, there exists a pair of maps $\xi, \tilde{\xi} : \binom{[t]}{k} \rightarrow \mathbb{F}^s$ with $s = \binom{t}{\lfloor k/2 \rfloor}$ such that for every $A, B \in \binom{[t]}{k}$

$$\begin{aligned} \langle \xi(A), \tilde{\xi}(A) \rangle &= 1, \\ \langle \xi(A), \tilde{\xi}(B) \rangle &= \langle \xi(B), \tilde{\xi}(A) \rangle, \\ \langle \xi(A), \tilde{\xi}(B) \rangle &= 0, \text{ if } A \neq B \text{ and } (|A \cap B| = k \bmod 2). \end{aligned}$$

Proof. The proof is almost the same as that of Lemma 7. Equipped with the polynomial f from Claim 8 or Lemma 17, it is sufficient to modify the definition of ξ as follows:

$$\xi(A)_C = \begin{cases} \alpha_C, & \text{if } C \subseteq A \\ 0, & \text{if } C \not\subseteq A. \end{cases}, \quad \tilde{\xi}(A)_C = \begin{cases} 1, & \text{if } C \subseteq A \\ 0, & \text{if } C \not\subseteq A. \end{cases} \quad \blacktriangleleft$$

Proof sketch of Theorem 12. In Theorem 9, replace the matrices H_A by the pair

$$H_A := e_A \otimes \xi(A), \quad \tilde{H}_A = e_A \otimes \tilde{\xi}(A).$$

They satisfy the conditions from Lemma 14 and we can proceed as in Theorem 1. \blacktriangleleft

5.2 A tensor product construction

We now outline an alternative construction of non-trivial sum-of-squares identities. While it gives different types of identities, it does not seem to give better bounds asymptotically.

Instead of the products of anticommuting matrices e_A , one can take the *tensor* product of matrices satisfying Hurwitz-Radon conditions (4). Namely, given such matrices $H_1, \dots, H_m \in \mathbb{F}^{n \times s}$, and $a \in [m]^\ell$, let

$$H_a := H_{a_1} \otimes H_{a_2} \cdots \otimes H_{a_\ell}.$$

Observe that every H_a satisfies $H_a H_a^t = I_{n^\ell}$ and that

$$H_a H_b^t + H_b H_a^t = 0,$$

whenever a and b have *odd* Hamming distance (i.e., they differ in an odd number of coordinates). As in Lemma 7, we can find a map $\xi : [m]^\ell \rightarrow \mathbb{C}^s$ with $s \leq (4m)^{\ell/2}$ such that

$$\begin{aligned} \langle \xi(a), \xi(a) \rangle &= 1, \\ \langle \xi(a), \xi(b) \rangle &= 0, \text{ if } a \neq b \text{ have even Hamming distance.} \end{aligned}$$

This gives for every ℓ

$$\sigma_{\mathbb{C}}(n^\ell, m^\ell) \leq \sigma_{\mathbb{C}}(n, m)^\ell (4m)^{\ell/2}$$

For example, starting with $\sigma_{\mathbb{C}}(8, 8) = 8$, we have

$$\sigma_{\mathbb{C}}(8^\ell, 8^\ell) \leq 8^{11\ell/6}.$$

6 Open problems

Let Even_t denote the set of even-sized subsets of $[t]$. A map $\xi : \text{Even}_t \rightarrow \mathbb{F}^s$ will be called a t -parity representation of dimension s if for every $A, B \in \text{Even}_t$

$$\begin{aligned}\langle \xi(A), \xi(A) \rangle &= 1, \\ \langle \xi(A), \xi(B) \rangle &= 0, \text{ if } A \neq B \text{ and } |A \cap B| \text{ is even.}\end{aligned}$$

► **Problem 1.** Over \mathbb{C} , does there exist a t -parity representation of dimension $2^{(0.5+o(1))t}$?

If this were the case, we could improve the bound of Theorem 1 to $\sigma_{\mathbb{C}}(n, n) \leq n^{1.5+o(1)}$. A more surprising consequence would be that

$$\sigma_{\mathbb{C}}(n, n^2) \leq n^{2+o(1)}.$$

The constant 0.5 in Problem 1 cannot be improved: since there exists a family of $2^{\lfloor t/2 \rfloor}$ subsets of $[t]$ with pairwise even intersection, every t -parity representation must have dimension at least $2^{\lfloor t/2 \rfloor}$ (cf. Remark 11). On the other hand, Lemma 7 implies that there exists a t -parity representation of dimension at most $2^{(H(0.25)+o(1))t} < 2^{0.82t}$.

Our results do not apply to sum-of-squares composition formulas over the real numbers. Since \mathbb{R} is one of the most natural choices of the underlying field, it is desirable to extend the construction in this direction. This motivates the following:

► **Problem 2.** Over \mathbb{R} , does there exist a t -parity representation of dimension $O(2^{ct})$ with $c < 1$?

References

- 1 Marco L. Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness amplification for non-commutative arithmetic circuits. In *Proceedings of the 33rd Computational Complexity Conference, CCC '18*, 2018.
- 2 A. Geramita and N. Pullman. A theorem of Hurwitz and Radon and orthogonal projective modules. *Proceedings of The American Mathematical Society*, 42:51–51, January 1974. doi: 10.1090/S0002-9939-1974-0332764-4.
- 3 I. Haviv. On minrank and the Lovász Theta function. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2018.
- 4 I. Haviv. Topological bounds on the dimension of orthogonal representations of graphs. *European Journal of Combinatorics*, 81:84–97, 2019.
- 5 P. Hrubeš. On families of anticommuting matrices. *Linear Algebra and Applications*, 493:494–507, 2016.
- 6 P. Hrubeš, A. Wigderson, and A. Yehudayoff. Non-commutative circuits and the sum of squares problem. In *STOC' 10 Proceedings of the 42nd symposium on Theory of Computing*, pages 667–676, 2010.
- 7 P. Hrubeš, A. Wigderson, and A. Yehudayoff. An asymptotic bound on the composition number of integer sums of squares formulas. *Canadian Mathematical Bulletin*, 56:70–79, 2013.
- 8 A. Hurwitz. Über die Komposition der quadratischen Formen von beliebigvielen Variablen. *Nach. Ges. der Wiss. Göttingen*, pages 309–316, 1898.
- 9 L. Lovász. On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(1):1–7, 1979.
- 10 N. Nisan. Lower bounds for non-commutative computation. In *Proceeding of the 23th STOC*, pages 410–418, 1991.
- 11 J. Radon. Lineare scharen orthogonalen Matrizen. *Abh. Math. Sem. Univ. Hamburg*, 1(2-14), 1922.
- 12 D. B. Shapiro. Quadratic forms and similarities. *Bull. Amer. Math. Soc.*, 81(6), 1975.
- 13 D. B. Shapiro. *Compositions of quadratic forms*. De Gruyter expositions in mathematics 33, 2000.

A Proof of Lemma 7 in positive characteristic

Given non-negative integers $\bar{n} = (n_1, \dots, n_d)$ let $B(\bar{n})$ be the $d \times d$ matrix $\{B(\bar{n})_{i,j}\}_{i,j \in [d]}$ with

$$B(\bar{n})_{i,j} = \binom{n_j}{i-1}.$$

We assume that $\binom{n}{k} = 0$ whenever $n < k$; this guarantees $\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k!}$.

► **Lemma 16.** *If $\bar{n} = (r, r+2, \dots, r+2(d-1))$ for some non-negative integer r then $\det(B(\bar{n})) = 2^{\binom{d}{2}}$.*

Proof. We claim that

$$\det(B(\bar{n})) = \left(\prod_{i=1}^{d-1} i! \right)^{-1} \det(V(\bar{n})),$$

where $V(\bar{n})$ is the Vandermonde matrix with entries $V(\bar{n})_{i,j} = n_j^{i-1}$. To see this, multiply every i -th row of $B(\bar{n})$ by $(i-1)!$ to obtain matrix $B'(\bar{n})$ with

$$\det(B'(\bar{n})) = \left(\prod_{i=1}^{d-1} i! \right) \det(B(\bar{n})).$$

An i -th row r_i of $B'(\bar{n})$ is of the form $(n_1^{i-1} + g_i(n_1), \dots, n_d^{i-1} + g_i(n_d))$ where g_i is a polynomial of degree $< (i-1)$. This means that r_i equals the i -th row of $V(\bar{n})$ plus a suitable linear combination of the preceding rows of $V(\bar{n})$. Therefore, $\det(B'(\bar{n})) = \det(V(\bar{n}))$.

Given \bar{n} as in the assumption, we obtain

$$\begin{aligned} \det(V(\bar{n})) &= \prod_{1 \leq j_1 < j_2 \leq d} (n_{j_2} - n_{j_1}) = \prod_{1 \leq j_1 < j_2 \leq d} (2j_2 - 2j_1) \\ &= 2^{\binom{d}{2}} \prod_{1 \leq j_1 < j_2 \leq d} (j_2 - j_1) = 2^{\binom{d}{2}} \prod_{i=1}^{d-1} i!. \end{aligned}$$

This shows that $\det(B(\bar{n})) = 2^{\binom{d}{2}}$. ◀

► **Lemma 17.** *Let p be an odd prime. Given $0 \leq k \leq t$, there exists a multilinear polynomial $f \in \mathbb{F}_p(x_1, \dots, x_t)$ of degree at most $d = \lfloor k/2 \rfloor$ such that for every $a \in \{0, 1\}^t$*

$$f(a) = \begin{cases} 1, & \text{if } |a| = k \\ 0, & \text{if } |a| < k \text{ and } (|a| = k \bmod 2). \end{cases}$$

Proof. We look for f of the form $f = \sum_{j=0}^d c_j S_t^j$ where S_t^j is the elementary symmetric polynomial $S_t^j = \sum_{|A|=j} \prod_{i \in A} x_i$. Given $a \in \{0, 1\}^t$,

$$f(a) = \sum_{j=0}^d c_j \binom{|a|}{j} \bmod p.$$

We are therefore looking for a solution of the linear system

$$B(\bar{n}) (c_0, \dots, c_d)^t = (0, \dots, 0, 1)^t,$$

where $\bar{n} = (0, 2, \dots, 2d)$, if k is even, and $\bar{n} = (1, 3, \dots, 2d+1)$, if k is odd. By the previous lemma, $B(\bar{n})$ is invertible over \mathbb{F}_p and such a solution exists. ◀

► **Lemma 18.** *If \mathbb{F} is a field of odd characteristic p , there exists a (k, t) -parity representation of dimension $(p-1)\binom{t}{\lfloor k/2 \rfloor}$.*

Proof. If every element of \mathbb{F}_p has a square root in \mathbb{F} , the proof is the same as over \mathbb{C} . In general, proceed as follows. Since every non-zero element of \mathbb{F}_p is a sum of at most $(p-1)$ ones, we can write

$$f(x_1, \dots, x_t) = \sum_{C \in \mathcal{C}} \prod_{i \in C} x_i,$$



where \mathcal{C} is a multiset of $s \leq (p-1)\binom{t}{\lfloor k/2 \rfloor}$ subsets of $[t]$. For $A \in \binom{[t]}{k}$, let $\xi(A) \in \mathbb{F}^s$ be a vector whose coordinates are indexed by elements C of \mathcal{C} so that

$$\xi(A)_C = \begin{cases} 1, & \text{if } C \subseteq A \\ 0, & \text{if } C \not\subseteq A. \end{cases} \quad \blacktriangleleft$$

► **Remark 19.**

- (i) Over \mathbb{F}_{p^2} or a larger field, the factor of $(p-1)$ in Lemma 18 can be dropped. This is because every element of \mathbb{F}_p has a square root in \mathbb{F}_{p^2} .
- (ii) For specific values of k , a stronger bound is possible. For example, if $k = 2p^\ell - 1$, there is a (k, t) -parity representation of dimension $\binom{t}{\lfloor k/2 \rfloor}$. It follows from Lucas' theorem that in this case, f in Lemma 17 can be taken simply as the elementary symmetric polynomial of degree $\lfloor k/2 \rfloor$. This polynomial has only $\binom{t}{\lfloor k/2 \rfloor}$ monomials.

Hard Submatrices for Non-Negative Rank and Communication Complexity

Pavel Hrubeš   

Institute of Mathematics of ASCR, Prague, Czech Republic

Abstract

Given a non-negative real matrix M of non-negative rank at least r , can we witness this fact by a small submatrix of M ? While Moitra (SIAM J. Comput. 2013) proved that this cannot be achieved exactly, we show that such a witnessing is possible approximately: an $m \times n$ matrix of non-negative rank r always contains a submatrix with at most r^3 rows and columns with non-negative rank at least $\Omega(\frac{r}{\log n \log m})$. A similar result is proved for the 1-partition number of a Boolean matrix and, consequently, also for its two-player deterministic communication complexity. Tightness of the latter estimate is closely related to the log-rank conjecture of Lovász and Saks.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Non-negative rank, communication complexity, extension complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.13

Funding *Pavel Hrubeš*: This work was supported by Czech Science Foundation GAČR grant 19-27871X.

Acknowledgements The author wants to thank Anup Rao for useful comments.

1 Introduction

The rank of a matrix is one of the most versatile concepts from linear algebra. A basic property of matrix rank is the following: if a matrix M has rank at least r then it contains an $r \times r$ submatrix of rank r . Put differently, the fact that $\text{rk}(M) \geq r$ can be witnessed by a hard $r \times r$ submatrix. Can we extend this witnessing property to other matrix complexity measures? We will consider two such measures: the *non-negative rank* of a non-negative real matrix and the *1-partition number* of a Boolean matrix.

Given a matrix with non-negative real entries, its non-negative rank is defined similarly to rank, except that we want to express the matrix as a sum of non-negative rank-one matrices. This quantity has numerous applications in communication complexity and linear optimization [20], and other fields (cf. [15]). In [20], Yannakakis has discovered a geometric interpretation of non-negative rank in terms of linear projections of polytopes. This connection has been extended and exploited in many subsequent results, see, e.g., [18, 2, 5], including the current paper.

If M is a 0, 1-matrix, its 1-partition number can be defined as the smallest r such that M can be written as a sum of r rank-one Boolean matrices. This is an important concept in communication complexity [11, 16]. Interpreting a 0, 1-matrix as the adjacency matrix of a bipartite graph, it is also equivalent to the *biclique partition number* (see [3] and references within).

If M has non-negative rank $\geq r$, can this fact be witnessed by a small submatrix? The short answer is no. In [15], Moitra presented an $n \times n$ matrix M of non-negative rank 4 such that every submatrix with less than $n/3$ columns has non-negative rank at most 3 – in particular, M contains no constant-size submatrix of non-negative rank 4. In Section 6.3, we will give a different example where the gap is more dramatic. Similarly, we will see that the most optimistic form of witnessing fails for 1-partition number. On the positive side, we will



© Pavel Hrubeš;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 13; pp. 13:1–13:12
Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



show that a weaker form of witnessing nevertheless holds: if a matrix has non-negative rank r then it contains a submatrix of size bounded by a polynomial in r whose non-negative rank is close to r ; similarly for 1-partition number.

The two-player deterministic communication complexity of M can be characterized by the logarithm of the 1-partition number of M . Hence our witnessing result for 1-partition number can be restated in the language of communication complexity: if a Boolean function has a large communication complexity, this fact can be approximately witnessed by a relatively small set of inputs. It should be noted that this statement immediately follows from the log-rank conjecture of Lovász and Saks (presented in [14]). This conjecture relates the communication complexity of a Boolean matrix with its rank. It implies that for a Boolean matrix M , the three parameters – rank, 1-partition number, non-negative rank – are essentially the same, with their logarithm being polynomially related to the communication complexity of M . This allows us deduce a witnessing property for these measures from the witnessing property of matrix rank. Our result to confirms this prediction of the conjecture and it may therefore be interpreted as a vote in its favor. On the other hand, the log-rank conjecture implies a stronger form of witnessing than what we actually prove. Hence, in principle, a counterexample to the conjecture may be given by a matrix for which this predicted form of witnessing fails (see Section 5 for more details). According to [6], the witnessing problem for communication complexity has been previously posed by H. Halem.

Our witnessing results could be easily converted to non-trivial *approximation* algorithms to compute non-negative rank or the 1-partition number. These algorithms would run in polynomial time whenever the complexity parameter in question is fixed. Interestingly, *exact* algorithms of this form were given by Moitra [15] and Chandran et al. [3]. While there are similarities between these algorithms and the witnessing perspective, these algorithms ultimately do not search for a witness.

On a more abstract level, the witnessing problem can be posed with respect to any complexity measure whatsoever. A related result in Boolean circuit complexity are “antichackers” of Lipton and Young [13]. In their work, it is shown that if a Boolean function f requires a Boolean circuit of size s then there is a subset of inputs of size roughly s such that f restricted to this subset still requires circuit size roughly s . A related topic are “hard-core predicates” of Impagliazzo [10]. Recently, Göös et al. [6] studied deterministic query complexity from this perspective. An example from the opposite side of the spectrum is the chromatic number of a graph. It is known that a large chromatic number imposes almost no local structure on a graph and cannot be witnessed by a small subgraph [4, 17].

2 Main results

Given an $m \times n$ matrix M with real non-negative entries, its *non-negative rank*, $\text{rk}_+(M)$, is the smallest s such that M can be written as

$$M = LR,$$

where L and R are non-negative matrices of dimensions $m \times s$ and $s \times n$, respectively.

We will show that every M with large non-negative rank contains a relatively small submatrix of large non-negative rank.

► **Theorem 1.** *Let M be an $m \times n$ non-negative real matrix with $n \geq 2$. Then for every $k \leq n$, M contains an $m \times k$ submatrix of columns with non-negative rank $\Omega(R)$, where $R := \min\left(\left(\frac{k}{\log n}\right)^{\frac{1}{3}}, \frac{\text{rk}_+(M)}{\log n}\right)$.*

A remarkable consequence is the following:

- M contains an $s_1 \times s_2$ submatrix with $s_1, s_2 \leq \text{rk}_+(M)^3$ and non-negative rank $\Omega(\frac{\text{rk}_+(M)}{\log n \log m})$. Moreover, If M is a square matrix then so is the submatrix.

In some cases, a stronger conclusion is possible. For example, if $\text{rk}_+(M) = n$ then every $m \times k$ submatrix of M has non-negative rank k . Theorem 1 becomes interesting if $\log n \ll \text{rk}_+(M) \ll n$. For example, if M is $n \times n$ with $\text{rk}_+(M)$ roughly n^ϵ , we obtain an $n^{3\epsilon} \times n^{3\epsilon}$ submatrix of non-negative rank roughly n^ϵ , and also an $n^\epsilon \times n^\epsilon$ submatrix of non-negative rank roughly $n^{\epsilon/3}$. How far from truth is the estimate from Theorem 1 is an interesting question. In Section 6.3, we will see that the result gives a qualitatively correct picture: the exponent $1/3$ can be replaced by $1/2$ at best.

Given a Boolean matrix $M \in \{0, 1\}^{m \times n}$, let us define its *1-partition number*, $\chi_1(M)$, as the smallest s such that M can be written as

$$M = LR, \quad \text{with } L \in \{0, 1\}^{m \times s}, R \in \{0, 1\}^{s \times m},$$

where the operations are over \mathbb{R} . The definition emphasizes the analogy with rk_+ , and χ_1 is also sometimes referred to as *binary rank*. On the other hand, the phrase “partition number” comes from communication complexity. The name is justified: it is easy to see that $\chi_1(M)$ equals the smallest s such that the 1-entries of M can be partitioned into s 1-monochromatic rectangles (i.e., rank-one Boolean matrices). Finally, when M is viewed as the adjacency matrix of a bipartite graph, $\chi_1(M)$ also appears under the name *biclique partition number* [3].

In the case of χ_1 , we obtain a similar but simpler result:

- ▶ **Theorem 2.** *Let M be an $m \times n$ Boolean matrix with $n \geq 2$. Then for every $k \leq n$, M contains an $m \times k$ submatrix of k columns with 1-partition number $\Omega(\min(\sqrt{k}, \frac{\chi_1(M)}{\log n}))$.*

One consequence is the following (cf. Corollary 6):

- if $\chi_1(M) = p$ then M contains a $p \times p$ submatrix with 1-partition number $\Omega(p^{1/4})$.

The results on 1-partition number imply similar statements in communication complexity; they will be presented in Section 5. Whether these witnessing results can be significantly improved is an intriguing question. It is intimately related to the log-rank conjecture; this connection is discussed in Section 5.

Theorems 1 and 2 are proved in Sections 6.2 and 4, respectively. The proof of Theorem 2 is self-contained. Theorem 1 uses geometrical interpretation of non-negative rank in terms of extended formulations of polytopes and also employs known bounds on complexity of quantifier elimination.

Notation

All logarithms are in base 2 and $[n] := \{1, \dots, n\}$.

3 A combinatorial lemma

Both Theorems 1 and 2 rely on a simple combinatorial lemma.

- ▶ **Lemma 3.** *Let $\mathcal{A} \subseteq 2^{[n]}$ be a family of subsets of $[n]$. Assume that $1 \leq k \leq n$ is such that every k -element subset of $[n]$ is contained in some $A \in \mathcal{A}$. Then there exists a subfamily $\mathcal{A}' \subseteq \mathcal{A}$ of size $|\mathcal{A}'| \leq O(|\mathcal{A}|^{\frac{1}{k}} \log(n/k))$ with $\bigcup \mathcal{A}' = [n]$. In particular, if $|\mathcal{A}| \leq 2^k$ then $|\mathcal{A}'| \leq O(\log n)$.*

13:4 Hard Submatrices for Non-Negative Rank and Communication Complexity

Proof. Assume that $|\mathcal{A}| \leq a^k$. Let t be the size of a largest set in \mathcal{A} . Then we have

$$\binom{n}{k} \leq a^k \binom{t}{k}.$$

Hence $t \geq \frac{n}{ea}$, using the estimates $\binom{n}{k}^k \leq \binom{n}{k}$, $\binom{t}{k} \leq \left(\frac{et}{k}\right)^k$. Take some $A_0 \in \mathcal{A}$ of size t . Let

$$\mathcal{A}_1 := \{A \setminus A_0 : A \in \mathcal{A}\}.$$

Then every subset of $U_1 := [n] \setminus A_0$ of size *at most* k is contained in some member of \mathcal{A}_1 . The size of U_1 is at most $n(1 - \frac{1}{ea})$. Similarly, take a largest set A_1 from \mathcal{A}_1 and obtain a new family $\mathcal{A}_2 \subseteq 2^{U_2}$ on $U_2 := U_1 \setminus A_1$. After s steps, the size of U_s is at most $n(1 - \frac{1}{ea})^s$ and after $s \leq O(a \log(n/k))$ steps we have $|U_s| \leq k$. This guarantees that the largest set in \mathcal{A}_s is U_s itself and $[n] = \bigcup_{i=0}^s A_i$. By construction, every A_i is contained in some element of the original family \mathcal{A} . ◀

For some range of parameters, the lemma can be also proved from the Min Max Theorem of Lipton and Young in [13] which would also give an approximate version of it.

An application (which will not be explicitly used) is the following. A subadditive measure on $[n]$ is a function $\mu : 2^{[n]} \rightarrow \mathbb{R}$ such that $\mu(A_1 \cup A_2) \leq \mu(A_1) + \mu(A_2)$ holds for every $A_1, A_2 \subseteq [n]$.

► **Corollary 4.** *Let μ be a subadditive measure on $[n]$. Assume $1 \leq k \leq n$ and that every k -element subset of $[n]$ has measure at most s . Let N be the number of \subseteq -maximal subsets of $[n]$ of measure at most s . Then $\mu([n]) \leq O(sN^{\frac{1}{k}} \log(n/k))$.*

4 1-Partition number

In this section, we prove Theorem 2.

Let M be an $m \times n$ matrix with rows indexed by $[n] = \{1, \dots, n\}$. Given $A \subseteq [n]$, M_A denotes the submatrix obtained by removing the rows outside of A from M . Observe that¹

$$\chi_1(M_{A_1 \cup A_2}) \leq \chi_1(M_{A_1}) + \chi_1(M_{A_2}), \quad (1)$$

and so $\chi_1(M_A)$ can be viewed as a subadditive measure on $[n]$ whenever M is fixed.

If a matrix M has rank r , its rows are a linear combination of a subset of r rows of M . This means that every column of M is determined by a fixed subset of r coordinates. If M is Boolean, this leads to the following useful fact:

■ if M has distinct columns then $n \leq 2^{\text{rk}(M)}$ (similarly for rows).

► **Lemma 5.** *Let M be an $m \times n$ Boolean matrix of rank r . Given $s \in [n]$, let \mathcal{A} be the collection of maximal subsets $A \subseteq [n]$ with $\chi_1(M_A) \leq s$ (i.e., $\chi_1(M_A) \leq s$ and $\chi_1(M_{A'}) > s$ for every $A' \supsetneq A$). Then $|\mathcal{A}| \leq 2^{(r+s)^2}$.*

Proof. Let $v_1, \dots, v_n \in \mathbb{R}^m$ be the columns of M . Given $L \in \{0, 1\}^{m \times s}$, let

$$L^* := \{i \in [n] : \exists y \in \{0, 1\}^s v_i = Ly\}.$$

Let $\mathcal{L} := \{L^* : L \in \{0, 1\}^{m \times s}\}$.

¹ If A_1, A_2 are disjoint, this is quite obvious. Otherwise consider $A_1, A_2 \setminus A_1$.

We claim that $\mathcal{A} \subseteq \mathcal{L}$. If $\chi_1(M_A) \leq s$, we can write $M_A = LR$ with $L \in \{0, 1\}^{m \times s}$ and $R \in \{0, 1\}^{s \times |A|}$. This means that every v_i , $i \in A$, is a Boolean linear combination of the columns of L and $A \subseteq L^*$. Furthermore, if A is maximal, we must have $A = L^*$.

We now want to estimate the size of \mathcal{L} . The set L^* consists of indices $i \in [n]$ so that there exists $x \in \mathbb{R}^n$, $y \in \mathbb{R}^s$ satisfying

$$Mx - Ly = 0 \tag{2}$$

such that $y \in \{0, 1\}^s$ and x is the i -th unit vector. Since M has rank r and L has rank at most s , the system (2) is equivalent to a subsystem of $t := \min((s + r), m)$ equations. These correspond to rows of the matrix (M, L) . Hence, in order to determine L^* , it is sufficient to specify a t -element subset of $[m]$ together with the $t \times s$ submatrix of L . This gives the estimate

$$|\mathcal{L}| \leq \binom{m}{t} 2^{ts} \leq 2^{t(s + \log m)}.$$

Finally, we can assume that M has distinct rows and so $\log m \leq r$, obtaining the bound $2^{(r+s)^2}$. ◀

► **Theorem 2 (restated).** *Let M be an $m \times n$ Boolean matrix with $n \geq 2$. Then for every $k \leq n$, M contains an $m \times k$ submatrix of k columns with 1-partition number $\Omega(\min(\sqrt{k}, \frac{\chi_1(M)}{\log n}))$.*

Proof. Let r be the rank of M . We will assume $r \leq \frac{k^{1/2}}{2}$. Otherwise, observe that M contains a full rank $r \times r$ submatrix, χ_1 is lower-bounded by rank, and the conclusion of the theorem follows.

Let s be the maximum $\chi_1(M_A)$ over all $A \subseteq [n]$ of size k . Let \mathcal{A} be the family from the previous lemma. If $|\mathcal{A}| \geq 2^k$, we have $2^k \leq 2^{(s+r)^2}$ and therefore $s \geq \frac{k^{1/2}}{2}$ from the assumption on r .

Assume $|\mathcal{A}| \leq 2^k$. By Lemma 3, there exists a subfamily $\mathcal{A}' \subseteq \mathcal{A}$ of size $O(\log n)$ which covers $[n]$. Using (1), this implies $\chi_1(M) \leq O(s \log n)$ and so $s \geq \Omega(\chi_1(M)/\log n)$. ◀

► **Corollary 6.** *Let M be as above with $\chi_1(M) = p$. Then M contains*

- (i) *a submatrix of at most p^2 columns with partition number $\Omega(p/\log n)$,*
- (ii) *a submatrix with at most p^2 rows and columns with partition number $\Omega(p/(\log n \log m))$.*

If M is a square matrix then so is the submatrix.

- (iii) *a submatrix with p columns with partition number $\Omega(p^{\frac{1}{2}})$*
- (iv) *a $p \times p$ submatrix with partition number $\Omega(p^{\frac{1}{4}})$.*

Proof. Part (i). If $n \leq p^2$, M itself satisfies the statement. Otherwise apply the theorem with $k = p^2$.

Part (ii). Apply (i) again to the transpose of the submatrix obtained in (i). If $m = n$, we can enlarge the submatrix to a square matrix.

Part (iii). Without loss of generality, we can assume that the columns of M are distinct. This implies that M has rank at least $\log n$. If $\sqrt{p} \leq p/\log n$, apply the theorem to obtain the desired matrix. Otherwise, we have $p \geq \log^2 n$. M contains a submatrix of p columns of rank at least $\min(p, \log n) \geq \sqrt{p}$.

Part (iv) follows by taking the submatrix from (iii), and applying (iii) to its transpose. ◀

► **Remark 7.** The bound of Theorem 2 can be slightly improved to give $\Omega(\sqrt{k \log(1 + \frac{\chi_1(M)}{k^{1/2} \log n})})$, as long as $k^{1/2} \leq \chi_1(M)/\log n$. For example, if $k = \chi_1(M)/\log n$, we obtain a submatrix of k columns with 1-partition number $\Omega(\sqrt{k \log k})$.

Furthermore, M always contains a submatrix M' of k columns with $\chi_1(M') \geq \chi_1(M) \cdot \left\lceil \frac{n}{k} \right\rceil^{-1}$, which gives better parameters if $\chi_1(M)$ is close to n .

4.1 A somewhat non-trivial example

We now give a finite example which shows that the most optimistic form of witnessing fails for χ_1 .

► **Theorem 8.** *There exists a 5×6 Boolean matrix M with $\chi_1(M) = 5$ such that every 5×5 submatrix of M has 1-partition number at most 4.*

Proof. Let

$$M := \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

We first argue that $\chi_1(M) > 4$, which implies $\chi_1(M) = 5$ since M has 5 rows.

Suppose that $\chi_1(M) \leq 4$. Then there exists a set of Boolean row-vectors $V = \{v_1, \dots, v_4\}$ such that every row of M is their Boolean linear combination; i.e., of the form $\sum_{i \in A} v_i$ for some $A \subseteq \{1, \dots, 4\}$. Note that in this expression, the non-zero coordinates of v_i , $i \in A$, are a subset of the non-zero coordinates of the given row. Using this observation, it is easy to see that V must consist of the first 4 rows of M . If $\chi_1(M) \leq 4$ this means that the last row of M is a Boolean combination of the first four rows, which is clearly impossible.

We now show that every submatrix obtained by removing a column from M has χ_1 at most 4.

First, assume that M' has been obtained by removing the third column. The resulting matrix, together with a partition into four 1-monochromatic rectangles a, b, c, d , is as follows:

$$M' = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} a & 0 & 0 & a & b \\ 0 & c & c & 0 & b \\ 0 & 0 & d & d & 0 \\ 0 & 0 & d & d & b \\ a & c & c & a & b \end{pmatrix}.$$

Second, assume that M'' has been obtained by removing the last column. The resulting matrix, together with its partition, is the following:

$$M'' = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} a & 0 & 0 & 0 & a \\ 0 & b & 0 & b & 0 \\ 0 & 0 & c & d & d \\ 0 & 0 & 0 & d & d \\ a & b & c & b & a \end{pmatrix}.$$

Finally, note that if we remove from M the first or the second column, we obtain M' (up to a permutation of rows and columns). And, if we remove the fourth or fifth column, we obtain M'' . Hence indeed, every 5×5 submatrix has χ_1 at most 4 ◀

By placing n copies of the matrix from Theorem 8 on diagonal, we obtain:

► **Corollary 9.** *For every n , there exists a $5n \times 6n$ Boolean matrix M with $\chi_1(M) = 5n$ such that every submatrix obtained by removing a column of M has 1-partition number strictly less than $5n$.*

5 Communication complexity, and a comparison with the log-rank conjecture

Given an $m \times n$ Boolean matrix M , consider the following two-player game: Alice knows $i \in [m]$, Bob knows $j \in [n]$, and they are supposed to compute the value of $M_{i,j}$. Denote by $\text{cc}(M)$ the deterministic communication complexity of this game. For details about the communication model, see for example [11, 16].

In order to relate communication complexity with χ_1 , we need the following classical fact (the first inequality is due to Yao, the second is due to Yannakakis [20]): if M is non-constant then

$$\log(\chi_1(M) + 1) \leq \text{cc}(M) \leq O(\log^2 \chi_1(M)). \quad (3)$$

► **Proposition 10.** *Let M be a Boolean matrix with communication complexity c . Then there exist $k \geq \Omega(\sqrt{c})$ and a $2^k \times 2^k$ submatrix of M with communication complexity at least $k/4 - O(1)$.*

Proof. From (3), there exists $k \geq \Omega(\sqrt{c})$ with $\chi_1(M) \geq 2^k$. Corollary 6, part (iv), gives $2^k \times 2^k$ submatrix M' with $\chi_1(M') \geq \Omega(2^{k/4})$. By (3), we have $\text{cc}(M') \geq k/4 - O(1)$. ◀

It is worthwhile to compare this with what is predicted by the log-rank conjecture [14] of Lovász and Saks.

► **Log-rank conjecture.** *There is a constant α such that $\text{cc}(M) \leq O(\log^\alpha(\text{rk}(M)))$ for any non-zero Boolean matrix M .*

► **Proposition 11.** *Assume the log-rank conjecture. Then every Boolean matrix with communication complexity c contains a $2^k \times 2^k$ submatrix M' with $\chi_1(M') = 2^k$, communication complexity $k + 1$, and $k \geq \Omega(c^{1/\alpha})$.*

Proof. If M has communication complexity c then, by the log-rank conjecture, M has rank at least 2^k with $k \geq \Omega(c^{1/\alpha})$. Hence M contains a full-rank $2^k \times 2^k$ submatrix M' . Since $\chi_1(M') \geq \text{rk}(M')$, we have $\chi_1(M') = 2^k$. If c is sufficiently large, so that $k \geq 1$, then M' is non-constant and we obtain $\text{cc}(M') \geq k + 1$ by (3). ◀

This is almost what has been proved in Proposition 10. One difference is that the constant α in Proposition 11 is unconditionally set to 2 in Proposition 10. However, there is a more important qualitative difference. The submatrix presented in Proposition 11 has *highest possible* communication complexity: the protocol in which Alice sends her input to Bob and Bob sends back the answer (or vice versa), is optimal. Any other protocol cannot save even *one bit* of communication. In contrast, Proposition 10 presents a submatrix with only a very high communication complexity. To summarize, Proposition 10 confirms a prediction of the log-rank conjecture. But with worse parameters than what the conjecture predicts: consequently the bound in the proposition is far from tight, or the conjecture is false.

Another consequence is:

► **Remark 12.** In order to solve the log-rank conjecture, it is sufficient to focus on $2^k \times 2^k$ matrices with communication complexity at least $k/4 - O(1)$.

6 Non-negative rank

6.1 Extended formulations and separation complexity

Let us first make a short detour into extended formulations of convex polyhedra.

A *polyhedron* $P \subseteq \mathbb{R}^r$ is a (possibly unbounded) set defined by a finite number of linear constraints. Following [20, 18, 2], define the *extension complexity* of P , $\text{xc}(P)$, as the smallest s such that P is a linear projection of a polyhedron $Q \subseteq \mathbb{R}^m$ where Q can be defined using s inequalities (and any number of equalities). Observe that P with extension complexity s can be expressed in the standard form

$$x \in P \text{ iff } \exists_{y \in \mathbb{R}^s} Cx + Dy = b, y \geq 0,$$

where $C \in \mathbb{R}^{t \times r}$, $D \in \mathbb{R}^{t \times s}$ and $b \in \mathbb{R}^t$ for some t .

Let V be a finite subset of \mathbb{R}^r . Given $A \subseteq V$, its *separation complexity*, $\text{sep}_V(A)$, is the minimum $\text{xc}(P)$ over all polyhedra $P \subseteq \mathbb{R}^r$ with²

$$P \cap V = A;$$

such a P is called a *separating* polyhedron for A . In other words, $\text{sep}_V(A)$ is the smallest s so that we can distinguish points in A from points in $V \setminus A$ by means of a linear program with s inequalities. Moreover, such a program can be rewritten as

$$x \in A \text{ iff } (x \in V \text{ and } \exists_{y \in \mathbb{R}^s} Cx + Dy = b, y \geq 0).$$

The notion of separation complexity has been studied in [7, 8, 9] in the case when $V = \{0, 1\}^n$ is the Boolean cube. The following theorem is of independent interest and can be seen as an extension of similar results in [7, 9]. The proof is a considerable simplification of the previous ones.

► **Theorem 13.** *Let V be a non-empty finite subset of \mathbb{R}^r . Given a parameter $s \geq 1$, let \mathcal{A} be the collection of subsets A of V with $\text{sep}_V(A) \leq s$. Then*

$$|\mathcal{A}| \leq 2^{O((s+r+s)^2 \log |V|)}.$$

The proof is delegated to the appendix.

An immediate consequence of Theorem 13 is a theorem from [9]:

■ if $V = \{0, 1\}^n$ then there exists $A \subseteq V$ with $\text{sep}_V(A) \geq 2^{n^{\frac{1}{3}(1-o(1))}}$.

6.2 Submatrices of large non-negative rank

In order to apply Theorem 13, we also need a connection between extension complexity and non-negative rank. This is provided by the notion of slack matrix introduced in [20]. Following [20, 2], we now define what it is. Let V be a sequence v_1, \dots, v_{m_1} of points in \mathbb{R}^r and $L(x)$ a system $\ell_1(x) \geq b_1, \dots, \ell_{m_2}(x) \geq b_{m_2}$ of inequalities in \mathbb{R}^r . The slack matrix with respect to V and $L(x)$ is the $m_2 \times m_1$ matrix S such that

$$S_{i,j} = \ell_i(v_j) - b_i.$$

Let $P_0 := \text{conv}(V)$ be the convex hull of V and $P_1 := \{x \in \mathbb{R}^r : L(x) \text{ holds}\}$. If $P_0 \subseteq P_1$ then S is non-negative. In [2], we can find:

² If no such polyhedron exists, which may happen if V is not convexly independent, we set $\text{sep}_V(A) := \infty$.

► **Lemma 14** ([2]). *Let $P_0 \subseteq P_1$ and S be as above. Define $xc(P_0, P_1)$ as the minimum $xc(P)$ over all polyhedra with $P_0 \subseteq P \subseteq P_1$. Then*

$$rk_+ S - 1 \leq xc(P_0, P_1) \leq rk_+ S.$$

► **Theorem 1** (restated). *Let M be an $m \times n$ non-negative real matrix with $n \geq 2$. Then for every $k \leq n$, M contains an $m \times k$ submatrix of columns with non-negative rank $\Omega(R)$, where $R := \min\left(\left(\frac{k}{\log n}\right)^{\frac{1}{3}}, \frac{rk_+(M)}{\log n}\right)$.*

Proof. Let r be the rank of M . We can write $M = LR$ where $L \in \mathbb{R}^{m \times r}$, $R \in \mathbb{R}^{r \times n}$. Let $V \subseteq \mathbb{R}^r$ be the set of columns v_1, \dots, v_n of R . (Without loss of generality, the columns of M are distinct). Given $A \subseteq [n]$, let M_A be the submatrix obtained by deleting columns outside of A from M . Also let $V_A := \{v_i : i \in A\}$. Then M_A can be interpreted as the slack matrix of the polytope $P_A = \text{conv}(V_A)$ and the polyhedron $Q = \{x \in \mathbb{R}^d : Lx \geq 0\}$.

Suppose that for every A of size k , $rk_+(M_A) \leq s$. Then for every such A , there is a polyhedron Q_A with $V_A \subseteq Q_A \subseteq Q$ with $xc(Q_A) \leq s$. Let $A^* := V \cap Q_A$. Then Q_A is a separating polyhedron for $A^* \supseteq A$. Let \mathcal{A} be the collection of A^* over all A of size k . Theorem 13 implies

$$|\mathcal{A}| \leq 2^{c \log n (s+r)^3},$$

where c is an absolute constant.

We will assume $r \leq \left(\frac{k}{2^c \log n}\right)^{1/3}$. Otherwise M contains a full rank $r \times r$ submatrix, rk_+ is lower-bounded by rank, and the conclusion of the theorem follows.

If $|\mathcal{A}| \geq 2^k$, we obtain $c \log n (s+r)^3 \geq k$ and hence $s \geq \Omega((k/\log n)^{1/3})$ from the assumption on r .

Assume $|\mathcal{A}| \leq 2^k$. By Lemma 3, there exists a subfamily $\mathcal{A}' \subseteq \mathcal{A}$ of size $O(\log n)$ which covers $[n]$. This implies (note that (1) holds also for non-negative rank) $rk_+(M) \leq O(s \log n)$ and $s \geq \Omega(rk_+(M)/\log n)$. ◀

The following is proved similarly to Corollary 6:

► **Corollary 15.** *Let M be a non-negative $m \times n$ matrix with $rk_+(M) = p$. Then M contains*

- (i) *an $s_1 \times s_2$ submatrix with $s_1, s_2 \leq p^3$ with non-negative rank $\Omega\left(\frac{p}{\log n \log m}\right)$. If $m = n$, we can assume $s_1 = s_2$.*
- (ii) *a $p \times p$ submatrix with non-negative rank $\Omega\left(\frac{p^{\frac{1}{3}}}{\log^{\frac{1}{3}} n \log m}\right)$.*

6.3 Tightness

In [15], Moitra has constructed a non-negative matrix M with the following properties:

- M is $3rn \times 3rn$, $rk_+(M) \geq 4r$, any submatrix with $< n$ columns has non-negative rank at most $3r$.

Observe that in order to witness the non-negative rank of this M exactly, one needs a constant fraction of the columns of M . On the other hand, the gap between the non-negative rank of M and that of its submatrices is quite mild.

We now give a different example which is of a similar flavor as the bound from Theorem 1. It also shows that the constant $\frac{1}{3}$ in the theorem can be replaced by $\frac{1}{2}$ at best. The example follows from very non-trivial results of Kwan et al. [12]. A similar bound would follow from the more general result of Shitov [19].

► **Theorem 16.** For every n , there exists an $n \times n$ matrix with non-negative rank $\Omega(\sqrt{n})$ such that every $n \times k$ submatrix has non-negative rank $O(\sqrt{k})$.

Proof. From [12], there exists an n -vertex polygon $P \subseteq \mathbb{R}^2$ with vertices lying on the unit circle with extension complexity $\Omega(\sqrt{n})$. Let M be its slack matrix with columns corresponding to vertices v_1, \dots, v_n of P . From Lemma 14, we have $\text{rk}_+(M) \geq \Omega(\sqrt{n})$. Given an $n \times k$ submatrix M' with columns i_1, \dots, i_k , Lemma 14 shows that $\text{rk}_+(M')$ is at most the extension complexity of $\text{conv}(v_{i_1}, \dots, v_{i_k})$ (plus 1). Using another result from [12], every k -gon with vertices on the unit circle has extension complexity at most $O(\sqrt{k})$. ◀

7 Open problems

Our first two open problems are concerned with tightness of the bounds in Theorems 1 and 2.

► **Open problem 1.** Let M be $m \times n$ non-negative matrix. Does M contain a submatrix of at most $\text{rk}_+(M)^2$ columns with non-negative rank $\Omega(\text{rk}_+(M))$?

► **Open problem 2.** Find a Boolean matrix M with $\chi_1(M) = p$ such that every $p \times p$ submatrix has 1-partition number much smaller than p .

As far as we can see, the bound from Problem 1 is consistent with what we know about non-negative rank, and would be optimal. The task is to improve Corollary 15(i) in two different ways: first, to reduce the dependence on $\text{rk}_+(M)$ from cubic to quadratic and, second, to eliminate the logarithmic dependence on the size of M altogether. For Problem 2, Theorem 8 gives an M with submatrices of χ_1 strictly less than p ; there should exist a construction with a larger gap.

As discussed in Section 5, in order to solve the log-rank conjecture, it is enough to focus on matrices with large 1-partition number. The following is the extreme case of this question:

► **Open problem 3.** Suppose M is $n \times n$ Boolean matrix with $\chi_1(M) = n$. How small can the rank of M be?

References

- 1 S. Basu, R. Pollack, and M.F. Roy. *Algorithms in real algebraic geometry*. Springer-Verlag, 2006.
- 2 G. Braun, S. Fiorini, S. Pokutta, and D. Steuer. Approximation limits of linear programs (beyond hierarchies). *Math. of Operations Research*, 40(3), 2015.
- 3 S. Chandran, D. Issac, and A. Karrenbauer. On the parameterized complexity of biclique cover and partition. In *International Symposium on Parameterized and Exact Computation*, 2016.
- 4 P. Erdős. Problems and results in combinatorial analysis and graph theory. *Annals of Discrete Mathematics*, 38:81–92, 1988.
- 5 S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Linear vs. semidefinite extended formulations: exponential separation and strong lower bounds. In *Symposium on the Theory of Computing*, 2011.
- 6 M. Göös, I. Newman, A. Riazanov, and D. Sokolov. Hardness condensation by restriction. *ECCC*, 2023.
- 7 P. Hrubeš. On ϵ -sensitive monotone computations. *Computational Complexity*, 29(2), 2020.
- 8 P. Hrubeš. On the complexity of computing a random Boolean function over the reals. *Theory of Computing*, 16(9):1–12, 2020.
- 9 P. Hrubeš and N. Talebanfard. On the extension complexity of polytopes separating subsets of the Boolean cube. *Disc. and Comp. Geom.*, 70(1), 2022.

- 10 R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 538–545, 1995.
- 11 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1996.
- 12 Matthew Kwan, Lisa Sauermann, and Yufei Zhao. Extension complexity of low-dimensional polytopes. *Transactions of the American Mathematical Society*, 375(6), 2022.
- 13 Richard Lipton and Neal Young. Simple strategies for large zero-sum games with applications to complexity theory. *CoRR*, cs.CC/0205035, May 2002. doi:10.1145/195058.195447.
- 14 L. Lovász and M. Saks. Lattices, mobius functions and communications complexity. In *29th Annual Symposium on Foundations of Computer Science (FOCS 1988)*, pages 81–90, 1988.
- 15 Ankur Moitra. An almost optimal algorithm for computing nonnegative rank. In *SIAM J. Comput.*, 2013. URL: <https://api.semanticscholar.org/CorpusID:1920044>.
- 16 Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020.
- 17 V. Rödl and R. A. Duke. On graphs with small subgraphs of large chromatic number. *Graphs and Combinatorics*, 1:91–96, 1985.
- 18 Thomas Rothvoß. Some 0/1 polytopes need exponential size extended formulations. *CoRR*, abs/1105.0036, 2011. arXiv:1105.0036.
- 19 Y. Shitov. Sublinear extension of polygons. *arXiv*, 2014. arXiv:1412.0728.
- 20 Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991.

A Proof of Theorem 13

The proof uses known results on quantifier elimination which we first outline. We follow the monograph of Basu, Pollack and Roy [1]. Theorem 13 requires an elimination of only a single block of existential quantifiers, so we focus on this case only.

For $b \in \mathbb{R}$, let

$$\operatorname{sgn}(b) := \begin{cases} 1, & b > 0, \\ 0, & b = 0, \\ -1, & b < 0. \end{cases}$$

Given $b = \langle b_1, \dots, b_m \rangle \in \mathbb{R}^m$, let $\operatorname{sgn}(b) := \langle \operatorname{sgn}(b_1), \dots, \operatorname{sgn}(b_m) \rangle \in \{-1, 0, 1\}^m$. Let $F = F(z, y)$ be a sequence of m polynomials $f_1, \dots, f_m \in \mathbb{R}[z, y]$ in variables $z = \{z_1, \dots, z_{k_1}\}$ and $y = \{y_1, \dots, y_{k_2}\}$. Given $a \in \mathbb{R}^{k_1}$, define $\operatorname{SGN}_1(F, a) \subseteq \{-1, 0, 1\}^m$

$$\operatorname{SGN}_1(F, a) := \{\operatorname{sgn}(F(a, b)) : b \in \mathbb{R}^{k_2}\}.$$

Let

$$\operatorname{SGN}(F) := \{\operatorname{SGN}_1(F, a) : a \in \mathbb{R}^{k_1}\}.$$

Theorem 14.16 from [1] provides the following bound on the size of SGN :

► **Theorem ([1]).** *If every polynomial in F has degree at most d then*

$$|\operatorname{SGN}(F)| \leq m^{(k_1+1)(k_2+1)} d^{O(k_1)O(k_2)}. \quad (4)$$

We now apply this result to the case of Theorem 13. Let V, s, \mathcal{A} be as in the assumption. Every $A \in \mathcal{A}$ can be described by a linear system with s inequalities. Namely, for every $x \in V$,

$$x \in A \text{ iff } \exists_{y \in \mathbb{R}^s} Cx + Dy = b, y \geq 0, \quad (5)$$

where $C \in \mathbb{R}^{t \times r}$, $D \in \mathbb{R}^{t \times s}$ and $b \in \mathbb{R}^t$. Since $Cx + Dy = b$ is a system of equations in $r + s$ variables x, y , we can also assume $t = r + s$.

13:12 Hard Submatrices for Non-Negative Rank and Communication Complexity

Let us view the parameters C, D, b in (5) as variables. Let z be the set of these variables, of size $k_1 = (r+s)(r+s+1)$. Given $v \in V$, let $F_v(z, y)$ be the sequence of $(r+s)$ polynomials

$$Cv + Dy - b$$

in variables z and $y = \{y_1, \dots, y_s\}$. Let $F(z, y)$ be the union of $F_v(z, y)$ over all $v \in V$, together with the polynomials y_1, \dots, y_s . Hence F consists of $m = s + |V|(r+s)$ polynomials of degree at most two.

$F(z, y)$ is set up so that

$$|\mathcal{A}| \leq |\text{SGN}(F)|.$$

To see this, observe that whenever the parameters z are fixed, the set $A \subseteq V$ given by (5) is uniquely determined by $\text{SGN}_1(F(z, y))$. Since every $A \in \mathcal{A}$ is obtained by some fixing of the parameters, we indeed obtain $|\mathcal{A}| \leq |\text{SGN}(F(z, y))|$.

Finally, we can apply (4) to estimate $|\text{SGN}(F)|$ with $m = s + |V|(r+s)$, $k_1 = (r+s)(r+s+1)$, $k_2 = s$, and $d = 2$. To simplify the expression, we can assume $s+r \leq |V|$; otherwise the upper bound asserted in Theorem 13 exceeds the trivial bound $|\mathcal{A}| \leq 2^{|V|}$. This means that $m \leq 2|V|^2$. If we loosen the bound (4) as $|\text{SGN}(F)| \leq (dm)^{O(k_1)O(k_2)}$, we obtain (recall that $s \geq 1$)

$$|\text{SGN}(F)| \leq 2^{O(s(s+r)^2 \log |V|)},$$

as required.

Complexity of Robust Orbit Problems for Torus Actions and the *abc*-Conjecture

Peter Bürgisser  

Institute of Mathematics, Technische Universität Berlin, Germany

Mahmut Levent Doğan  

Institute of Mathematics, Technische Universität Berlin, Germany

Visu Makam  

Radix Trading, Amsterdam, The Netherlands

Michael Walter  

Faculty of Computer Science, Ruhr-Universität Bochum, Germany

Avi Wigderson 

School of Mathematics, Institute for Advanced Study, Princeton, NJ, USA

Abstract

When a group acts on a set, it naturally partitions it into orbits, giving rise to *orbit problems*. These are natural algorithmic problems, as symmetries are central in numerous questions and structures in physics, mathematics, computer science, optimization, and more. Accordingly, it is of high interest to understand their computational complexity. Recently, [16] gave the first polynomial-time algorithms for orbit problems of *torus actions*, that is, actions of commutative continuous groups on Euclidean space. In this work, motivated by theoretical and practical applications, we study the computational complexity of *robust* generalizations of these orbit problems, which amount to *approximating* the distance of orbits in \mathbb{C}^n up to a factor $\gamma \geq 1$. In particular, this allows deciding whether two inputs are approximately in the same orbit or far from being so. On the one hand, we prove the NP-hardness of this problem for $\gamma = n^{\Omega(1/\log \log n)}$ by reducing the closest vector problem for lattices to it. On the other hand, we describe algorithms for solving this problem for an approximation factor $\gamma = \exp(\text{poly}(n))$. Our algorithms combine tools from invariant theory and algorithmic lattice theory, and they also provide group elements witnessing the proximity of the given orbits (in contrast to the algebraic algorithms of prior work). We prove that they run in polynomial time if and only if a version of the famous number-theoretic *abc*-conjecture holds – establishing a new and surprising connection between computational complexity and number theory.

2012 ACM Subject Classification Computing methodologies → Algebraic algorithms; Computing methodologies → Combinatorial algorithms; Theory of computation → Algebraic complexity theory

Keywords and phrases computational invariant theory, geometric complexity theory, orbit problems, *abc*-conjecture, closest vector problem

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.14

Related Version *ArXiv Version*: <https://arxiv.org/abs/2405.15368>

Funding *Peter Bürgisser*: Supported by the ERC under the European Union’s Horizon 2020 research and innovation programme (grant agreement no. 787840).

Mahmut Levent Doğan: Supported by the ERC under the European Union’s Horizon 2020 research and innovation programme (grant agreement no. 787840).

Visu Makam: Partially supported by NSF Grant CCF-1900460 and the University of Melbourne.

Michael Walter: Supported by the European Union (ERC, SYMOPTIC, 101040907), by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2092 CASA – 390781972, by the BMBF (QuBRA, 13N16135), and by the Dutch Research Council (NWO grant OCENW.KLEIN.267).

Avi Wigderson: Supported by the NSF Grant CCF-1900460.



© Peter Bürgisser, Mahmut Levent Doğan, Visu Makam, Michael Walter, and Avi Wigderson;

licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 14; pp. 14:1–14:48



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Acknowledgements The authors thank Matías Bender, Alperen A. Ergür, Jonathan Leake, and Philipp Reichenbach for productive discussions.

1 Introduction

Computational invariant theory was a central topic of 19th century algebra, see the historical account in [57]. In the second half of the 20th century, structural progress was made through Mumford’s invention of geometric invariant theory [73] and computational progress came through the theory of Gröbner bases, see [84]. More recently, it was realized that algorithmic questions in invariant and representation theory are deeply connected with the core complexity questions of P vs. NP and VP vs. VNP.

On the one hand, Mulmuley and Sohoni’s Geometric Complexity Theory (GCT) [71] highlights the inherent symmetries of complete problems of these complexity classes. This was the starting point of several specific invariant theoretic and representation theoretic attacks on the VP vs. VNP questions, which has led to many new questions, techniques, and much faster algorithms: for example, see [70, 31, 15, 63].

The other connection is through the work of Impagliazzo and Kabanets [52], which uses Valiant’s completeness theory for VP and VNP to construct efficient deterministic algorithms for the basic PIT (Polynomial Identity Testing) problem. This problem, again thanks to Valiant’s completeness result, has natural symmetries which resemble basic problems of invariant theory. Major progress was recently achieved in this direction in [40, 33, 50, 51, 25]. This work was followed by [26, 43, 49, 34, 19, 18, 20, 17]; we refer to [17] for a description of the state-of-art.

In addition to these fundamental connections to computational complexity, orbit problems appear in many other areas, for example in physics, since the symmetries are often given by the actions of Lie groups, see [6, 65, 39]. Recently, a connection of algebraic statistics to orbit problems was discovered [4, 5]: the problem of finding maximum likelihood estimates in certain statistical models and the “flip flop” algorithm from statistics is precisely related to the invariant theory and the scaling algorithms of [33, 19]; see also [32, 27, 28]. In particular, the setting of log-linear models in [5] relates precisely to the setting of this work.

1.1 Background and high-level summary of results

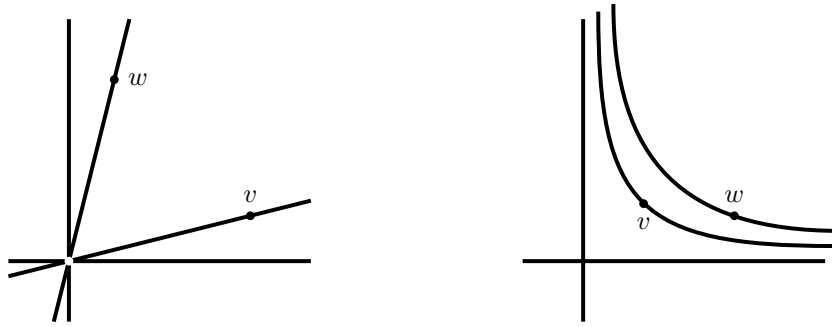
In this paper, we study actions of the commutative group $T := (\mathbb{C}^\times)^d$ and its subgroup $K := (S^1)^d$ on vector spaces $V := \mathbb{C}^n$.¹ The group T is called an *algebraic torus* of rank d , and the subgroup K is known as a *compact torus*. The action of either group partitions the vector space V into *orbits*. That is, for a vector $v \in V$, we consider the set of all vectors reachable from v by applying group elements:

$$\mathcal{O}_v := \{t \cdot v \mid t \in T\}, \quad \mathcal{C}_v := \{k \cdot v \mid k \in K\}.$$

The sets \mathcal{O}_v and \mathcal{C}_v are called the *T-orbit* and *K-orbit* of v , respectively. As K is compact, so is \mathcal{C}_v . However, the *T-orbit* \mathcal{O}_v is in general not closed and hence it is natural to also consider its *orbit closure* $\overline{\mathcal{O}_v}$.² See Figure 1 for two illustrative examples. While torus actions are very well-understood from a structural perspective, they give rise to interesting and challenging computational problems:

¹ Here, \mathbb{C}^\times denotes the multiplicative group of the nonzero complex numbers and $S^1 := \{z \in \mathbb{C} \mid |z| = 1\}$.

² We take the closure with respect to either the Euclidean or the Zariski topology, as they coincide here [72, I §10].



■ **Figure 1** (a) Action of $T = \mathbb{C}^\times$ on $V = \mathbb{C}^2$ given by $t \cdot (x, y) = (tx, ty)$: All orbits are lines through the origin, with the origin excluded. Thus, all orbit closures intersect in the origin. (b) Action of $T = \mathbb{C}^\times$ on $V = \mathbb{C}^2$ given by $t \cdot (x, y) = (tx, t^{-1}y)$: All orbits of vectors with non-zero coordinates $x, y \neq 0$ are closed (they are hyperbolas).

► **Problem 1.1.** *Given a torus action on a vector space V and $v, w \in V$:*

1. **Orbit equality:** *Decide if $\mathcal{O}_v = \mathcal{O}_w$.*
2. **Orbit closure intersection:** *Decide if $\overline{\mathcal{O}_v} \cap \overline{\mathcal{O}_w} \neq \emptyset$.*
3. **Orbit closure containment:** *Decide if $\mathcal{O}_w \subseteq \overline{\mathcal{O}_v}$.*
4. **Compact orbit equality:** *Decide if $\mathcal{C}_v = \mathcal{C}_w$.*

These problems capture and relate to a broad class of “isomorphism”, “classification”, or “transformation” problems. Their computational complexity was determined only very recently: [16] found that all four problems can be decided in polynomial time. This is perhaps surprising, because for actions of noncommutative groups these problems are believed to differ in their computational complexity and, e.g., Problem 1.1 (3) is known to be hard for some actions [10].

In many applications, e.g., the ones in statistics or physics mentioned above, the vectors v, w are not given exactly, but only up to a certain error. If this is the case, the orbit equality problem has to be replaced by a robust generalization, which meaningfully tolerates small perturbations in the input.

The goal of this paper is to define such robust generalizations of the orbit equality problem and to investigate their computational complexity. More precisely, we aim to *approximate the distance between two orbits* given by vectors $v, w \in \mathbb{C}^n$ up to an approximation factor $\gamma > 1$. In particular, this allows deciding whether two points are approximately in the same orbit or far from being so.

At first glance, this appears to simplify the situation, since we no longer need to distinguish between an orbit and its closure. However, surprisingly, the picture arising is far more intricate than for the problems that were dealt with in [16]. On the one hand, we prove NP-hardness when $\gamma = n^{\Omega(1/\log \log n)}$. On the other hand, using a combination of tools from invariant theory and algorithmic lattice theory, we give algorithms that achieve an approximation factor $\gamma = \exp(\text{poly}(n))$ and run in polynomial time if and only if a version of the well-known number-theoretic *abc*-conjecture holds! Our algorithms also return a witness $t \in T$ such that the distance between $t \cdot v$ and w is at most γ times the distance between the orbits. We give precise technical statements below.

We note that our results are not the first examples of an open problem in number theory having an impact on the complexity of algorithms. As is widely known, the *generalized (or extended) Riemann hypothesis* has been used, e.g., for testing primality in polynomial

time [68] (an unconditional deterministic polynomial time algorithm was later given in [1]), the containment of knottedness in NP [58], or the containment of Hilbert’s Nullstellensatz in the polynomial hierarchy [56]. To our best knowledge, however, our work, together with [30, 78], constitutes one of the first examples that the *abc*-conjecture has some bearing on the performance of numerical algorithms.

In the remainder of this introduction we first recap the structure of torus actions and define the input model and complexity parameters used in all our computational problems, algorithms, and results (Section 1.2). We then state the computational problems (Section 1.3) and discuss our hardness and algorithmic results (Sections 1.4 and 1.5). Next, we explain how the “separation hypotheses” that ensure that our algorithms run in polynomial time are intimately related to well-known variants of the *abc*-conjecture in number theory (Section 1.6). Finally, we sketch the proof idea of our lattice lifting result (Section 1.7), and we conclude with open problems for future research (Section 1.9).

1.2 Torus actions: structure, input model, parameters

Every rational action of $T = (\mathbb{C}^\times)^d$ or $K = (S^1)^d$ on a finite-dimensional complex vector space $V = \mathbb{C}^n$ can be parameterized by a *weight matrix* $M \in \mathbb{Z}^{d \times n}$, as follows: For $t = (t_1, t_2, \dots, t_d) \in T$ and $v = (v_1, v_2, \dots, v_n) \in V$,

$$t \cdot v := \left(v_1 \prod_{i=1}^d t_i^{M_{i1}}, v_2 \prod_{i=1}^d t_i^{M_{i2}}, \dots, v_n \prod_{i=1}^d t_i^{M_{in}} \right). \quad (1.1)$$

See Figure 1 for two examples with weight matrix $M = \begin{pmatrix} 1 & 1 \end{pmatrix}$ and $M = \begin{pmatrix} 1 & -1 \end{pmatrix}$, respectively.

Our computational problems and algorithms take as their input torus actions as well as vectors $v, w \in \mathbb{C}^n$. The former are encoded in terms of the weight matrix M , with each entry represented in binary. The latter are assumed to be vectors $v, w \in \mathbb{Q}(i)^n$, i.e., their components are Gaussian rationals (complex numbers of the form $q + ir$ with $q, r \in \mathbb{Q}$) and given by encoding the numerators and denominators of the real and imaginary parts in binary.

Throughout the paper, we denote by B the maximum bit-length of the entries of M , and by b the maximum bit-length of the components of v and w , respectively. The parameters B and b will have different effects in the bounds.

1.3 Robust orbit problems: definitions

We will now define precisely the computational problems that we address in this paper. Given a torus action on a vector space $V = \mathbb{C}^n$ and two vectors $v, w \in V$, we wish to approximate the distance of the orbit under either the action of the algebraic torus T or the compact torus K .

We start with the latter. Because $K = (S^1)^d$ is compact, so are its orbits. Thus, if two K -orbits are distinct then they must have a finite distance with respect to any norm on V . We find it natural to use the Euclidean norm $\|v\| := \sqrt{\sum_{j=1}^n |v_j|^2}$. Given two vectors $v, w \in \mathbb{C}^n$, we denote their Euclidean distance by $\text{dist}(v, w) := \|v - w\|$ and we extend this notation to arbitrary subsets $A, B \subseteq \mathbb{C}^n$ by setting $\text{dist}(A, B) := \inf_{a \in A, b \in B} \|a - b\|$. Then we are interested in the following approximation problem, where the approximation factor γ is a parameter that may depend on d, n, b , or B .

► **Problem 1.2.** *The compact orbit distance approximation problem with approximation factor $\gamma \geq 1$ is defined as follows: Given $v, w \in (S^1)^n$, compute a number D such that*

$$\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq D \leq \gamma \text{dist}(\mathcal{C}_v, \mathcal{C}_w).$$

We now consider the algebraic torus $T = (\mathbb{C}^\times)^d$. Since it is not compact, in general neither are its orbits and hence two distinct orbits need not have any finite distance – in other words, we can have $\mathcal{O}_v \neq \mathcal{O}_w$ but $\text{dist}(\mathcal{O}_v, \mathcal{O}_w) = 0$. This can be due to two phenomena: (a) when orbits are not closed, they can still intersect in their closure; (b) even if orbits are closed, they may still become arbitrarily close at infinity. See Figure 1 for an illustration. The first phenomenon is natural: if $\overline{\mathcal{O}_v} \cap \overline{\mathcal{O}_w} \neq \emptyset$ then it can be natural to define the distance to be zero. The second phenomenon however is undesirable – clearly we would like to be able to tell apart, e.g., the two hyperbolas in Figure 1 (b)! To overcome this problem, we will instead consider a *logarithmic distance*, which always assigns a positive distance between distinct orbits, even when their closures intersect.

For simplicity, we assume that v, w have non-zero coordinates, that is, they are elements of $(\mathbb{C}^\times)^n \subseteq V$. The key idea is to “linearize” or “flatten” the group action by using the (coordinatewise) complex logarithm map:

$$\text{Log}: (\mathbb{C}^\times)^n \rightarrow \mathbb{C}^n / 2\pi i \mathbb{Z}^n, \quad v \mapsto (\log v_1, \log v_2, \dots, \log v_n), \quad (1.2)$$

which is a group isomorphism. This is natural because it converts the multiplicative action of T into an additive one. Indeed, one finds from (1.1) that Log maps any T -orbit \mathcal{O}_v to the image of the affine subspace $\text{im}(M^T) + \text{Log}(v)$ under the quotient map $\mathbb{C}^n \rightarrow \mathbb{C}^n / 2\pi i \mathbb{Z}^n$. For distinct orbits these will have a finite Euclidean distance because the corresponding subspaces are parallel (see Figure 2). This motivates using the following distance: Given $v, w \in (\mathbb{C}^\times)^n$, we define their *logarithmic distance* by

$$\delta_{\log}(v, w) := \text{dist}(\text{Log}(v) + 2\pi i \mathbb{Z}^n, \text{Log}(w) + 2\pi i \mathbb{Z}^n) := \min_{\alpha \in 2\pi i \mathbb{Z}^n} \|\text{Log}(v) - \text{Log}(w) - \alpha\|, \quad (1.3)$$

using the natural Euclidean distance on the quotient $\mathbb{C}^n / 2\pi i \mathbb{Z}^n$, and we extend this to arbitrary subsets $A, B \subseteq (\mathbb{C}^\times)^n$ as above. In particular, we have $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) = 0$ if and only if $\mathcal{O}_v = \mathcal{O}_w$, which is exactly the property that we wanted. Thus we arrive at the following approximation problem, where the approximation factor γ may depend on d, n, b , or B as above.³

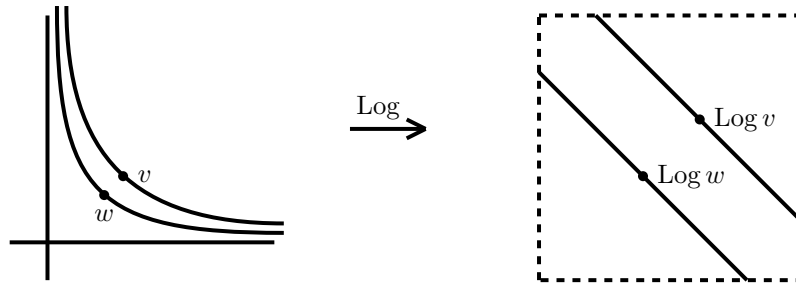
► **Problem 1.3.** *The algebraic orbit distance approximation problem with approximation factor $\gamma \geq 1$ is defined as follows: Given $v, w \in (\mathbb{C}^\times)^n$, compute a number D such that*

$$\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \leq D \leq \gamma \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w).$$

1.4 Robust orbit problems: hardness

The appearance of a lattice in the above suggests that there might be some hardness lurking behind this problem. In fact, we find that *both* orbit distance approximation problems are NP-hard for a sufficiently small approximation factor. We will prove the following in Section 6 by showing that there is a polynomial time reduction from the *closest vector problem* (CVP) to Problems 1.2 and 1.3.

³ One can also use the metric δ_{\log} in definition of Problem 1.2 in place of dist . However, this is essentially equivalent, since for $v, w \in (S^1)^n$ the two metrics are related by $\frac{2}{\pi} \delta_{\log}(v, w) \leq \text{dist}(v, w) \leq \delta_{\log}(v, w)$, see Section 3.



■ **Figure 2** The logarithm maps the “hyperbolic” orbits (left) into parallel “lines” (right), which have a finite Euclidean distance. We define the logarithmic distance of the former as the Euclidean distance of the latter.

► **Theorem 1.4.** *There is a constant $c > 0$ such that Problems 1.2 and 1.3 for $\gamma = n^{c/\log \log n}$ are NP-hard.*

We note that a solution to Problem 1.2 allows distinguishing between the cases $\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq \varepsilon$ and $\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \geq \gamma\varepsilon$ on input ε , and similarly for Problem 1.3. The CVP problem has recently attracted significant interest due to its relevance to *lattice-based cryptosystems*, which are conjectured to be secure against quantum computers [37, 47, 48, 35].

Interestingly, our reduction from CVP is not straightforward, but rather uses a quantitative lattice lifting result in the spirit of [23] that might be of independent interest. We sketch the main idea for the K -action and for δ_{\log} instead of the Euclidean distance (which by Footnote 3 makes no difference). Let $\theta := \frac{1}{2\pi i} \text{Log}(v)$ and $\phi := \frac{1}{2\pi i} \text{Log}(w)$. Then,

$$\frac{1}{2\pi} \delta_{\log}(\mathcal{C}_v, \mathcal{C}_w) = \text{dist}(P(\theta - \phi), P(\mathbb{Z}^n)), \quad (1.4)$$

where P denotes the orthogonal projection from \mathbb{R}^n onto the orthogonal complement of the row space of M . Note that the right-hand side quantity amounts to a CVP for the lattice $P(\mathbb{Z}^n)$. Our lattice lifting result states that, up to scaling, *every* full-dimensional lattice \mathcal{L} can be obtained as the orthogonal projection of a cubic lattice \mathbb{Z}^n , where n is not much larger than $\dim(\mathcal{L})$. Moreover, this projection can be computed in polynomial time. This yields the desired reduction from CVP.

► **Theorem 1.5 (Lattice lifting).** *Suppose $\mathcal{L} \subset \mathbb{R}^m$ is a lattice of rank m given by a generator matrix $G \in \mathbb{Z}^{m \times m}$, i.e., $\mathcal{L} = G(\mathbb{Z}^m)$. Then we can, in polynomial time, compute $n \geq m$, a scaling factor $s \in \mathbb{Z}_{>0}$, and an orthonormal basis $v_1, v_2, \dots, v_n \in \mathbb{Q}^n$ such that*

$$\mathcal{L} = sP(\mathcal{L}'), \quad \text{where } \mathcal{L}' := \mathbb{Z}v_1 + \mathbb{Z}v_2 + \dots + \mathbb{Z}v_n$$

and $P: \mathbb{R}^n \rightarrow \mathbb{R}^m$ denotes the orthogonal projection onto the first m coordinates. Moreover, we may assume that $n = O(m \log m + m \log \langle G \rangle)$, where $\langle G \rangle$ denotes the bit-length of G .

To the best of our knowledge, Theorem 1.5 was not previously observed in the literature. We sketch its proof in Section 1.7 below and give the full proof in Section 6.2.

1.5 Robust orbit problems: algorithms

The picture changes markedly if we allow for larger approximation ratios γ , where we recall that B denotes the maximum bit-length of the entries of M . Roughly speaking, our intuitive result is the following:

If the distance between any two distinct orbits is no smaller than $\exp(-\text{poly}(d, n, B, b))$, then these distances can be approximated to $\gamma = \exp(\text{poly}(d, n, B))$ in polynomial time.

To state this precisely, let us define the *compact* and the *algebraic separation parameters* as

$$\text{sep}_K(d, n, B, b) := \min_{M, v, w} \text{dist}(\mathcal{C}_v, \mathcal{C}_w) \quad \text{and} \quad \text{sep}_T(d, n, B, b) := \min_{M, v, w} \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w), \quad (1.5)$$

respectively, where the minima are taken over all weight matrices $M \in \mathbb{Z}^{d \times n}$ with entries of maximum bit-length at most B and over all vectors $v, w \in (\mathbb{Q}(i)^\times)^n$ with components of bit-length at most b such that v and w are in distinct orbits.

► **Separation Hypothesis 1.6** (for the compact torus).

$$\text{sep}_K(d, n, B, b) \geq \exp(-\text{poly}(d, n, b, B)).$$

► **Separation Hypothesis 1.7** (for the algebraic torus).

$$\text{sep}_T(d, n, B, b) \geq \exp(-\text{poly}(d, n, b, B)).$$

The latter hypothesis is readily seen to imply the former (this follows essentially from Footnote 3, see Corollary 3.9). While both hypotheses appear geometric in nature, it turns out that they are intimately related to well-known quantitative versions of the *abc*-conjecture in number theory. We will explain this connection in Section 1.6. Assuming these hypotheses, we can solve the orbit distance approximation problems with an exponential approximation factor:

► **Theorem 1.8.** *If Separation Hypothesis 1.6 holds, then there is a polynomial-time algorithm that solves Problem 1.2 for an approximation factor $\gamma = \exp(\text{poly}(d, n, B))$. Similarly, if Separation Hypothesis 1.7 holds, then there is a polynomial-time algorithm that solves Problem 1.3 for $\gamma = \exp(\text{poly}(d, n, B))$.*

In fact, we show in Section 5 without any hypotheses that for $\gamma = \exp(\text{poly}(d, n, B))$, there is an algorithm solving Problem 1.3 in time $O(\text{poly}(d, b, n, B) \log(\text{sep}_T^{-1}(d, n, B, b)))$, and similarly for Problem 1.2. This algorithm runs in polynomial time if and only if the separation hypothesis is true, see Remark 5.11. Our algorithm also solves the corresponding *search problem* (see Theorem 5.4): When $\mathcal{O}_v \neq \mathcal{O}_w$, it finds a group element $t \in T$ such that

$$\delta_{\log}(t \cdot v, w) \leq \gamma \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w). \quad (1.6)$$

In particular, t can serve as a witness of the approximate distance of the orbits. In general, the coordinates of any t satisfying Equation (1.6) may require a superpolynomial bit-length. Accordingly, our algorithm will output instead a vector $x \in \mathbb{Q}(i)^d$ such that $t = \text{Exp}(x)$. When $\mathcal{O}_v = \mathcal{O}_w$, then it will in general not be possible to output such a group element, since $t \cdot v = w$ will in general not have a rational solution (neither in t nor in x).

To explain the main idea behind proving Theorem 1.8 and motivate the separation hypothesis, we recall the algorithm for solving the orbit equality problem in [16]. For simplicity, we consider vectors $v, w \in (\mathbb{C}^\times)^n$ with closed T -orbits. The algorithm in [16] first computes a basis \mathcal{B} of the lattice $\mathcal{K} := \{\alpha \in \mathbb{Z}^n \mid M\alpha = 0\}$ in polynomial time. Each basis vector $\alpha \in \mathcal{B}$ determines an invariant function (Laurent monomials) of the form

$$f_\alpha(x) := x_1^{\alpha_1} x_2^{\alpha_2} \dots x_n^{\alpha_n}.$$

Then, $\mathcal{O}_v = \mathcal{O}_w$ if and only if v and w take the same value on all these functions [16, Corollary 5.2]. Even though the α have polynomial bit-length, testing whether $f_\alpha(v) = f_\alpha(w)$ is not obvious, as the bit-length of the evaluations can be exponentially large in the input size. Nevertheless, it is possible to determine equality of these numbers in polynomial time without actually computing them, see [16, Proposition 5.5].

When dealing with the problem of approximating orbit distances, one may use similar ideas. One finds that one now needs to test whether $f_\alpha(v)$ and $f_\alpha(w)$ are *close* (rather than equal), for a suitable choice of a metric. This turns out to be substantially more difficult than the equality testing,⁴ and indeed it must be by the hardness results in Section 1.4. To solve it with an exponential approximation factor we can proceed as follows: Let $H \in \mathbb{Z}^{k \times n}$ denote the matrix with rows the vectors in \mathcal{B} . We prove in Section 3 that the logarithmic distance between the orbits \mathcal{O}_v and \mathcal{O}_w can be approximated by $\|H \operatorname{Log}(v) - H \operatorname{Log}(w)\|$, up to a factor $\sigma_{\max}(H)/\sigma_{\min}(H) = \exp(\operatorname{poly}(d, n, B))$. Here, $\sigma_{\max}(H), \sigma_{\min}(H)$ denote the largest and smallest singular values of H , respectively. It remains to approximate $\|H \operatorname{Log}(v) - H \operatorname{Log}(w)\|$ with relative error. The key challenge is in computing rational approximations of the logarithms. This can be done by standard algorithms, but we have to make sure that polynomially many bits of accuracy are sufficient. This is guaranteed by the separation hypothesis! The idea of relating the distance between two orbits with the difference between the values that the invariants take on these orbits, and approximating the distance between the orbits in this way is inspired by the invariant theory and it will be important later when we explain the close relationship between the robust orbit problems and the *abc*-conjecture. On the other hand, we will also develop an alternative algorithm to the robust orbit problems in Section 6.1 by providing a polynomial time reduction to CVP under the assumption that Separation Hypothesis 1.7 is true by using Equation (1.4).

Finally, we study the complexity of the orbit distance approximation problems when n is fixed. In this situation, Separation Hypotheses 1.6 and 1.7 are true, as we will discuss in Section 1.6, and hence there are polynomial-time algorithms for Problems 1.2 and 1.3 with $\gamma = \exp(\operatorname{poly}(d, B))$. However, we can in fact choose γ to be much smaller:

► **Theorem 1.9.** *For fixed n , Problem 1.2 can be solved in polynomial time for the approximation factor $\gamma = 2$.*

► **Theorem 1.10.** *For fixed n , and any fixed $\gamma > 1$, Problem 1.3 can be solved in polynomial time. More specifically, there is a polynomial-time algorithm that on input $M \in \mathbb{Z}^{d \times n}$, $v, w \in \mathbb{Q}(i)^n$, and $\varepsilon > 0$ outputs a number D such that $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \leq D \leq (1 + \varepsilon) \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w)$.*

We prove these results at the end of Section 6.1.

1.6 Separation hypotheses and the *abc*-conjecture

The famous *abc*-conjecture by Oesterlé and Masser [75] states that for every $\varepsilon > 0$ there exists $\kappa_\varepsilon > 0$ such that, if a and b are coprime positive integers and $c = a + b$, then

$$c < \kappa_\varepsilon \operatorname{rad}(abc)^{1+\varepsilon},$$

where $\operatorname{rad}(abc)$ is the *radical* of abc , i.e., the product of all primes dividing abc . Motivated by an earlier conjecture of Szpiro, the *abc*-conjecture is one of the most important open problems in number theory, due to its many consequences [38, 88].

Baker [8] observed that the *abc*-conjecture is intimately related to lower bounds for linear forms in logarithms. Let v_1, v_2, \dots, v_n be positive integers and e_1, e_2, \dots, e_n be integers such that the product $v_1^{e_1} v_2^{e_2} \dots v_n^{e_n} \neq 1$ is different from one. How close to 1 can such a product be? Equivalently, how close can the linear combination of logarithms,

$$\Lambda(v, e) := e_1 \log v_1 + \dots + e_n \log v_n, \tag{1.7}$$

⁴ Just testing an inequality $f_\alpha(v) \geq f_\alpha(w)$ in polynomial time is only known assuming a certain strengthening of the *abc*-conjecture; see [30].

be to zero? We note that if $|v_1^{e_1} \dots v_n^{e_n} - 1| \leq \frac{1}{2}$, then

$$\frac{1}{2} |\Lambda(v, e)| \leq |v_1^{e_1} \dots v_n^{e_n} - 1| \leq 2 |\Lambda(v, e)|,$$

so lower bounds for $|\Lambda(v, e)|$ and for $|v_1^{e_1} \dots v_n^{e_n} - 1|$ are equivalent; see Equation (4.4). Improving work by Baker and Wüstholz [89], Matveev [66] proved that if $\Lambda(v, e) \neq 0$, then

$$|\Lambda(v, e)| \geq e^{-B \cdot b^{O(n)}}. \tag{1.8}$$

Here and below B and b denote upper bounds on the bit-length of the e_j and v_j , respectively. Note that for fixed n , this bound is polynomial in B and b . Furthermore, every lower bound for $|\Lambda(v, e)|$ implies a version of the *abc*-conjecture: Stewart, Yu and Tijdemann [82, 81] used the Baker-Wüstholz-Matveev bound (1.8) and its p -adic version by Van der Poorten [85] to prove the inequality

$$\log c < \kappa_\varepsilon \operatorname{rad}(abc)^{\frac{1}{3} + \varepsilon}.$$

To our knowledge, this bound is the strongest bound to date given towards a solution to the *abc*-conjecture and demonstrates the strong connection between the *abc*-conjecture and lower bounds for linear forms in logarithms. Baker [8] proved that a stronger lower bound than (1.8) (together with its p -adic version) is equivalent to a certain strengthening of the *abc*-conjecture. We will say more about this connection in Section 4.

It is widely conjectured that (1.8) is *not* optimal. In particular, famous conjectures in number theory, such as Waldschmidt’s conjectures [86, Conjecture 14.25, p. 547], [87, Conjecture 4.14], or the Lang-Waldschmidt conjecture for Gaussian rationals [59, Introduction to Chapters X and XI] imply that the following hypothesis holds (in fact, they make even stronger predictions!):

► **Number-Theoretic Hypothesis 1.11.** *For any $n \in \mathbb{Z}_{\geq 0}$, $v_1, v_2, \dots, v_n \in \mathbb{Q}(i)^\times$, and $e_1, e_2, \dots, e_n \in \mathbb{Z}$ such that, with Log the principal branch of the complex logarithm,*

$$\Lambda(v, e) := e_1 \operatorname{Log} v_1 + e_2 \operatorname{Log} v_2 + \dots + e_n \operatorname{Log} v_n \neq 0,$$

we have

$$|\Lambda(v, e)| \geq \exp(-\operatorname{poly}(n, b, B)),$$

where B and b are the maximum bit-lengths of the e_j and v_j , respectively.

Here we prove a novel connection between this number-theoretic conjecture and the separation hypothesis of group orbits, which can be seen as further evidence for the latter:

► **Theorem 1.12.** *Separation Hypothesis 1.7 and Number-Theoretic Hypothesis 1.11 are equivalent. Moreover, Separation Hypothesis 1.6 is equivalent to Number-Theoretic Hypothesis 1.11 when the latter is restricted to $|v_1| = \dots = |v_n| = 1$.*

We prove this in Section 4 by using similar ideas as sketched in Section 1.5. There, we also show that Matveev’s bound (1.8) implies the lower bound $\operatorname{sep}_T(d, n, B, b) \geq e^{-B^{O(1)} \cdot b^{O(n)}}$, which is only exponentially small for constant n .

1.7 Proof sketch of the lattice lifting theorem

We now give a summary of the idea behind the proof of Theorem 1.5. We call a collection u_1, u_2, \dots, u_n of vectors in \mathbb{R}^m , $n \geq m$, an *eutactic star of scale s* if each u_j can be extended to a vector $v_j \in \mathbb{R}^n$ such that the v_j are pairwise orthogonal and of the same norm s . Note that $u_i = P(v_i)$ where P denotes the projection onto the first m coordinates. Let $X \in \mathbb{R}^{m \times n}$ denote the matrix whose i -th column is u_i . It is easy to see that the collection u_1, u_2, \dots, u_n forms an eutactic star of scale s iff $XX^T = s^2 I_m$.

Assume that the lattice $\mathcal{L} \subset \mathbb{R}^m$ is given as the \mathbb{Z} -span of the columns of the matrix $G \in \mathbb{Z}^{m \times m}$. Our aim is to understand when \mathcal{L} is generated (as a lattice) by an eutactic star. We first note that the collection u_1, u_2, \dots, u_n is contained in \mathcal{L} iff there exists an integer matrix $L \in \mathbb{Z}^{m \times n}$ such that $X = GL$. To enforce further that the vectors u_i generate \mathcal{L} as a lattice, we require L to be *right invertible*, meaning there exists $R \in \mathbb{Z}^{n \times m}$ such that $LR = I_m$. In Proposition 6.8 we prove:

$$\begin{aligned} \mathcal{L} \text{ is generated by an eutactic star of scale } s \in \mathbb{Z}_{>0} &\iff \\ &\exists \text{ right invertible } L \in \mathbb{Z}^{m \times n} \text{ s.t. } (GL)(GL)^T = s^2 I_m. \end{aligned}$$

Therefore, writing \mathcal{L} as in Theorem 1.5 amounts to finding such L and s . In Section 6.3 we show that for given $G \in \mathbb{Z}^{m \times m}$, such L and s can be computed in polynomial time. For this, we choose the integer s so large that the matrix

$$A := s^2(G^{-1}G^{-T}) - I_m.$$

is positive definite. In Theorem 6.10 we prove that a decomposition of the form $A = (L')(L')^T$ can be computed in polynomial time (efficient Waring decomposition). Note that this amounts to writing the given positive definite quadratic form $x^T Ax$ as sum of squares of rational linear forms. Such decompositions are known to exist if $n = m + 3$ [69], and randomized polynomial time algorithms computing them are available [77]. However, to the best of our knowledge, it is open whether such small decompositions can be computed in deterministic polynomial time. Instead, we will use the simple Lemma 6.12, which writes an integer D as a sum of $O(\log \log D)$ many squares, but can be easily shown to run in deterministic polynomial time. Finally, one checks that $L := [I_m \quad L']$ satisfies $(GL)(GL)^T = s^2 I_m$, which is what we wanted to get.

1.8 The method of Kempf-Ness

A major challenge is to extend the results in this paper beyond torus actions! The Kempf-Ness theorem [55] indicates a strategy to do so. This is a general theorem on rational actions of reductive algebraic groups, that in principle allows to reduce orbit closure intersection problems to compact orbit equality problems. It states that the vectors of minimal norm in the orbit closure of $v \in V$ form a single K -orbit that we denote by \mathcal{C}_{v^*} . Here K denotes a maximal compact subgroup of a reductive algebraic group G that is assumed to act isometrically on $V = \mathbb{C}^n$. Moreover, the Kempf-Ness theorem states that for $v, w \in \mathbb{C}^n$,

$$\overline{\mathcal{O}_v} \cap \overline{\mathcal{O}_w} \neq \emptyset \iff \mathcal{C}_{v^*} = \mathcal{C}_{w^*}.$$

In an ideal world, this reduces the problem of testing $\overline{\mathcal{O}_v} \cap \overline{\mathcal{O}_w} \neq \emptyset$ to the problem of testing $\mathcal{C}_{v^*} = \mathcal{C}_{w^*}$, provided that we can compute v^* on input v . Unfortunately, even when the coordinates of v are rational, v^* may not have rational entries. Instead one may work with a numerical approximation of v^* . This was successfully carried out in [3] for the left-right action on tuples of complex matrices. As a result, polynomial time algorithms for the orbit closure problems for the left-right action were obtained along this way.

For torus actions, approximating v^* becomes a convex optimization problem. More precisely, we consider the *Kempf-Ness function* corresponding to v ,

$$f(x) := \log \|e^{x/2} \cdot v\|^2 = \log \left(\sum_{i=1}^n q_i e^{\omega_i^T x} \right),$$

where $\omega_1, \omega_2, \dots, \omega_n \in \mathbb{Z}^d$ denote the columns of M and $q_i := |v_i|^2$. We have $\min_x f(x) = 2 \log \|v^*\|$ by the definition of v^* . This function finds applications in a surprising number of different areas: In machine learning [21, 22, 46, 64], $f(x)$ is known as the *log-sum-exp* function and it is used as a smooth approximation to the piecewise affine function $L(x) := \max_i (\langle x, \omega_i \rangle + \log q_i)$: we have $L(x) \leq f(x) \leq L(x) + \log n$. The optimization of the Kempf-Ness function also has uses in the area of statistical physics: The convex program $\inf_x f(x)$ is the Lagrange dual of an entropy maximization problem with mean constraints. More precisely, one has

$$\inf_x f(x) = \sup \left\{ -D_{KL}(p||q) \mid \sum_{i=1}^n p_i \omega_i = 0, \sum_{i=1}^n p_i = 1, \forall i, p_i \geq 0 \right\}$$

where $D_{KL}(p||q) := \sum_{i=1}^n p_i \log(p_i/q_i)$ denotes the *Kullback-Leibler divergence* between the probability distribution p and (possibly non-normalized) distribution q . For more on this connection, we refer to [80, 83, 60, 20].

In Section 7, we consider the problem of approximating the optimal solution of $f(x)$. We conjecture that a point $x \in \mathbb{R}^d$ that is ε -close to $\arg \min f(x)$ in the Euclidean distance can be computed in polynomial time, see Conjecture 7.3. We show that if the conjecture is true then the logarithmic distance $\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*})$ between two Kempf-Ness orbits can be efficiently approximated. This leads to new polynomial time algorithms for the orbit equality problems, provided the separation hypothesis holds. We interpret our finding as some positive evidence towards the feasibility of the Kempf-Ness approach in more general settings, but also interpret it as a warning about the difficulties to be encountered.

1.9 Conclusion and outlook

Summarizing, we defined the problem of approximating orbit distances for torus actions and showed that these problems are NP-hard for an almost-polynomial factor, but can be solved in polynomial time within an at most single exponentially large factor under the assumption that a variant of the famous *abc*-conjecture is true. Our results point to an exciting connection between orbit problems and deep theorems and conjectures in the areas of number theory and algorithmic lattice theory.

Let us point out that our approximation algorithm for the robust orbit problem, jointly with our reduction result from CVP to this problem, yield a new polynomial time approximation algorithm for CVP, which is not based on the LLL algorithm. Unfortunately, the resulting approximation factor γ not only depends on the rank m of the lattice, but also on the bit-length of the generators of the lattice, which cannot compete with the well-known $2^{O(m)}$ -approximation algorithms for CVP (see, for example, [67, Lemma 2.12]). It is an interesting question whether competitive algorithms for CVP can be developed along this way and whether there are other uses of our lattice lifting result (Theorem 1.5). A further interesting problem is if one can remove the separation hypothesis by allowing for significantly larger approximation factors γ .

1.10 Organization of the paper

The preliminary Section 2 collects known facts from different areas that we need to establish our main results. Section 3 is devoted to an analysis of the structure of the orbits in the logarithmic space $\mathbb{C}^n/2\pi i\mathbb{Z}^n$, and the logarithmic distance metric δ_{\log} defined in Equation (1.3). We show that $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w)$ is approximated by a linear form in $\text{Log } v, \text{Log } w$. The goal of Section 4 is to explain the connection between the *abc*-conjecture and Number-Theoretic Hypothesis 1.11 in more detail and to formally prove Theorem 1.12. Section 5 describes our algorithms to solve Problem 1.3 and Problem 1.2 in polynomial time, proving Theorem 1.8. The goal of Section 6 is to prove the hardness results by reducing the closest vector problem to the orbit distance problems (Problem 1.2 and Problem 1.3). There, we also provide the proof of Theorem 1.5 on efficient lattice lifting and we prove Theorem 1.9 and Theorem 1.10. Finally, Section 7 is devoted to the analysis of the Kempf-Ness approach for general torus actions. We conjecture a result on the complexity of approximating the optimal solution of the Kempf-Ness function and based on that, we devise a numerical algorithm for deciding the equality of two orbits, which runs in polynomial time if Separation Hypothesis 1.7 is true.

2 Preliminaries

The goal of this section is to collect known facts from different areas: invariants for torus actions, singular values of matrices, the complexity of numerically computing elementary functions. We end with general observations on quotient metrics.

2.1 Notation

Throughout the paper, $T = (\mathbb{C}^\times)^d$ denotes the algebraic torus of rank d and $V = \mathbb{C}^n$. We always denote by $M \in \mathbb{Z}^{d \times n}$ a weight matrix that determines the action of T on V . The Euclidean norm of $v \in \mathbb{C}^n$ is denoted $\|v\|$. We write $\text{dist}(v, w) := \|v - w\|$ for the Euclidean distance of $v, w \in \mathbb{C}^n$ and we extend this notation to nonempty subsets $A, B \subseteq \mathbb{C}^n$ by setting $\text{dist}(A, B) := \inf_{a \in A, b \in B} \|a - b\|$.

We denote by $\Re(z)$ and $\Im(z)$ the real and imaginary part of a complex number $z \in \mathbb{C}$, respectively. Moreover, $\log(z) \in \mathbb{C}/2\pi i\mathbb{Z}$ denotes complex logarithm. The componentwise defined logarithm and exponentiation functions are denoted by $\text{Log}: (\mathbb{C}^\times)^n \rightarrow \mathbb{C}^n/2\pi i\mathbb{Z}^n$ and $\text{Exp}: \mathbb{C}^n/2\pi i\mathbb{Z}^n \rightarrow (\mathbb{C}^\times)^n$.

The letters v, w denote vectors in V , while the Greek letters η, ζ refer to vectors that are exponentiated, as in $v = \text{Exp}(\eta)$. We always denote by b a bound for the bit-lengths of the components of the input vectors $v, w \in \mathbb{Q}(i)^n$ (or η, ζ), and B always denotes a bound for the bit-length of the entries of the weight matrix $M \in \mathbb{Z}^{d \times n}$.

2.2 Invariant theory of torus actions

In this section, we present a brief summary of the setting and the results of [16]. We will discuss the proof strategy sketched in Section 1.5 in more detail. As always, $T = (\mathbb{C}^\times)^d$ is the d -dimensional torus and $V = \mathbb{C}^n$.

Any rational action of T on V , up to some base change, can be simultaneously diagonalized and brought to the form Equation (1.1). The matrix $M = [M_{ij}] \in \mathbb{Z}^{d \times n}$ occurring in Equation (1.1) is called the *weight matrix* and its columns $\omega_1, \omega_2, \dots, \omega_n \in \mathbb{Z}^d$ are called the *weights* of the action. There is an induced action of T on the polynomial ring $\mathbb{C}[x_1, x_2, \dots, x_n]$, given by

$$(t \cdot f)(v) := f(t^{-1} \cdot v), \quad f \in \mathbb{C}[x_1, x_2, \dots, x_n], \quad t \in T, \quad v \in V.$$

A *polynomial invariant* is a fixed point of this action, i.e., $t \cdot f = f$ for all $t \in T$. Note that a polynomial is an invariant iff it is constant along orbits. We denote the ring of polynomial invariants by $\mathbb{C}[V]^T$. For a monomial $x^\alpha := x^{\alpha_1} x^{\alpha_2} \dots x^{\alpha_n} \in \mathbb{C}[x_1, x_2, \dots, x_n]$, we have

$$x^\alpha(t \cdot v) = t^{M\alpha} x^\alpha(v), \tag{2.1}$$

Equation (2.1) implies that the line $\mathbb{C}x^\alpha$ is preserved by the action of T . Moreover, x^α is an invariant iff $M\alpha = 0$. Hence a polynomial f is an invariant if and only if each monomial appearing in f is invariant. In particular, the space of invariant polynomials is linearly spanned by the invariant monomials.

The action of T leaves $X := (\mathbb{C}^\times)^n$ invariant. We have an induced action of T on the algebra

$$\mathbb{C}[X] = \mathbb{C}[x_1, x_1^{-1}, x_2, x_2^{-1}, \dots, x_n, x_n^{-1}],$$

of Laurent polynomials, the ring of regular functions $\mathbb{C}[X]$ of X , which is easier to study. The above observations extend: Equation (2.1) also holds for *Laurent monomials* x^α with exponent vector $\alpha \in \mathbb{Z}^n$, which is thus invariant iff $M\alpha = 0$. The space of invariant Laurent polynomials is linearly spanned by the invariant Laurent monomials. We call

$$\mathcal{K} := \{\alpha \in \mathbb{Z}^n \mid M\alpha = 0\}, \tag{2.2}$$

the *lattice of rational invariants* defined by M . Note that the rank of \mathcal{K} is given by $k = n - \text{rk } M$.

Thus $\alpha_1, \alpha_2, \dots, \alpha_k$ is a lattice basis of \mathcal{K} , then the Laurent monomials $x^{\alpha_1}, x^{\alpha_2}, \dots, x^{\alpha_k}$ generate $\mathbb{C}[X]^T$ as an algebra. Then, for $v, w \in (\mathbb{C}^\times)^n$, we have (see [16, Proposition 4.1])

$$\mathcal{O}_v = \mathcal{O}_w \iff \forall i \in [k] \quad x^{\alpha_i}(v) = x^{\alpha_i}(w). \tag{2.3}$$

There is an analogous result for the orbits of the compact torus $K = (S^1)^d$ (see [16, Proposition 8.1]): For two vectors $v, w \in (\mathbb{C}^\times)^n$, we have

$$\begin{aligned} \mathcal{C}_v = \mathcal{C}_w &\iff \forall \alpha \in \mathcal{K}, \quad x^\alpha(v) = x^\alpha(w) \text{ and } \forall i \in [n], |v_i| = |w_i| \\ &\iff \mathcal{O}_v = \mathcal{O}_w \text{ and } \forall i \in [n], |v_i| = |w_i|. \end{aligned} \tag{2.4}$$

The next theorem (see [16, Corollary 4.4 and Proposition 5.5]) shows that one can decide $\mathcal{O}_v = \mathcal{O}_w$ in polynomial time.

► **Theorem 2.1.**

- (a) We can compute in $\text{poly}(d, n, B)$ -time a basis for the lattice \mathcal{K} of T -invariant Laurent monomials. In particular, basis elements of \mathcal{K} have bit-lengths bounded by $\text{poly}(d, n, B)$.
- (b) Suppose $v, w \in (\mathbb{Q}(\mathbf{i})^\times)^n$ and $\alpha \in \mathbb{Z}^n$. Then in $\text{poly}(n, b, \langle \alpha \rangle)$ -time one can decide whether $x^\alpha(v) = x^\alpha(w)$.

Let $H \in \mathbb{Z}^{k \times n}$ be a matrix whose rows $\alpha_1^T, \alpha_2^T, \dots, \alpha_k^T$ form a basis of the lattice \mathcal{K} defined by M , see Equation (2.2). We call H a *matrix of rational invariants*.

► **Proposition 2.2.** We have $\ker H = \text{im } M^T$ and $\text{rk } H = k$. Moreover, $H(\mathbb{Z}^n) = \mathbb{Z}^k$, i.e., for every $\beta \in \mathbb{Z}^k$, there exists $\alpha \in \mathbb{Z}^n$ such that $H\alpha = \beta$. If the rows of H are produced from M by a polynomial time algorithm (as in Theorem 2.1), then the bit-length of H is at most $\text{poly}(d, n, B)$.

Proof. The first claim is obvious from the construction of \mathcal{K} and H . The second claim is obvious. For the third claim, we are going to use the integral analogue of Farkas' lemma (see [79, Corollary 4.1a]), which states that for $H \in \mathbb{Z}^{k \times n}, \beta \in \mathbb{Z}^k$,

$$H\alpha = \beta \text{ has a solution } \alpha \in \mathbb{Z}^n \iff \forall \gamma \in \mathbb{Q}^k \text{ with } H^T \gamma \in \mathbb{Z}^n, \text{ it holds that } \beta^T \gamma \in \mathbb{Z}.$$

To reach a contradiction, suppose $H(\mathbb{Z}^n) \neq \mathbb{Z}^k$, i.e., there exists a vector $\beta \in \mathbb{Z}^k \setminus H(\mathbb{Z}^n)$. Then there exists a rational vector $\gamma \in \mathbb{Q}^k$ such that $H^T \gamma \in \mathbb{Z}^n$ but $\beta^T \gamma \notin \mathbb{Z}$.

We note that if $H^T \gamma \in \mathbb{Z}^n$ for some $\gamma \in \mathbb{Q}^k$, then $H^T \gamma \in \mathcal{K}$ since $MH^T = 0$. Moreover, since the rows of H generate \mathcal{K} , we further have $H^T \gamma = H^T \tilde{\gamma}$ for some integral vector $\tilde{\gamma} \in \mathbb{Z}^k$. However, $\text{rk } H = k$ so we must have $\gamma = \tilde{\gamma}$, so γ is integral. This contradicts $\beta^T \gamma \notin \mathbb{Z}$. ◀

► **Remark 2.3.** The property that $H(\mathbb{Z}^n) = \mathbb{Z}^k$ will be used frequently throughout the paper. We note that this is equivalent to the diagonal entries of the Smith normal form of H being all one. Moreover, the proof of Proposition 2.2 shows that $H(\mathbb{Z}^n) = \mathbb{Z}^k$ holds whenever the rows of H form a basis of a *saturated lattice* in \mathbb{Z}^n with rank k . We recall that this means for $s \in \mathbb{Z}_{>0}$ and $\alpha \in \mathbb{Z}^n$, $s\alpha \in \mathcal{K}$ implies $\alpha \in \mathcal{K}$.

► **Remark 2.4.** We could ignore the dependence of the complexity parameters on d since $d \leq n$ can be assumed without loss of generality. For seeing this, assume $d > n$ and let $A \in \mathbb{Z}^{d \times n}$ denote the (row reduced) Hermite normal form of the weight matrix $M \in \mathbb{Z}^{d \times n}$. Then the orbits with respect to the actions defined by M and by A are the same. However, since the last $d - n$ rows of A are zero, the last $d - n$ coordinates of the torus act trivially on V and can be ignored.

2.3 Singular values

Suppose $k \leq n$. For $H \in \mathbb{C}^{k \times n}$, there exist unitary matrices $X \in \text{U}(k), Y \in \text{U}(n)$ such that XHY is a diagonal matrix with non-negative real diagonal entries $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq 0$. The values σ_i are called the *singular values* of the matrix H . We denote by $\sigma_{\max}(H) := \sigma_1$ and $\sigma_{\min}(H) := \sigma_k$. We have $\sigma_{\min}(H) \neq 0$ if and only if $\text{rk } H = k$. Recall that $\|x\|$ stands for the Euclidean norm of $x \in \mathbb{C}^n$ and dist denotes the corresponding distance for subsets of \mathbb{C}^n .

The maximum and the minimum singular values $\sigma_{\max}(H), \sigma_{\min}(H)$ are characterized by the following properties:

$$\sigma_{\max}(H) = \max_{\|x\|=1} \|Hx\| \quad \text{and} \quad \forall x \quad \|Hx\| \geq \sigma_{\min}(H) \cdot \text{dist}(x, \ker H). \quad (2.5)$$

► **Lemma 2.5.** *Suppose $k \leq n$ and $H \in \mathbb{C}^{k \times n}$. For nonempty subsets $A, B \subset \mathbb{C}^n$ we have*

$$\sigma_{\min}(H) \text{dist}(A + \ker H, B + \ker H) \leq \text{dist}(HA, HB) \leq \sigma_{\max}(H) \text{dist}(A + \ker H, B + \ker H).$$

Proof. We first note that $\text{dist}(A + \ker H, B + \ker H)$ is the infimum of $\|a - b + u\|$ over $a \in A, b \in B, u \in \ker H$, and $\text{dist}(HA, HB)$ is the infimum of $\|H(a - b)\|$ over $a \in A, b \in B$.

For any $a \in A, b \in B$ and $u \in \ker H$ we have, by the equation in (2.5),

$$\text{dist}(HA, HB) \leq \|H(a - b)\| = \|H(a - b + u)\| \leq \sigma_{\max}(H) \|a - b + u\|.$$

Taking the infimum over a, b, u we have $\text{dist}(HA, HB) \leq \sigma_{\max}(H) \text{dist}(A + \ker H, B + \ker H)$.

For the other inequality we use the inequality in (2.5): For any $a \in A, b \in B$

$$\|H(a - b)\| \geq \sigma_{\min}(H) \text{dist}(a - b, \ker H) \geq \sigma_{\min}(H) \text{dist}(A + \ker H, B + \ker H).$$

Taking the infimum over a, b we get $\text{dist}(HA, HB) \geq \sigma_{\min}(H) \text{dist}(A + \ker H, B + \ker H)$. ◀

The distance $\text{dist}(A + \ker H, B + \ker H)$ depends only on $\ker H$ but not on H itself. Consequently, Lemma 2.5 gives different approximations for different matrices with the same kernels. The optimal choice is the orthogonal projection onto $\ker(H)^\perp$. The singular values of the orthogonal projection are all 1, so in this case the inequality from Lemma 2.5 becomes an equality and we get the following.

► **Corollary 2.6.** *For nonempty subsets $A, B \subset \mathbb{C}^n$ and an orthogonal projection $P : \mathbb{C}^n \rightarrow \mathbb{C}^n$, we have*

$$\text{dist}(A + \ker P, B) = \text{dist}(A + \ker P, B + \ker P) = \text{dist}(P(A), P(B)).$$

We will use Lemma 2.5 mostly in the case where H is the matrix of rational invariants defined in the previous section. In this case H is integral and the singular values can be bounded in terms of the bit-length of H .

► **Lemma 2.7.** *Suppose $k \leq n$ and $H \in \mathbb{Z}^{k \times n}$ is an integer matrix of rank $\text{rk } H = k$. Then*

$$\sigma_{\max}(H) \leq n \|H\|_{\max}, \quad \sigma_{\min}(H) \geq n^{-(n-1)} \|H\|_{\max}^{-(n-1)},$$

where $\|H\|_{\max} := \max_{i,j} |H_{ij}|$ is the max norm of H . Consequently, if $B := \langle H \rangle$ is the bit-length of H , then

$$\kappa(H) := \sigma_{\max}(H) / \sigma_{\min}(H) \leq n^n 2^{Bn}.$$

Given an H as above, one can compute in polynomial time a number $D \in \mathbb{Q}$ such that $\sigma_{\min}(H) \leq D \leq 2\sigma_{\min}(H)$.

Proof. The upper bound on $\sigma_{\max}(H)$ follows from $\|Hx\| \leq \sqrt{nk} \|H\|_{\max} \|x\|$. For the lower bound, note that the product of the singular values of H equals $\prod_{i=1}^k \sigma_i(H) = \sqrt{\det(HH^T)}$. Since H has rank k , the matrix HH^T is invertible and positive definite. Hence, $\det(HH^T) \geq 1$ and we deduce with part one that

$$\sigma_{\min}(H) \geq \sigma_{\max}(H)^{1-k} \sqrt{\det(HH^T)} \geq n^{-(k-1)} \|H\|_{\max}^{-(k-1)} \geq n^{-(n-1)} \|H\|_{\max}^{-(n-1)},$$

which shows the second inequality.

We refer to [76] for the algorithmic claim on computing $\sigma_{\min}(H)$. ◀

2.4 Complexity of elementary functions

Two important functions used in this paper are the complex logarithm and exponentiation, \log and \exp . Both functions are transcendental and rarely assume rational values on rational inputs. Fortunately, they can be efficiently approximated by rational functions with the arithmetic mean-geometric mean iteration of Gauss, Lagrange and Legendre. We refer to the paper [13] and the book [12] for a detailed study of the AM-GM iteration and for the following results.

► **Lemma 2.8.** *Suppose \log denotes the standard branch of the complex logarithm whose imaginary part satisfies $\Im(\log(z)) \in (-\pi, \pi]$, and Log is the componentwise logarithm. Assume that $v \in (\mathbb{Q}(i)^\times)^n$ is a vector with Gaussian rational entries. Given $\varepsilon \in \mathbb{Q}_{>0}$, in $\text{poly}(n, \langle v \rangle, \log \varepsilon^{-1})$ -time we can compute an approximation to $\text{Log } v$ with absolute error ε . That is, we can compute a vector $\eta \in \mathbb{Q}^n + 2\pi i \mathbb{Q}^n$ such that*

$$\|\text{Log } v - \eta\| < \varepsilon.$$

► **Lemma 2.9.** *Suppose \exp denotes the complex exponentiation, $\exp(\rho + i\theta) = e^\rho(\cos(\theta) + i\sin(\theta))$ and Exp is the componentwise exp. Assume that $\eta \in \mathbb{Q}^n + 2\pi i\mathbb{Q}^n$. Given $\varepsilon \in \mathbb{Q}_{>0}$, in $\text{poly}(n, \langle \eta \rangle, \log \varepsilon^{-1})$ -time we can compute an approximation to $\text{Exp}(\eta)$ with relative error ε . That is, we can compute a vector $v \in (\mathbb{Q}(i)^\times)^n$ such that*

$$\frac{\|\text{Exp}(\eta) - v\|}{\|\text{Exp}(\eta)\|} < \varepsilon.$$

2.5 Quotient topology and quotient metric

Throughout the paper, we consider various group actions on various metric spaces. The orbit space, i.e., the set of orbits of the action, has a natural quotient topology and it is an important question for us to decide when it is possible to carry over the metric space structure to the orbit space. This is possible in all cases we consider in this section.

► **Definition 2.10.** *Suppose X is a topological space and \sim is an equivalence relation on X . We denote by $X/\sim := \{[x] \mid x \in X\}$ the set of equivalence classes of X , where $[x]$ is the equivalence class of x . Let $\pi : X \rightarrow X/\sim$ be the canonical projection map. The quotient topology on X/\sim is defined by calling a subset U of X/\sim open iff $\pi^{-1}(U)$ is open in X .*

Assume that (X, δ) is a metric space with distance function δ . We define the induced distance function on X/\sim by

$$\delta([x], [y]) := \inf\{\delta(x', y') \mid x \sim x', y \sim y'\}.$$

Clearly, this is a *pseudo-metric* on X/\sim , which means that $\delta(x, x) = 0$, $\delta(x, y) = \delta(y, x)$ and $\delta(x, z) \leq \delta(x, y) + \delta(y, z)$ for all $x, y, z \in X$. Moreover, one can show that δ is *compatible* with the quotient topology on X/\sim , which means that the balls $\{[x] \mid \delta([x], [y]) < r\}$ form a basis of the quotient topology on X/\sim , see [45, Theorem 4]. A pseudo-metric is a metric iff $\delta(x, y) = 0$ implies $x = y$.

We note that the induced distance is not always a metric. Indeed, the quotient topology on X/\sim does not need to be Hausdorff. (The so-called line with two origins is an example for this.)

However, in the following two cases, the pseudo-metric $\delta([x], [y])$ is indeed a metric.

► **Proposition 2.11.** *Suppose (X, δ) is a metric space and K is a compact topological group acting continuously on X . For $x \in X$, we denote by \mathcal{C}_x the orbit of x . Then $\delta(\mathcal{C}_x, \mathcal{C}_y)$ defines a metric on the space of orbits $X/K := \{\mathcal{C}_x \mid x \in X\}$, which is compatible with the quotient topology.*

Proof. The orbit \mathcal{C}_x is compact since it is the image of K under the continuous map $k \mapsto k \cdot x$. By the above observation, $\text{dist}(\mathcal{C}_x, \mathcal{C}_y)$ defines a compatible pseudo-metric on X/K . Different orbits $\mathcal{C}_x, \mathcal{C}_y$ are disjoint. Since they are compact, they have a positive distance: $\text{dist}(\mathcal{C}_x, \mathcal{C}_y) > 0$. This shows that the induced distance is a metric. ◀

For the second example, let V be a finite dimensional real vector space and $\mathcal{L} \subseteq V$ be a lattice. This means that \mathcal{L} is the set of \mathbb{Z} -linear combinations of a collection of linearly independent vectors in V . Alternatively, a lattice \mathcal{L} can be defined as an additive subgroup of V that is *discrete*, i.e., there exists a positive constant $\varepsilon > 0$ such that for any distinct lattice points $x, y \in \mathcal{L}$ we have $\text{dist}(x, y) \geq \varepsilon$. Thus, any convergent sequence of lattice points must be eventually constant.

► **Proposition 2.12.** *Let V be a finite dimensional real vector space, \mathcal{L} a lattice in V , and δ a translation invariant metric on V that is compatible with the standard topology on V . Then $\delta(x + \mathcal{L}, y + \mathcal{L})$ defines a metric on the quotient space V/\mathcal{L} which is compatible with the quotient topology.*

Proof. We argue as in the proof of Proposition 2.11. Suppose that $\delta(x + \mathcal{L}, y + \mathcal{L}) = 0$. Then there exist sequences u_i and v_i in \mathcal{L} such that $\delta(x + u_i, y + v_i) = \delta(x - y, v_i - u_i)$ converges to zero. Since the sequence $v_i - u_i$ of lattice points converges, it must be eventually constant. Hence $x - y \in \mathcal{L}$, which completes the proof. ◀

Sometimes in the paper the lattice \mathcal{L} will be given as an orthogonal projection of another lattice. Note that the orthogonal projection of a lattice is not always a lattice. (For instance, the orthogonal projection of \mathbb{Z}^2 onto $\mathbb{R}(1, y)$ with irrational y is not discrete.)

► **Lemma 2.13.** *Suppose $U \subset V$ is a rational subspace of $V = \mathbb{R}^n$, spanned by linearly independent rational vectors $u_1, \dots, u_k \in \mathbb{Q}^n$ and $\mathcal{L} \subset V$ is a lattice, generated by integral vectors. Then, the orthogonal projection of \mathcal{L} to U is a lattice.*

Proof. Using Gram-Schmidt orthogonalization, we may assume that the u_i are pairwise orthogonal. Then the orthogonal projection $P : \mathbb{R}^n \rightarrow U$ is given by $P(v) = \sum_{i=1}^k \frac{\langle u_i, v \rangle}{\langle u_i, u_i \rangle} u_i$. This shows that $P(\mathbb{Q}^n) \subset \mathbb{Q}^n$. Therefore, if $v_1, v_2, \dots, v_l \in \mathbb{Z}^n$ generate \mathcal{L} , there is a positive integer N such that $N P(v_i) \in \mathbb{Z}^n$ for all i . Hence $N P(\mathcal{L}) \subset \mathbb{Z}^n$, which implies that $P(\mathcal{L})$ is discrete. ◀

3 Logarithmic image of orbits

The goal of this section is to discuss the structure of the logarithmic image of orbits in the quotient space $\mathbb{C}^n/2\pi i\mathbb{Z}$, to study the metric δ_{\log} defined in (1.3), and to prove that $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w)$ is approximated by a linear form in $\text{Log } v$ and $\text{Log } w$.

We assume the setting of Section 2.2: $T = (\mathbb{C}^\times)^d$ acts on $V = \mathbb{C}^n$ via the weight matrix $M \in \mathbb{Z}^{d \times n}$. The columns of M are called the weights of the action and the *weight polytope* $P \subset \mathbb{R}^d$ is defined as the convex hull of these weights. We consider orbits of vectors $v \in (\mathbb{C}^\times)^n$, thus we assume that all components of v are nonzero. We note that \mathcal{O}_v is closed in $(\mathbb{C}^\times)^n$, see [16, Prop. 5.1]. However, \mathcal{O}_v may not be closed. It is well known that \mathcal{O}_v is closed iff 0 lies in the interior of P ; see [16, Section 3]. This will only become relevant later in Section 7.2.

3.1 Structure of logarithms of orbits

The key idea is to use the group isomorphism $\text{Log}: (\mathbb{C}^\times)^n \rightarrow \mathbb{C}^n/2\pi i\mathbb{Z}^n$ provided by the componentwise complex logarithm, compare Equation (1.2). The inverse is given by the (componentwise) exponential function Exp .

The action of the group T on $(\mathbb{C}^\times)^n$ induces via Log an action of T on $\mathbb{C}^n/2\pi i\mathbb{Z}^n$, which we shall denote by $*$. This action is simpler to understand, since it works by *translations*. More specifically, for $x \in \mathbb{C}^n$ and $v \in (\mathbb{C}^\times)^n$, we have

$$\text{Log}(e^x \cdot v) = \text{Log } v + M^T x, \quad (3.1)$$

see Figure 3. Note that, by the periodicity of \exp , the right-hand side in $\mathbb{C}^n/2\pi i\mathbb{Z}^n$ indeed only depends on $x \bmod 2\pi i\mathbb{Z}^n$. (We could also consider the corresponding action of the Lie algebra $\text{Lie}(T) = \mathbb{C}^d$, which has the same orbits.) This leads us to the following definition.

14:18 Complexity of Robust Orbit Problems for Torus Actions and the abc -Conjecture

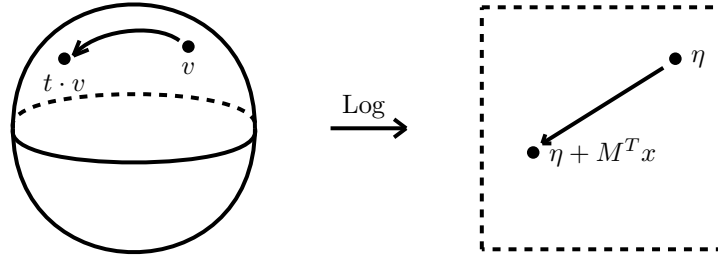
► **Definition 3.1.** We denote by $*$ the induced action of T on $\mathbb{C}^n/2\pi i\mathbb{Z}^n$ via translations defined by

$$e^x * \eta := \eta + M^T x, \quad \text{for } x \in \mathbb{C}^d, \eta \in \mathbb{C}^n/2\pi i\mathbb{Z}^n.$$

By construction, the orbits \mathcal{O}_v and \mathcal{C}_v of $v \in (\mathbb{C}^\times)^n$ are mapped to the corresponding orbits of $\eta = \text{Log } v$. If we denote

$$T * \eta := \{t * \eta \mid t \in T\} \quad \text{and} \quad K * \eta := \{k * \eta \mid k \in K\},$$

then $\text{Log}(\mathcal{O}_v) = T * \text{Log}(v)$ and $\text{Log}(\mathcal{C}_v) = K * \text{Log}(v)$.



■ **Figure 3** The logarithm *flattens* the group action. The vector $\eta = \text{Log } v$ is translated to $\eta + M^T x$ via the group element $t = e^x \in T$.

We observe that the $*$ -orbits of T (and K) are the *images of affine subspaces* under the canonical surjection $\mathbb{C}^n \rightarrow \mathbb{C}^n/2\pi i\mathbb{Z}^n$. More concretely, we denote by $U_{\mathbb{C}} := \text{im } M^T = \{M^T x \mid x \in \mathbb{C}^d\}$ the row space of M . Then it immediately follows from the definition of the $*$ -action that

$$T * \eta = \left(\eta + U_{\mathbb{C}} + 2\pi i\mathbb{Z}^n \right) / 2\pi i\mathbb{Z}^n. \quad (3.2)$$

We equip now $\mathbb{C}^n/2\pi i\mathbb{Z}^n$ with the quotient metric Δ induced by the Euclidean metric on \mathbb{C}^n , denoted by dist (by Proposition 2.12 this is indeed a metric). Explicitly,

$$\begin{aligned} \Delta(\eta, \zeta) &:= \text{dist}(\eta - \zeta, 2\pi i\mathbb{Z}^n) \\ &= \sqrt{\|\rho - \tau\|^2 + 4\pi^2 \text{dist}^2(\theta - \phi, \mathbb{Z}^n)}, \end{aligned} \quad (3.3)$$

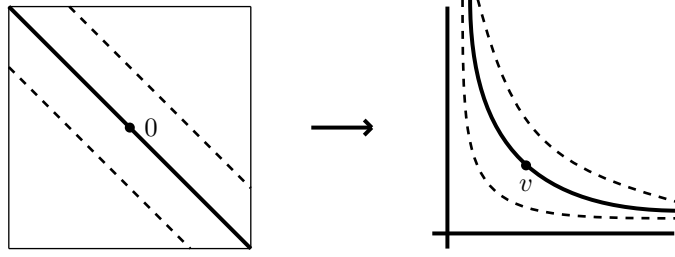
where $\eta = \rho + 2\pi i\theta$ and $\zeta = \tau + 2\pi i\phi$ with $\rho, \theta, \tau, \phi \in \mathbb{R}^n$ (note that the imaginary parts θ, ϕ are only determined modulo \mathbb{Z}^n).

► **Proposition 3.2.** The metric Δ on $X := \mathbb{C}^n/2\pi i\mathbb{Z}^n$ induces a metric

$$\Delta(T * \eta, T * \zeta) := \inf_{t \in T} \Delta(t * \eta, \zeta)$$

on the space X/T of T -orbits with respect to the $*$ -action, which is compatible with the quotient topology. An analogous result holds for K -orbits.

Proof. Denote by $P : \mathbb{C}^n \rightarrow U_{\mathbb{C}}^{\perp}$ the orthogonal projection onto the orthogonal complement of $U_{\mathbb{C}}$, thus $\ker P = U_{\mathbb{C}}$. Lemma 2.13 shows that $2\pi i P(\mathbb{Z}^n)$ is a lattice and Proposition 2.12 implies that $Y := U_{\mathbb{C}}^{\perp}/P(2\pi i\mathbb{Z}^n)$ is a metric space with respect to the quotient metric of the Euclidean metric.



■ **Figure 4** Suppose \mathbb{C}^\times acts on \mathbb{C}^2 via $t \cdot (x, y) = (tx, t^{-1}y)$ as in Figure 1. The logarithmic orbit $T * 0$ is mapped via Exp to the orbit \mathcal{O}_v of $v := (1, 1)$. The image shows the ε -neighbourhood of $T * 0$ and its image under Exp .

The projection P induces a surjective group morphism $P' : X \rightarrow Y$. From Equation (3.2) we see that P' is constant on T -orbits. More specifically, P' induces a continuous bijection $P'' : X/T \rightarrow Y$. By definition, P'' preserves the distance Δ :

$$\Delta(T * \eta, T * \zeta) = \Delta(\eta, \zeta).$$

Since Δ is a metric on Y , this implies that the pseudo-metric Δ on X/T in fact is a metric. Therefore, $\Delta(T * \eta, T * \zeta) = 0$ implies $T * \eta = T * \zeta$. The claim about K -orbits follows analogously. ◀

► **Remark 3.3.** The fact that Δ is metric on X/T is our essential gain. Note that the orbit space $(\mathbb{C}^\times)/T$ equipped with the induced Euclidean distance $\text{dist}(\mathcal{O}_v, \mathcal{O}_w)$ between T -orbits is not a metric: it can be zero even though $\mathcal{O}_v \neq \mathcal{O}_w$, as illustrated in the case of the hyperbolas in Figure 1.

Equation (3.3) implies the important equations

$$\Delta(T * \eta, T * \zeta) = \text{dist}(\eta - \zeta + U_{\mathbb{C}}, 2\pi i \mathbb{Z}^n), \quad \Delta(K * i\theta, K * i\phi) = \text{dist}(\theta - \phi + U, \mathbb{Z}^n), \quad (3.4)$$

where $U_{\mathbb{C}}$ and U are the complex and real row spaces of M , respectively.

The logarithmic distance of $v, w \in (\mathbb{C}^\times)^n$, introduced in Equation (1.3), can now be expressed as

$$\delta_{\log}(v, w) = \Delta(\text{Log } v, \text{Log } w). \quad (3.5)$$

Hence the distances of orbits in the metric δ_{\log} are given by

$$\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) = \Delta(T * \text{Log } v, T * \text{Log } w), \quad \delta_{\log}(\mathcal{C}_v, \mathcal{C}_w) = \Delta(K * \text{Log } v, K * \text{Log } w). \quad (3.6)$$

The action $*$ splits into a real and imaginary part: for $x = y + iz$ with $y, z \in \mathbb{R}^d$ and $\eta = \rho + 2\pi i\theta$ with $\rho \in \mathbb{R}^n, \theta \in \mathbb{C}^n$, we have by Definition 3.1,

$$e^x * \eta = (\rho + M^T y) + i(2\pi\theta + M^T z) = e^y * \rho + e^{iz} * (2\pi i\theta), \quad (3.7)$$

hence

$$T * \eta = \left((\rho + U) + i(2\pi\theta + U + 2\pi\mathbb{Z}^n) \right) / 2\pi i \mathbb{Z}^n.$$

Let $\zeta = \tau + 2\pi i\phi$. Together with Equations (3.3) and (3.4), we obtain the following formula for the Δ -distance between $*$ -orbits:

$$\begin{aligned} \Delta^2(T * \eta, T * \zeta) &= \text{dist}^2(\rho - \tau, U) + 4\pi^2 \Delta^2(K * i\theta, K * i\phi) \\ &= \text{dist}^2(\rho - \tau, U) + 4\pi^2 \text{dist}^2(\theta - \phi + U, \mathbb{Z}^n) \end{aligned} \quad (3.8)$$

and similarly,

$$\begin{aligned}\Delta^2(K * \eta, K * \zeta) &= \|\rho - \tau\|^2 + 4\pi^2 \Delta^2(K * i\theta, K * i\phi) \\ &= \|\rho - \tau\|^2 + 4\pi^2 \operatorname{dist}^2(\theta - \phi + U, \mathbb{Z}^n).\end{aligned}\tag{3.9}$$

The contributions of the real parts, $\operatorname{dist}^2(\rho - \tau, U)$ in the T -case and $\|\rho - \tau\|^2$ in the K -case, are easy to compute. On the other hand, the contribution of the imaginary parts $\operatorname{dist}^2(\theta - \phi + U, \mathbb{Z}^n)$, turn out to be difficult to compute. We will show in Section 6 that approximating $\operatorname{dist}(\theta - \phi + U, \mathbb{Z}^n)$ within a subpolynomial factor is NP-hard, by providing a polynomial time reduction from the *closest vector problem* (CVP) to it. By contrast, deciding whether this distance equals zero can be done in polynomial time, see Remark 3.4. (This is analogous to the CVP problem, see Section 6.1.)

► **Remark 3.4.** Given a subspace $U \subset \mathbb{R}^n$ by generators $v_1, \dots, v_m \in \mathbb{Z}^n$ and $t \in \mathbb{Q}^n$, we can decide $(t + U) \cap \mathbb{Z}^n \neq \emptyset$ in polynomial time. This can be seen by putting the matrix with columns v_1, \dots, v_m into Smith normal form, which is possible in polynomial time [54].

3.2 Approximation of orbit distances by linear forms in logarithms: T -orbits

We use now invariant theory to analyze the orbits. Let $M \in \mathbb{Z}^{k \times n}$ be a weight matrix and denote by $H \in \mathbb{Z}^{k \times n}$ a matrix of rational invariants of $H \in \mathbb{Z}^{k \times n}$, see Proposition 2.2.

For $v, w \in (\mathbb{C}^\times)^n$, Equation (2.3) characterizes the equality of orbits $\mathcal{O}_v = \mathcal{O}_w$ by $H\eta = H\zeta$, where $\eta = \operatorname{Log} v$ and $\zeta = \operatorname{Log} w$. Our goal is a robust version of this: to show that the closeness of the corresponding logarithmic orbits $T * \eta$ and $T * \zeta$, measured in terms of the metric Δ on $\mathbb{C}^n / 2\pi i \mathbb{Z}^n$, is quantitatively related to the closeness of $H\eta$ and $H\zeta$, measured in terms of the metric Δ on the quotient space $\mathbb{C}^k / 2\pi i \mathbb{Z}^k$. The correction factors are provided by the minimum and maximum singular values of H .

► **Proposition 3.5.** *If $\eta, \zeta \in \mathbb{C}^n / 2\pi i \mathbb{Z}^n$, we have for the distance of T -orbits:*

$$\sigma_{\min}(H) \Delta(T * \eta, T * \zeta) \leq \Delta(H\eta, H\zeta) \leq \sigma_{\max}(H) \Delta(T * \eta, T * \zeta).$$

For $v, w \in (\mathbb{C}^\times)^n$ we have

$$\sigma_{\min}(H) \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \leq \Delta(H \operatorname{Log} v, H \operatorname{Log} w) \leq \sigma_{\max}(H) \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w).$$

If η, ζ are purely imaginary, then the same bounds hold for the distances of K -orbits.

Proof. We have $\Delta(T * \eta, T * \zeta) = \operatorname{dist}(\eta - \zeta + U_{\mathbb{C}}, 2\pi i \mathbb{Z}^n)$ by Equation (3.4), where $U_{\mathbb{C}} = \ker H = \operatorname{im} M^T$. The first assertion follows from Lemma 2.5 with $A = \{\eta - \zeta\}$ and $B = 2\pi i \mathbb{Z}^n$. The second assertion is just a rewriting of the first, using Equation (3.6). The statement on K -orbits follows analogously. ◀

3.3 Approximation of orbit distances by linear forms in logarithms: K -orbits

Now the task is to provide an approximation for the distance $\operatorname{dist}(\mathcal{C}_v, \mathcal{C}_w)$ between K -orbits, similarly to Proposition 3.5. The analogous formula in Corollary 3.8 below is more complicated. It involves a priori upper and lower bounds on the norms of v and w . For $0 < r \leq R$ we consider the closed region defined by

$$D_{r,R} := \{v \in \mathbb{C}^n \mid \forall i \in [n] \ r \leq |v_i| \leq R\}.\tag{3.10}$$

► **Lemma 3.6.** *Let $v, w \in D_{r,R}$ and put $\eta = \text{Log } v$, $\zeta = \text{Log } w \in \mathbb{C}^n / 2\pi i \mathbb{Z}^n$. Then*

$$\frac{2r}{\pi} \Delta(\eta, \zeta) \leq \|v - w\| \leq R \Delta(\eta, \zeta).$$

Proof. It is enough to show the equality for $n = 1$. We write here $\eta = \rho + i\theta$, $\zeta = \tau + i\phi$ (dropping the factor 2π to simplify notation). The cosine theorem gives

$$|v - w|^2 = |v|^2 + |w|^2 - 2|v||w| \cos(\theta - \phi) = (|v| - |w|)^2 + 2|v||w|(1 - \cos(\theta - \phi)).$$

The first contribution on the right-hand side can be upper and lower bounded by

$$r^2(\rho - \tau)^2 \leq (|v| - |w|)^2 \leq R^2(\rho - \tau)^2$$

since the mean value theorem for exp implies (w.l.o.g. $\rho < \tau$)

$$r \leq |v| = e^\rho \leq \left| \frac{e^\tau - e^\rho}{\tau - \rho} \right| \leq e^\tau = |w| \leq R.$$

The second contribution on the right-hand side can be upper and lower bounded by

$$\frac{4r^2}{\pi^2} \text{dist}(\theta - \phi, 2\pi\mathbb{Z})^2 \leq 2|v||w|(1 - \cos(\theta - \phi)) \leq R^2 \text{dist}(\theta - \phi, 2\pi\mathbb{Z})^2,$$

using the inequality $\frac{4}{\pi^2}\psi^2 \leq 2 - 2\cos\psi \leq \psi^2$ for $\psi \in [-\pi, \pi]$. Bringing the inequalities together and observing that $4/\pi^2 \leq 1$, we obtain

$$\frac{4r^2}{\pi^2} ((\rho - \tau)^2 + \text{dist}(\theta - \phi, 2\pi\mathbb{Z})^2) \leq |v - w|^2 \leq R^2 ((\rho - \tau)^2 + \text{dist}(\theta - \phi, 2\pi\mathbb{Z})^2),$$

which completes the proof. ◀

We can now relate $\text{dist}(\mathcal{C}_v, \mathcal{C}_w)$ to $\delta_{\text{log}}(\mathcal{C}_v, \mathcal{C}_w) = \Delta(K * \text{Log } v, K * \text{Log } w)$.

► **Proposition 3.7.** *For $v, w \in D_{r,R}$ we have*

$$\frac{2r}{\pi} \Delta(K * \text{Log } v, K * \text{Log } w) \leq \text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq R \Delta(K * \text{Log } v, K * \text{Log } w).$$

Proof. Clearly, the K -action preserves $D_{r,R}$. Lemma 3.6 implies that

$$\frac{2r}{\pi} \Delta(\text{Log}(k \cdot v), w) \leq \|k \cdot v - w\| \leq R \Delta(\text{Log}(k \cdot v), w)$$

for all $k \in K$. The claim follows from by taking the infimum over $k \in K$. ◀

Combining Proposition 3.7 with Equation (3.8) and Proposition 3.5, we obtain the following corollary.

► **Corollary 3.8.** *Assume that $0 < r \leq R$ and $v, w \in D_{r,R}$. Suppose we have $\text{Log } v = \rho + 2\pi i\theta$ and $\text{Log } w = \tau + 2\pi i\phi$. Let H by a matrix of rational invariants as in Proposition 2.2. Then,*

$$\frac{4r^2}{\pi^2} (\|\rho - \tau\|^2 + \frac{4\pi^2}{\sigma_{\max}^2(H)} \Delta^2(H\theta, H\phi)) \leq \text{dist}^2(\mathcal{C}_v, \mathcal{C}_w) \leq R^2 (\|\rho - \tau\|^2 + \frac{4\pi^2}{\sigma_{\min}^2(H)} \Delta^2(H\theta, H\phi)).$$

We finally relate the separation parameters $\text{sep}_K(d, n, B, b)$ and $\text{sep}_T(d, n, B, b)$ defined in Equation (1.5).

► **Corollary 3.9.** *We have $\text{sep}_T(d, n, B, b) \leq n 2^{O(b)} \text{sep}_K(d, n, B, b)$.*

14:22 Complexity of Robust Orbit Problems for Torus Actions and the *abc*-Conjecture

Proof. Suppose that $\text{sep}_K(d, n, B, b) = \text{dist}(\mathcal{C}_v, \mathcal{C}_w)$. We let r be the minimum and R be the maximum of $|v_i|, |w_i|, i = 1, 2, \dots, n$. Clearly, $2^{-b} \leq r$. If moreover $\mathcal{O}_v \neq \mathcal{O}_w$, then Proposition 3.7 implies

$$\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \geq \frac{2r}{\pi} \Delta(K * \text{Log } v, K * \text{Log } w) \geq \frac{2r}{\pi} \Delta(T * \text{Log } v, T * \text{Log } w) \geq \frac{2r}{\pi} \text{sep}_T(d, n, B, b),$$

hence $\text{sep}_T(d, n, B, b) \leq \frac{\pi}{2r} \text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq 2^{O(b)} \text{sep}_K(d, n, B, b)$. On the other hand, if $\mathcal{O}_v = \mathcal{O}_w$, then Equation (2.4) implies that $|v_i| \neq |w_i|$ for some i . In this case, $\text{sep}_K(d, n, B, b) \geq \left| |v_i| - |w_i| \right| \geq 2^{-b}$. We note that since $|v_i/w_i| \leq 2^{2b}$, we have $\log(|v_i|/|w_i|) \leq 2b$. Furthermore, $\text{dist}(\theta, 2\pi\mathbb{Z}^n) \leq \sqrt{n}\pi$ for any $\theta \in \mathbb{R}^n$. Hence, $\text{sep}_T(d, n, B, b) \leq \Delta(\text{Log } v, \text{Log } w) \leq \sqrt{n}(2b + \pi\sqrt{n})$. Altogether,

$$\text{sep}_T(d, n, B, b) \leq 2\sqrt{n}b + \pi n \leq (\pi + 2)n2^b \leq (\pi + 2)n2^{2b} \text{sep}_K(d, n, B, b),$$

which proves the assertion. ◀

4 The separation hypotheses and the *abc*-conjecture

The goal of this section is to explain in detail the connection between the *abc*-conjecture and Number-Theoretic Hypothesis 1.11 and to prove Theorem 1.12, i.e., to show that Separation Hypothesis 1.7 and Number-Theoretic Hypothesis 1.11 are equivalent.

4.1 The *abc*-conjecture and the Number-Theoretic Hypothesis 1.11

Oesterlé and Masser’s *abc* conjecture [75, Conjecture 3] claims the following.

► **Conjecture 4.1** (*abc* conjecture). *For every $\varepsilon > 0$ there exists a constant $\kappa_\varepsilon > 0$ such that for all nonzero coprime integers a, b, c satisfying $a + b + c = 0$, we have*

$$\max(|a|, |b|, |c|) \leq \kappa_\varepsilon \text{rad}(abc)^{1+\varepsilon}.$$

The radical $\text{rad}(abc)$ is defined as the product of the distinct primes dividing abc , taken with multiplicity one.

This simple looking conjecture is one of the most powerful statements in number theory. In the words of Dorian Goldfeld [36]: “The *abc* conjecture is the most important unsolved problem in Diophantine analysis” since “it provides a way of reformulating an infinite number of Diophantine problems – and, if it is true, of solving them.”

Baker [8] proposed a more precise version of the conjecture: he conjectured that there exist absolute constants $\kappa, \kappa' > 0$ (not depending on ε) such that for all $\varepsilon > 0$

$$\max(|a|, |b|, |c|) \leq \kappa \varepsilon^{-\kappa' \omega(ab)} \text{rad}(abc)^{1+\varepsilon},$$

where $\omega(ab)$ denotes the number of distinct prime factors of ab . Moreover, Baker observed that⁵ the minimum of the right-hand side over all $\varepsilon > 0$ occurs when $\varepsilon = \omega(ab)/\log N$ and suggested the following ε -free version of the conjecture:

⁵ Baker attributes this observation to Andrew Granville.

► **Conjecture 4.2** (Baker's refinement of the *abc* conjecture, [7, Conjecture 3]). *There exist constants $\kappa, \kappa' > 0$ such that for all nonzero coprime integers a, b, c satisfying $a + b + c = 0$, we have, setting $N := \text{rad}(abc)$,*

$$\max(|a|, |b|, |c|) \leq \kappa N \left(\frac{\log N}{\omega(ab)} \right)^{\kappa' \omega(ab)}. \quad (4.1)$$

Baker's refinement of the *abc*-conjecture is closely related to lower bounds for linear forms in logarithms. The following was observed in [8]. We include a proof for the sake of completeness.

► **Theorem 4.3.** *Suppose Baker's refinement of the *abc* conjecture (Conjecture 4.2) is true. Then there exists a constant $\kappa > 0$ such that for any list of positive integers v_1, v_2, \dots, v_n and any list of integers e_1, e_2, \dots, e_n satisfying $v^e := \prod_{i=1}^n v_i^{e_i} \neq 1$, we have*

$$\log |v^e - 1| \geq -\kappa \log(\max_i |e_i|) \sum_{i=1}^n \log v_i.$$

Proof. W.l.o.g. $e_i \neq 0$ for all i , $E := \max |e_i| \geq 2$, and $u := v_1 v_2 \dots v_n \geq 3$. We write $v^e = \frac{a'}{b'} = \frac{a}{b}$, where

$$a' := \prod_{e_i > 0} v_i^{e_i}, \quad b' := \prod_{e_i < 0} v_i^{-e_i}, \quad d := \gcd(a', b'), \quad a := a'/d, \quad b := b'/d.$$

We have $a'b' = \prod v_i^{|e_i|}$, hence $\text{rad}(a'b') = \text{rad}(u)$. On the other hand, $a'b' = d^2 ab$, which gives $\text{rad}(a'b') = d \text{rad}(ab)$. Hence $\text{rad}(ab)$ is a divisor of $\text{rad}(u)$, which implies $\omega(ab) \leq \omega(u)$.

Setting $c := a - b$, we have $(-a) + b + c = 0$ and $\gcd(a, b, c) = 1$. Since a, b are positive, we have $\max(a, b, |c|) = \max(a, b)$. We claim that $|c| \leq u^E$, where $E := \max |e_i|$. Indeed, $c \leq a \leq a' \leq u^E$, and similarly, $-c \leq b \leq b' \leq u^E$.

With the above estimates, we obtain

$$N = \text{rad}(abc) = \text{rad}(c) \text{rad}(ab) \leq |c| \text{rad}(u) \leq |c| u \leq u^{E+1}, \quad \omega(ab) \leq \omega(u). \quad (4.2)$$

Conjecture 4.2 implies

$$\max(a, b) \leq \kappa N \left(\frac{\log N}{\omega(ab)} \right)^{\kappa' \omega(ab)} \leq \kappa N (\log N)^{\kappa' \omega(ab)}.$$

Using Equation (4.2), we can bound this as

$$\max(a, b) \leq \kappa |c| u ((E + 1) \log u)^{2\kappa' \omega(u)}.$$

Since $|v^e - 1| = |c/b|$ we get

$$|v^e - 1| \geq |c| / \max(a, b) \geq \kappa^{-1} u^{-1} ((E + 1) \log u)^{-2\kappa' \omega(u)}.$$

Taking logarithms of both sides, we obtain

$$\log |v^e - 1| \geq -\log \kappa - \log u - 2\kappa' \omega(u) (\log(E + 1) + \log \log u).$$

It is known that $\omega(u) \log \log u = O(\log u)$, see [44, §22.10]. Using this, we conclude that indeed $\log |v^e - 1| \geq -\kappa'' \log E \log u$ for a suitable constant $\kappa'' > 0$. ◀

14:24 Complexity of Robust Orbit Problems for Torus Actions and the *abc*-Conjecture

Given $v_1, v_2, \dots, v_n \in \mathbb{Z}_{>0}$ and $e_1, e_2, \dots, e_n \in \mathbb{Z}$, we denote by $\Lambda(v, e)$ the *linear form in logarithms*

$$\Lambda(v, e) := \log v^e = e_1 \log v_1 + e_2 \log v_2 + \dots + e_n \log v_n. \quad (4.3)$$

A bound similar to the one in the previous theorem can also be given in terms of the quantity $\log |\Lambda(v, e)|$, which relates the *abc*-conjecture to Number-Theoretic Hypothesis 1.11.

► **Corollary 4.4.** *Assume Baker's refinement of the abc-conjecture is true. Then there exists a constant $\kappa > 0$ such that for all $v_1, v_2, \dots, v_n \in \mathbb{Z}_{>0}$ and $e_1, e_2, \dots, e_n \in \mathbb{Z}$ with $\Lambda(v, e) \neq 0$, we have*

$$\log |\Lambda(v, e)| \geq -\kappa \log(\max_i |e_i|) \sum_{i=1}^n \log v_i.$$

In particular, Number-Theoretic Hypothesis 1.11 is true if $v_1, \dots, v_n \in \mathbb{Q}_{>0}$.

Proof. First note that the function $h(x) := \log(x)/(x-1)$ is monotonically decreasing and satisfies $\frac{1}{2} \leq h(x) \leq \frac{3}{2}$ on the interval $[\frac{1}{2}, \frac{3}{2}]$. This implies for $|x-1| \leq \frac{1}{2}$,

$$\frac{1}{2} |\log(x)| \leq |x-1| \leq 2 |\log(x)|. \quad (4.4)$$

For showing the stated bound with positive integers v_i , we may assume without loss of generality that $|v^e - 1| \leq \frac{1}{2}$. Then Equation (4.4) implies $\frac{1}{2} |\Lambda(v, e)| \leq |v^e - 1|$ and Theorem 4.3 gives the desired bound.

The assertion for positive rationals $v_i = \frac{p_i}{q_i}$ follows from the observation that $\Lambda(v, e) = \Lambda(\tilde{v}, \tilde{e})$, where \tilde{v} is defined by concatenating p and q , and \tilde{e} by concatenating e and $-e$. ◀

► **Remark 4.5.** Conversely, lower bounds similar to the one in Corollary 4.4, together with their p -adic versions imply the *abc*-conjecture. For a prime p , let $|x|_p$ denote the p -adic absolute value of $x \in \mathbb{Q}$, e.g., $|p^\nu y|_p = p^{-\nu}$ for $y \in \mathbb{Z}$ not divisible by p and $\nu \in \mathbb{Z}$. Suppose the v_i are nonzero integers, $e_i \in \mathbb{Z}$ and write $v_1^{e_1} v_2^{e_2} \dots v_n^{e_n} = a/b$ for coprime integers a, b . If p is a prime dividing $c := a - b$, then $|b|_p = 1$ and $|c/b|_p = |c|_p < 1$. Hence the p -adic logarithm $\log(a/b) = \log(1 + c/b)$ exists and $|\log(1 + c/b)|_p = |c|_p$. In analogy with Equation (4.3), we define

$$|\Lambda(v, e)|_p := |\log(1 + c/b)|_p = |c|_p.$$

Baker considered the product of linear forms in logarithms

$$\Xi := \min(1, |\Lambda(v, e)|) \prod_{p \text{ prime}} \min(1, p |\Lambda(v, e)|_p),$$

where the product is over all primes. He proved that a slightly stronger lower bound for Ξ than the one of Corollary 4.4 is equivalent to his refinement of the *abc*-conjecture Equation (4.1); see [8, Section 5]. Moreover, any lower bound for Ξ implies a version of the *abc*-conjecture: Stewart and Yu [82], refining an earlier work by Stewart and Tijdemann [81], used this approach to prove the inequality

$$\log c < \kappa_\varepsilon N^{\frac{1}{3} + \varepsilon},$$

using the Baker-Wüstholz bound for the linear forms in logarithms [89] and its p -adic versions by van der Poorten [85]. We refer to [9, Chapter 3.7] (see also [38]) for a summary of these results.

In Theorem 4.3 and Corollary 4.4 we assumed that the vectors v_i to be positive rationals. More generally, for our purposes, we would like to allow the v_i to be nonzero Gaussian rationals. In the following, assume that $\log : \mathbb{C}^\times \rightarrow \mathbb{C}$ denotes the principal branch of the logarithm, so $\Im(\log z) \in (-\pi, \pi]$, e.g., $\log(-1) = i\pi$. In analogy, we define the linear form in logarithms

$$\Lambda(v, e) := e_1 \log v_1 + e_2 \log v_2 + \dots + e_n \log v_n. \tag{4.5}$$

How small can $|\Lambda(v, e)|$ be, provided it is nonzero?

In the introduction, we formulated Number-Theoretic Hypothesis 1.11 and noted that it follows from famous conjectures in number theory, such as Waldschmidt’s conjectures [86, Conjecture 14.25], [87, Conjecture 4.14] or the Lang-Waldschmidt conjecture for Gaussian rationals [59, Introduction to Chapters X and XI].

We also note that there are analogues of the abc -conjecture in different number fields [11]. Recall that $\mathbb{Z}[i]$ is a unique factorization domain, so for an element $x \in \mathbb{Z}[i]$ there exist distinct prime elements p_1, p_2, \dots, p_k , exponents $\mu_1, \mu_2, \dots, \mu_k \in \mathbb{Z}_{>0}$, and a unit $u \in \mathbb{Z}[i]^\times$ such that $x = u \prod_{i=1}^k p_i^{\mu_i}$. So we can define the radical of x as $\text{rad}(x) := \prod_{i=1}^m p_i$. We may conjecture the Gaussian integer analogue of Baker’s abc -conjecture⁶: There exist constants $\kappa, \kappa' > 0$ such that for all nonzero $a, b, c \in \mathbb{Z}[i]$ with $a + b + c = 0$ and $\text{gcd}(a, b, c) = 1$, the radical $N := \text{rad}(abc)$ satisfies

$$\max(|a|, |b|, |c|) \leq \kappa |N| \left(\frac{\log |N|}{\omega(ab)} \right)^{\kappa' \omega(ab)}. \tag{4.6}$$

With essentially the same arguments as in the proof of Theorem 4.3 and Corollary 4.4, one shows that Equation (4.6) implies Number-Theoretic Hypothesis 1.11.

Number-Theoretic Hypothesis 1.11 lower bounds the Euclidean distance of $\Lambda(v, e)$ to 0. For our purposes, we need to lower bound the distance $\Delta(\Lambda(v, e), 0) := \text{dist}(\Lambda(v, e), 2\pi i \mathbb{Z})$ in the Δ -metric, see Equation (3.3). This follows easily.

► **Proposition 4.6.** *Assume Number-Theoretic Hypothesis 1.11. Suppose $v_1, \dots, v_n \in \mathbb{Q}(i)^\times$ and $e_1, \dots, e_n \in \mathbb{Z}$. If $\Delta(\Lambda(v, e), 0)$ is nonzero, then $\Delta(\Lambda(v, e), 0) \geq \exp(-\text{poly}(n, B, b))$, where B (resp. b) is a bound for the bit-length of e_i (resp. v_i).*

Proof. We have $\text{dist}(\Lambda(v, e), 2\pi i \mathbb{Z}) = |\Lambda(v, e) - d i \pi|$ for some $d \in \mathbb{Z}$ satisfying $|d| \leq O(n 2^B b)$. Note that $\Lambda(-1, -d) = -d \text{Log}(-1) = -d i \pi$. Therefore $\Lambda(v, e) - d i \pi = \Lambda(\tilde{v}, \tilde{e})$, where \tilde{v} is obtained from v by appending -1 and \tilde{e} is obtained from e by appending $-d$. Now we apply Number-Theoretic Hypothesis 1.11. ◀

4.2 Equivalence of Separation Hypotheses with the Number-Theoretic Hypothesis 1.11

We prove here Theorem 1.12 based on Proposition 3.5, which relates the logarithmic distance between the orbits \mathcal{O}_v and \mathcal{O}_w to $\Delta(H \text{Log } v, H \text{Log } w)$. The essential observation is that $\Delta(H \text{Log } v, H \text{Log } w)$ is determined by linear forms in logarithms.

Proof of Theorem 1.12. We will only prove the first equivalence; the second is shown in the same way, but using Corollary 3.8 instead of Proposition 3.5.

⁶ We have not seen this conjecture in the literature.

(\Rightarrow) We first assume Number-Theoretic Hypothesis 1.11 and deduce Separation Hypothesis 1.7. Suppose $v, w \in (\mathbb{Q}(i)^\times)^n$ have bit-lengths bounded by b and that the bit-length of the weight matrix M is bounded by B . By Proposition 2.2, there is a matrix H of rational invariants of bit-length $\text{poly}(d, n, B)$. We denote by $\alpha_1^T, \dots, \alpha_k^T$ the rows of $H \in \mathbb{Z}^{k \times n}$.

Assume that $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \neq 0$. By Proposition 3.5, $\Delta(H \text{Log } v, H \text{Log } w) = \Delta(H(\text{Log } v - \text{Log } w), 0)$ is nonzero. Hence at least one of the one-dimensional distances $\Delta(\alpha_j^T(\text{Log } v - \text{Log } w), 0)$ must be nonzero. By Proposition 4.6 it is lower bounded by $\exp(-\text{poly}(d, n, B, b))$. Proposition 3.5 also implies

$$\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \frac{1}{\sigma_{\max}(H)} \Delta(H \text{Log } v, H \text{Log } w) \geq \frac{1}{\sigma_{\max}(H)} \Delta(\alpha_j^T(\text{Log } v - \text{Log } w), 0). \quad (4.7)$$

Moreover, by Lemma 2.7, $\sigma_{\max}^{-1}(H)$ is lower bounded by $\exp(-\text{poly}(d, n, B))$. This finishes the proof of the first implication.

(\Leftarrow) Now we assume Separation Hypothesis 1.7 and deduce Number-Theoretic Hypothesis 1.11. Suppose $v_i \in (\mathbb{Q}(i)^\times)^n$ have bit-length bounded by b , and $e_i \in \mathbb{Z}$ have bit-lengths bounded by B such that $\Lambda(v, e) \neq 0$. We assume without loss of generality that $\gcd(e_1, e_2, \dots, e_n) = 1$. Put $e := (e_1, e_2, \dots, e_n) \in \mathbb{Z}^n$ and consider the rational hyperplane e^\perp of \mathbb{R}^n . Note $\Lambda(v, e) \neq 0$ expresses that $\text{Log } v \notin e^\perp$. There is a lattice basis $\alpha_1, \alpha_2, \dots, \alpha_{n-1} \in \mathbb{Z}^n$ of e^\perp having bit-length $\text{poly}(n, B)$; see Theorem 2.1 (a). Let $M \in \mathbb{Z}^{d \times n}$ denote the matrix with rows α_i , where $d := n - 1$.

We let $(\mathbb{C}^\times)^d$ act on \mathbb{C}^n via the weight matrix M . The vector e generates the lattice of invariant Laurent monomials (recall $\gcd(e_1, e_2, \dots, e_n) = 1$). In this case the matrix $H \in \mathbb{Z}^{1 \times n}$ of rational invariants is just $H = e^T$.

Consider $v := (v_1, v_2, \dots, v_n)$ and $w := (1, \dots, 1) \in \mathbb{C}^n$, so $\text{Log } w = 0$. Since H has only one row, we have $\sigma_{\max}(H) = \sigma_{\min}(H) = \|e\|$, which implies that the inequality from Proposition 3.5 is actually an equality:

$$\Delta(e^T \text{Log } v, 0) = \|e\| \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w). \quad (4.8)$$

If $\Delta(e^T \text{Log } v, 0) \neq 0$, then by Separation Hypothesis 1.7, $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \exp(-\text{poly}(n, b, B))$. Hence $|\Lambda(v, e)| \geq \Delta(e^T \text{Log } v, 0) \geq \exp(-\text{poly}(n, b, B))$.

When $\Delta(e^T \text{Log } v, 0) = 0$, we have $0 \neq \Lambda(v, e) = e^T \text{Log } v \in 2\pi\mathbb{Z}$ so $|\Lambda(v, e)| \geq 2\pi$. \blacktriangleleft

► Remark 4.7. Matveev's lower bound (1.8) combined with inequality Equation (4.7) shows that unconditionally

$$\text{sep}_T(d, n, B, b) \geq \exp(-B^{O(1)} b^{O(n)}).$$

The same lower bound holds for $\text{sep}_K(d, n, B, b)$.

5 Approximations of orbit distances

In this section, we provide polynomial time approximation algorithms for the orbit distance approximation problems. We solve Problem 1.2 in polynomial time and prove Theorem 1.8. The main theorem of this section is Theorem 5.2.

5.1 Approximate orbit distance problems

So far, we discussed four *group actions*: the actions of $T = (\mathbb{C}^\times)^d$ or $K = (S^1)^d$ on the spaces $V = \mathbb{C}^n$ and $\mathbb{C}^n / 2\pi i \mathbb{Z}^n$ given by a weight matrix $M \in \mathbb{Z}^{d \times n}$. We also defined a *metric* δ on the resulting spaces of orbits. Let us list all these cases:

$$(T, \delta_{\log}) \quad \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) := \Delta(\text{Log}(\mathcal{O}_v), \text{Log}(\mathcal{O}_w))$$

$$(K, \text{dist}) \quad \text{dist}(\mathcal{C}_v, \mathcal{C}_w) := \inf_{k \in K} \|k \cdot v - w\|$$

$$(T, \Delta) \quad \Delta(T * \eta, T * \zeta) := \inf_{t \in T} \Delta(t * \eta, \zeta)$$

$$(K, \Delta) \quad \Delta(K * \eta, K * \zeta) := \inf_{k \in K} \Delta(k * \eta, \zeta),$$

see Definition 3.1 for the $*$ -action and Equation (3.3) for the Δ -metric. We define an orbit distance approximation problem in each setting.

► **Definition 5.1.** *In each of above case, abbreviated (G, δ) , the orbit distance approximation problem $\text{ROP}(G, \delta)_\gamma$ with approximation factor $\gamma \geq 1$ is the following: on input a weight matrix $M \in \mathbb{Z}^{d \times n}$ and $v, w \in (\mathbb{Q}(i)^\times)^n$, resp. $\eta, \zeta \in \mathbb{Q}^n + 2\pi i \mathbb{Q}^n$, compute a number $D \in \mathbb{Q}_{\geq 0}$ such that*

$$\delta(G \cdot v, G \cdot w) \leq D \leq \gamma \delta(G \cdot v, G \cdot w).$$

We think of γ as a function of the input bit-length, determined by the usual parameters d, n, B, b .

We can now precisely state the main algorithmic result of this section. Recall from Equation (3.10) the regions $D_{r,R}$ defined for $0 < r \leq R$. Note that Theorem 1.8 follows from the second and third part of the next result.

► **Theorem 5.2.** *There are approximation factors $\gamma_i = \gamma_i(d, n, B)$ bounded $\exp(\text{poly}(d, n, B))$ such that:*

- (a) $\text{ROP}(G, \delta)_{\gamma_1}$ admits a polynomial time algorithm in the cases (T, Δ) and (K, Δ) .
- (b) $\text{ROP}(T, \delta_{\log})_{\gamma_2}$ admits a polynomial time algorithm, if Separation Hypothesis 1.7 holds.
- (c) $\text{ROP}(K, \text{dist})_\gamma$, when restricted to inputs in $D_{r,R}$, admits a polynomial time algorithm with approximation factor $\gamma = \frac{R}{r} \gamma_3(d, n, B)$, if Separation Hypothesis 1.6 holds.

One may wonder why the first part of Theorem 5.2 holds unconditionally. This is explained by the next result, which gives an unconditional separation of orbits with respect to the Δ -metric.

► **Proposition 5.3.** *If $\eta, \zeta \in \mathbb{Q}^n + 2\pi i \mathbb{Q}^n$ with $T * \eta \neq T * \zeta$, then*

$$\Delta(T * \eta, T * \zeta) \geq \exp(-\text{poly}(d, n, b, B)).$$

The analogous statement holds for the action of K .

Proof. We write $\eta = \rho + 2\pi i \theta$, $\zeta = \tau + 2\pi i \phi$ and recall from Equation (3.8) that

$$\Delta^2(T * \eta, T * \zeta) = \text{dist}^2(\rho - \tau, U) + 4\pi^2 \Delta^2(K * i \theta, K * i \phi).$$

The first contribution $\text{dist}^2(\rho - \tau, U)$, if nonzero, is easily lower bounded using the Gram-Schmidt orthogonalization. So we may assume $K * i \theta \neq K * i \phi$. In this case, Proposition 3.5 implies $\text{dist}(H(\theta - \phi), \mathbb{Z}^k) \neq 0$ and

$$\Delta(K * i \theta, K * i \phi) \geq \frac{1}{\sigma_{\max}(H)} \text{dist}(H(\theta - \phi), \mathbb{Z}^k),$$

where H is a matrix of rational invariants whose bit-length is bounded by Proposition 2.2. We can upper bound $\sigma_{\max}(H)$ by Proposition 2.2 and Lemma 2.7, which provides the desired lower bound for the T -action. The distance of $H(\theta - \phi)$ to \mathbb{Z}^k is also at most exponentially small since it is lower bounded by the reciprocal of the largest denominator of the entries of the rational vector $H(\theta - \phi)$. The proof for the K -action is analogous. ◀

We also show that our algorithms for approximating the distance between orbits can be modified to produce a group element witnessing the γ -proximity of orbits. To avoid repetition, we only state this for the case (T, δ_{\log}) , even though this can also be achieved in the other settings in a similar way.

► **Theorem 5.4.** *Assume Separation Hypothesis 1.7. On input M, v, w , one can compute in polynomial time a vector $x \in \mathbb{C}^d$ that satisfies*

$$\delta_{\log}(e^x \cdot v, w) \leq \gamma \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \quad (5.1)$$

if $\mathcal{O}_v \neq \mathcal{O}_w$. If $\mathcal{O}_v = \mathcal{O}_w$, then the algorithm correctly identifies this case instead of returning x . The approximation parameter is exponentially bounded in the bit-length of the input.

In the next section, we introduce the distance computation problem SLDP and exhibit a polynomial time algorithm for it. Theorems 5.2 and 5.4 are then proved in Section 5.3 by reducing the orbit distance approximation problems to SLDP.

5.2 The subspace-to-cubic-lattice distance problem

The orbit distance approximation problems are closely related to the following problem.

► **Definition 5.5.** *The subspace-to-cubic-lattice distance approximation problem $SLDP_\gamma$ with approximation factor γ is the task of computing on input*

- a target vector $t \in \mathbb{Q}^n$,
- a subspace $U \subset \mathbb{R}^n$, spanned by given linearly independent input vectors $u_1, \dots, u_{n-k} \in \mathbb{Z}^n$,
- a number $D \in \mathbb{Q}_{\geq 0}$ such that

$$\text{dist}(t + U, \mathbb{Z}^n) \leq D \leq \gamma \text{dist}(t + U, \mathbb{Z}^n).$$

We think of γ a function of the input bit-length.

► **Remark 5.6.** The problem $SLDP_\gamma$ is easy if $U = 0$. Indeed, $\text{dist}^2(t, \mathbb{Z}^n)$ can be exactly and efficiently computed by rounding the coordinates of t to integers. At the other extreme, the problem is trivial if $U = \mathbb{R}^n$. However, we will show in Section 6, that $SLDP_\gamma$ is NP-hard for constant γ (or more generally, an almost polynomial γ), by providing a reduction from the *closest vector problem* to it.

► **Proposition 5.7.** *There is an approximation factor γ bounded exponentially in the input bit-length such that $SLDP_\gamma$ admits a polynomial time algorithm. This algorithm can be modified to compute on input (t, U) a witness of proximity $(u, \alpha) \in U \times \mathbb{Z}^n$ such that*

$$\text{dist}(t + u, \alpha) \leq \gamma \text{dist}(t + U, \mathbb{Z}^n).$$

Proof. For given $M \in \mathbb{Z}^{(n-k) \times n}$ we compute $H \in \mathbb{Z}^{k \times n}$ such that $U := \text{im } M^T = \ker H$ in polynomial time, see Proposition 2.2. When applying Lemma 2.5 to $A = \{t\}$ and $B = \mathbb{Z}^n$, we get

$$\text{dist}(t + U, \mathbb{Z}^n) \leq D_1 := \frac{\text{dist}(Ht, \mathbb{Z}^k)}{\sigma_{\min}(H)} \leq \frac{\sigma_{\max}(H)}{\sigma_{\min}(H)} \text{dist}(t + U, \mathbb{Z}^n). \quad (5.2)$$

Hence, D_1 approximates $\text{dist}(t + U, \mathbb{Z}^n)$ within a factor of $\sigma_{\max}(H)/\sigma_{\min}(H)$. By Lemma 2.7 we can compute a rational approximation D of D_1 within a factor of 2 in polynomial time. Moreover $\sigma_{\max}(H)/\sigma_{\min}(H)$ is exponentially upper bounded in the input bit-length.

We show now how to compute the witness. By rounding the entries of Ht to nearest integers, we can compute an integral vector $\beta \in \mathbb{Z}^k$ such that $\text{dist}(Ht, \mathbb{Z}^k) = \text{dist}(Ht, \beta)$. Recall that $H(\mathbb{Z}^n) = \mathbb{Z}^k$. We can further efficiently compute $\alpha \in \mathbb{Z}^n$ such that $H\alpha = \beta$. By projecting $t - \alpha$ orthogonally onto U , we can compute in polynomial time $u \in U$ such that $\text{dist}(t + u, \alpha) = \text{dist}(t + U, \alpha)$. ◀

► **Remark 5.8.** In Section 6.1, we analyse an alternative algorithm (based on the LLL-algorithm), which solves SLDP_γ for $\gamma = 2^{O(n)}$ in polynomial time (Corollary 6.4). In comparison, the algorithm given in Proposition 5.7 provides an approximation factor of $\gamma = \kappa(H) := \sigma_{\max}(H)/\sigma_{\min}(H)$ which is bounded by $2^{n(B + \log_2 n)}$ in the worst case.

5.3 Proof of Theorem 5.2

The first part of this theorem follows from the following reduction, jointly with Proposition 5.7.

► **Lemma 5.9.** *Both $\text{ROP}(T, \Delta)_{2\gamma}$ and $\text{ROP}(K, \Delta)_{2\gamma}$ reduce to SLDP_γ in polynomial time, for any $\gamma \geq 1$. Hereby the ambient dimension does not change.*

Proof. We will only prove the reduction for $G = K$ since the $G = T$ case is analogous. Suppose that M, η, ζ are given as input, write $\eta = \rho + 2\pi i \theta, \zeta = \tau + 2\pi i \phi$, set $t := \theta - \phi$, and $U := \text{im } M^T$. We can compute an integral basis of U in polynomial time. Equation (3.4) and Equation (3.9) imply

$$\Delta^2(K * \eta, K * \zeta) = \|\rho - \tau\|^2 + 4\pi^2 \text{dist}^2(t + U, \mathbb{Z}^n).$$

If $\text{dist}(t + U, \mathbb{Z}^n) \leq D \leq \gamma \text{dist}(t + U, \mathbb{Z}^n)$, then

$$\Delta^2(K * \eta, K * \zeta) \leq \|\rho - \tau\|^2 + 4\pi^2 D^2 \leq \gamma^2 \|\rho - \tau\|^2 + 4\pi^2 D^2 \leq \gamma^2 \Delta^2(K * \eta, K * \zeta),$$

hence $D' := \sqrt{\|\rho - \tau\|^2 + 4\pi^2 D^2}$ is a γ -approximate solution of $\text{ROP}(K, \Delta)$. We compute a rational number D'' such that $D' \leq D'' \leq 2D'$. Then $\Delta(K * \eta, K * \zeta) \leq D'' \leq 2\gamma \Delta(K * \eta, K * \zeta)$, ◀

A reduction in the reverse direction will be needed later for the hardness proof, see Lemma 5.12. The next lemma studies that happens with ROP , when the metric δ_{\log} , resp. dist , are replaced by Δ . This lemma proves the second and third part of Theorem 5.2 by reducing it to its first part.

► **Lemma 5.10.** *Let $\gamma \geq 1$ be any function of the inputs, let $0 < r \leq R$ and put $\gamma_1 := \frac{2R}{r} \gamma$.*

- (a) *There is a polynomial time reduction from $\text{ROP}(T, \delta_{\log})_{2\gamma}$ to $\text{ROP}(T, \Delta)_{\gamma}$, provided Separation Hypothesis 1.7 holds.*
- (b) *There is a polynomial time reduction from $\text{ROP}(K, \text{dist})_{\gamma_1}$, when restricted to inputs in $D_{r,R}$, to $\text{ROP}(K, \Delta)_{\gamma}$, provided Separation Hypothesis 1.6 holds.*

Proof. We only give the proof for part two since the proof of part one is analogous. The only difference between the two cases is that in the second we need to use Proposition 3.7 which gives an equivalence between the metrics dist and δ_{\log} within an $O(R/r)$ -factor, and in the first we do not need it.

14:30 Complexity of Robust Orbit Problems for Torus Actions and the abc -Conjecture

Given $v, w \in D_{r,R}$, we test in polynomial time whether $\mathcal{C}_v = \mathcal{C}_w$ (see Section 2.2). If $\mathcal{C}_v = \mathcal{C}_w$ we return $D = 0$.⁷ Otherwise, the reduction just consists of replacing the exact values $\eta := \text{Log } v$ and $\zeta := \text{Log } w$ by approximations

$$\|\tilde{\eta} - \eta\| < \kappa/2, \quad \|\tilde{\zeta} - \zeta\| < \kappa/2, \quad (5.3)$$

where the accuracy is prescribed by the separation parameter. For $\kappa := \frac{1}{9R} \text{sep}_K(d, n, B, b)$, this can be achieved in polynomial time assuming Separation Hypothesis 1.6. For convenience, we denote the orbit distances of the exact vectors and the approximations computed by

$$\Delta := \Delta(K * \eta, K * \zeta), \quad \tilde{\Delta} := \Delta(K * \tilde{\eta}, K * \tilde{\zeta}).$$

It suffices to prove that $\tilde{\Delta} \leq D \leq \gamma \tilde{\Delta}$ implies

$$\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq \frac{9RD}{8} \leq \gamma_1 \text{dist}(\mathcal{C}_v, \mathcal{C}_w),$$

where $\gamma_1 = 2R\gamma/r$. We now prove this implication.

By Proposition 3.7, we have

$$\frac{2r}{\pi} \Delta \leq \text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq R \Delta, \quad (5.4)$$

which implies $\Delta \geq 9\kappa$ by our choice of κ . With Equation (5.3) and the triangle inequality, we get

$$|\tilde{\Delta} - \Delta| < \kappa, \quad \frac{\kappa}{\tilde{\Delta}} \leq \frac{1}{9}.$$

By dividing through Δ , this implies $\frac{8}{9} < \frac{\tilde{\Delta}}{\Delta} < \frac{10}{9}$. From the assumption $\tilde{\Delta} \leq D \leq \gamma \tilde{\Delta}$, we infer

$$\Delta \leq \frac{9}{8} D \leq \frac{10}{8} \gamma \Delta.$$

Equation (5.4) implies

$$\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq \frac{9RD}{8} \leq \frac{10\pi R\gamma}{16r} \text{dist}(\mathcal{C}_v, \mathcal{C}_w) < \frac{2R\gamma}{r} \text{dist}(\mathcal{C}_v, \mathcal{C}_w),$$

which completes the proof. \blacktriangleleft

We can prove Theorem 5.4 in a similar way.

Proof of Theorem 5.4. On input $v, w \in (\mathbb{C}^\times)^n$, we first test in polynomial time whether $\mathcal{O}_v = \mathcal{O}_w$ (see Section 2.2). If this is the case, we return $D = 0$. Otherwise, as in Equation (5.3), we compute approximations $\tilde{\eta}$ and $\tilde{\zeta}$ of the exact values $\eta := \text{Log } v$ and $\zeta := \text{Log } w$ with accuracy $\kappa = \text{sep}_T(d, n, B, b)/2$: Separation Hypothesis 1.7 guarantees that this can be done in polynomial time. We have for all $x \in \mathbb{C}^d$

$$\Delta(e^x * \eta, \zeta) \geq \Delta(T * \eta, T * \zeta) = \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \text{sep}_T(d, n, B, b) = 2\kappa.$$

Hence, using the triangle inequality, we get for all $x \in \mathbb{C}^d$

$$\left| \Delta(e^x * \tilde{\eta}, \tilde{\zeta}) - \Delta(e^x * \eta, \zeta) \right| < \kappa \leq \frac{1}{2} \Delta(e^x * \eta, \zeta),$$

⁷ To get a Karp reduction, we can simply return the ROP(K, Δ) instance $(\eta, \zeta) = (0, 0)$.

therefore

$$\frac{1}{2} \Delta(e^x * \eta, \zeta) < \Delta(e^x * \tilde{\eta}, \tilde{\zeta}) < \frac{3}{2} \Delta(e^x * \eta, \zeta). \quad (5.5)$$

Write $\tilde{\eta} = \tilde{\rho} + 2\pi i \tilde{\theta}$ and $\tilde{\zeta} = \tilde{\tau} + 2\pi i \tilde{\phi}$ and recall from Equation (3.8) that

$$\Delta^2(T * \tilde{\eta}, T * \tilde{\zeta}) = \text{dist}^2(\tilde{\rho} - \tilde{\tau}, U) + 4\pi^2 \text{dist}^2(\tilde{\theta} - \tilde{\phi} + U, \mathbb{Z}^n).$$

For the first summand, we compute the orthogonal projection u_1 of $\tilde{\rho} - \tilde{\tau}$ onto U . For the second summand, we use Proposition 5.7 to compute a witness (u_2, α) such that

$$\text{dist}(\tilde{\theta} - \tilde{\phi} + u_2, \alpha) \leq \gamma \text{dist}(\tilde{\theta} - \tilde{\phi} + U, \mathbb{Z}^n) \quad (5.6)$$

for some γ satisfying $\gamma = \exp(\text{poly}(B, n))$. We then compute $y, z \in \mathbb{R}^d$ such that $M^T y = u_1$ and $M^T z = u_2$ and compute the group element $x := y + 2\pi i z$. Then we have by Equation (3.3)

$$\begin{aligned} \Delta^2(e^x * \tilde{\eta}, \tilde{\zeta}) &= \text{dist}^2(\tilde{\rho} - \tilde{\tau}, M^T y) + 4\pi^2 \text{dist}^2(\tilde{\theta} - \tilde{\phi} + u_2, \alpha) \\ &\leq \text{dist}^2(\tilde{\rho} - \tilde{\tau}, U) + 4\pi^2 \gamma^2 \text{dist}^2(\tilde{\theta} - \tilde{\phi} + U, \mathbb{Z}^n) \\ &\leq \gamma^2 \Delta^2(T * \tilde{\eta}, T * \tilde{\zeta}), \end{aligned}$$

where we used Equation (5.6) for the second inequality. Combining with Equation (5.5), we obtain

$$\Delta(e^x * \eta, \zeta) \leq 2\Delta(e^x * \tilde{\eta}, \tilde{\zeta}) \leq 2\gamma \Delta(T * \eta, T * \zeta),$$

which finishes the proof. ◀

► **Remark 5.11.** The algorithm underlying the reduction in the proof of Lemma 5.10 runs in time polynomial in the input bit-length and $\log \text{sep}_T^{-1}(d, n, B, b)$. Thus it is a polynomial time algorithm if and only if the separation hypothesis is true.

5.4 Reductions from SLDP to ROP

The following reductions are needed in the next section. While the proofs are straightforward perturbation arguments, dealing with the relative errors requires some care, so that we write down the detailed arguments at least in one case.

► **Lemma 5.12.** *Let (G, δ) denote any of the four cases in Definition 5.1. Then there is a polynomial-time reduction from $\text{SLDP}_{2\gamma}$ to $\text{ROP}(G, \delta)_{\gamma}$, for any $\gamma \geq 1$. Hereby the ambient dimension is preserved.*

Proof. We only provide the argument for $(G, \delta) = (K, \text{dist})$, the case (T, δ_{\log}) being similar and the remaining two cases being trivial. Consider an instance U, t of SLDP, where $U \subset \mathbb{R}^n$ is a subspace of dimension $d := n - k$ given as the row span of $M \in \mathbb{Z}^{(n-k) \times n}$, and $t \in \mathbb{Q}^n$. The matrix M defines actions of $T = (\mathbb{C}^\times)^d$ and $K = (S^1)^d$ on \mathbb{C}^n . The essential connection is Equation (3.4), which implies that

$$\text{dist}(t + U, \mathbb{Z}^n) = \Delta(K * it, K * 0) = \Delta(T * it, T * 0). \quad (5.7)$$

By Proposition 5.3, there is a separation function $0 < \epsilon \leq 1$ such that $\log \epsilon^{-1}$ polynomially bounded in the parameters d, n, B, b and $\Delta(K * 2\pi it, K * 0) \geq \epsilon$ when the orbits are different.

14:32 Complexity of Robust Orbit Problems for Torus Actions and the abc -Conjecture

By Lemma 2.9 we compute $v \in (\mathbb{Q}(i)^\times)^n$ such that $\|v - \text{Exp}(2\pi i t)\| < \kappa := \epsilon/18$. Hence $v \in D_{r,R}$ for $r = 17/18$ and $R = 19/18$. Lemma 5.13 below expresses a Lipschitz property of Log and gives

$$\|\text{Log } v - 2\pi i t\|_\infty < \frac{\kappa}{1 - \kappa} < \frac{18}{17} \kappa.$$

Thus the distances $\Delta_1 := \Delta(K * \text{Log } v, 0)$ and $\Delta_2 := \Delta(K * 2\pi i t, 0)$ of the corresponding K -orbits to the zero orbit are close: $|\Delta_1 - \Delta_2| < \frac{18}{17} \kappa$. Using $\Delta_2 \geq \epsilon = 18\kappa$, this bounds the relative errors as $\frac{|\Delta_1 - \Delta_2|}{\Delta_2} \leq \frac{1}{17}$, and hence

$$\Delta_2 \leq \frac{17}{16} \Delta_1 \leq \frac{18}{17} \Delta_2.$$

Note that $\text{dist}(t + U, \mathbb{Z}^n) = \frac{1}{2\pi} \Delta_2$ by Equation (5.7). Consider the vector $w := (1, \dots, 1)$. Proposition 3.7 implies, using $v \in D_{r,R}$ with $r = 17/18$ and $R = 19/18$, that

$$\frac{2}{\pi} \frac{17}{18} \Delta_1 \leq \text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq \frac{19}{18} \Delta_1. \quad (5.8)$$

Now assume that $\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq D \leq \gamma \text{dist}(\mathcal{C}_v, \mathcal{C}_w)$. Then,

$$\Delta_2 \leq \frac{17}{16} \Delta_1 \leq \frac{17}{16} \frac{\pi}{2} \frac{18}{17} D \leq \frac{17}{16} \frac{\pi}{2} \frac{18}{17} \gamma \frac{19}{18} \frac{18}{17} \Delta_2 \leq 2 \frac{\Delta_2}{2\pi}.$$

This means that

$$\text{dist}(t + U, \mathbb{Z}^n) \leq \frac{9}{32} D \leq 2\gamma \text{dist}(t + U, \mathbb{Z}^n),$$

which completes the proof of the reduction. \blacktriangleleft

► **Lemma 5.13.** *For any $t \in \mathbb{R}^n$ and $z \in \mathbb{C}^n$, $\|z - \text{Exp}(2\pi i t)\| < \kappa < 1$ implies that $\|\text{Log}(z) - 2\pi i t\| < \kappa/(1 - \kappa)$.*

Proof. It is sufficient to verify the claim for $n = 1$, in which case the claim follows using $|\text{Log}' z| = \left| \frac{1}{z} \right| \leq \frac{1}{1 - \kappa}$. \blacktriangleleft

6 Hardness of robust orbit problems

The main goal of this section is to prove Theorem 1.4. Actually, we prove the following, slightly more general result, that covers all four settings introduced in Section 5.1.

► **Theorem 6.1.** *There is $c > 0$ such that the robust orbit distance approximation problem $\text{ROP}(G, \delta)_\gamma$ is NP-hard for the approximation factor $\gamma(n) = n^{c/\log \log n}$, for each of the four combinations of group actions and metrics considered in Definition 5.1.*

Of course, this implies the hardness of the decisional version of the considered orbit distance approximation problems, namely:

► **Corollary 6.2.** *There is no polynomial time algorithm that on input M, v, w and $\epsilon > 0$ decides*

$$\text{dist}(\mathcal{C}_v, \mathcal{C}_w) \leq \epsilon,$$

unless $P = NP$. The analogous result also holds for the logarithmic distance $\delta_{\log}(\mathcal{O}_v, \mathcal{O}_w)$ of the orbits of the T -action.

Proof. By binary search, making oracle calls to a hypothetical polynomial time algorithm for the above decision problem, we can compute an approximation to $\text{dist}(\mathcal{C}_v, \mathcal{C}_w)$ within any constant factor, say two. This provides a polynomial time algorithm for the robust orbit distance approximation problem with approximation factor $\gamma = 2$. By Theorem 6.1, this implies $P = NP$. ◀

The proof of Theorem 6.1 relies on the NP-hardness of the closest vector problem CVP_γ of algorithmic lattice theory due to Dinur et al. [29]. We discuss the closest vector problem in Section 6.1. The proof of Theorem 6.1 then proceeds by exhibiting a polynomial time reduction from $\text{CVP}_{2\gamma}$ to $\text{ROP}(G, \delta)_\gamma$. This goes in two steps: from Lemma 5.12 we already know that $\text{SLDP}_{2\gamma}$ can be reduced to $\text{ROP}(G, \delta)_\gamma$ in polynomial time. The main ingredient of the proof is a polynomial time reduction of CVP_γ to SLDP_γ , which is presented in Section 6.2. This relies on Theorem 1.5 on lattice lifting, whose proof is given in Section 6.3.

6.1 The closest vector problem

This problem attracted a lot of research due to lattice based cryptography, such as GGH, NTRU and homomorphic encryption [37, 47, 48, 35]. These cryptosystems are conjectured to be secure against quantum computers and rely on a (conjectured) hardness of CVP.

Throughout the section, m will always denote the dimension of a CVP instance and n will always denote the dimension of a SLDP instance.

► **Definition 6.3.** The closest vector problem *with approximation factor* $\gamma \geq 1$, denoted CVP_γ , is the task of computing on input

- a target vector $t \in \mathbb{Q}^m$,
- a lattice \mathcal{L} spanned by the columns of generator matrix $G \in \mathbb{Z}^{m \times m}$ with $\det G \neq 0$,
- a number $D \in \mathbb{Q}_{\geq 0}$ such that $\text{dist}(t, \mathcal{L}) \leq D \leq \gamma \text{dist}(t, \mathcal{L})$.

The first observation is that SLDP_γ can be easily reduced to CVP_γ . Indeed, if $P : \mathbb{R}^n \rightarrow U^\perp$ denotes the orthogonal projection along U , then $\text{dist}(t + U, \mathbb{Z}^n) = \text{dist}(P(t), \mathcal{L})$ by Corollary 2.6. Here, $\mathcal{L} := P(\mathbb{Z}^n)$ is a lattice by Lemma 2.13, and we can compute a lattice basis of \mathcal{L} in polynomial time. This reduces SLDP_γ to CVP_γ . We note that this reduction preserves γ . Moreover, the ambient dimension n of the SLDP instance upper bounds the lattice dimension m of the constructed CVP instance.

The key contribution in the proof of Theorem 6.1 is a polynomial time reduction in the reverse direction, see Theorem 6.7.

CVP_γ is known to be NP-hard for a nearly polynomial approximation ratio, $\gamma = m^{c/\log \log m}$ for some $c > 0$, see [29]. On the other hand, it is well known that CVP_γ can be solved in polynomial time with approximation factor $\gamma(m) = 2^{O(m)}$ by the well-known LLL-basis reduction algorithm of [61]. This implies the following.

► **Corollary 6.4.** SLDP_γ admits a polynomial time algorithm for the approximation factor $\gamma(n) = 2^{O(n)}$.

If we do not insist on polynomial time algorithms, then the lattice element $\alpha \in \mathcal{L}$ closest to the target vector t can be computed exactly. Kannan [53] showed that there is an algorithm for doing so, that runs in time $2^{O(m \log m)}$ times the input size. Combining this with the above reduction, we obtain:

► **Corollary 6.5.** SLDP_1 can be solved by a polynomial time algorithm when n is fixed, if we allow exact computation of square roots.

We use this to show polynomial time feasibility of Problem 1.2 and Problem 1.3 when n is fixed.

Proof of Theorem 1.9 and Theorem 1.10. Matveev's bound (1.8) implies Number-Theoretic Hypothesis 1.11 if n is fixed. Then also Separation Hypothesis 1.7 holds if n is fixed, as shown by the proof of Theorem 1.12. Note that the reductions in Lemma 5.10 preserve the ambient dimension. The assertion follows now with Corollary 6.5. ◀

► **Remark 6.6.** The interesting regime of CVP_γ is when γ is polynomial in m . Lattice based cryptosystems rely on the hardness of CVP_γ in this regime, but as of now, proving this is a major open problem in the area. In fact it is known that CVP_γ is in $\text{NP} \cap \text{coNP}$ [2], for $\gamma = 100\sqrt{m}$, which implies that CVP_γ cannot be NP-hard in this regime, unless $\text{NP} \subset \text{coNP}$ and the polynomial hierarchy collapses.

6.2 The reduction from CVP to SLDP via lattice liftings

We now establish the key reduction from CVP to SLDP by relying on the lattice lifting result Theorem 1.5.

► **Theorem 6.7.** *There is a polynomial time (Turing) reduction from CVP_γ to SLDP_γ (with the same γ). In more detail, given as input a CVP instance (t, \mathcal{L}) with ambient dimension m , the reduction either solves CVP exactly or it computes a scaling factor $s \in \mathbb{Z}_{>0}$ and an SLDP instance (\tilde{t}, U) of ambient dimension n where $\text{dist}(t, \mathcal{L}) = s \text{dist}(\tilde{t} + U, \mathbb{Z}^n)$ and $m \leq n \leq O(m^2 \log m)$.*

Proof of Theorem 6.7. An instance of CVP_γ consists of a rank m lattice \mathcal{L} in \mathbb{R}^m , given by a generator matrix, and a target vector $t \in \mathbb{Q}^m$. We apply Theorem 1.5. Thus, for a given such instance, we can compute in polynomial time an orthonormal basis $v_1, v_2, \dots, v_n \in \mathbb{Q}^n$, where $n \geq m$, and a scaling factor $s \in \mathbb{Z}_{>0}$, such that the lattice \mathcal{L}' generated by v_1, v_2, \dots, v_n satisfies $\mathcal{L} = sP(\mathcal{L}')$, where $P: \mathbb{R}^n \rightarrow \mathbb{R}^m$ denotes the orthogonal projection onto the first m coordinates. We view \mathbb{R}^m as the subspace of \mathbb{R}^n whose last $n - m$ coordinates are zero.

Consider the orthogonal matrix $Q \in \mathbb{Q}^{n \times n}$ with columns v_1, v_2, \dots, v_n , thus $Qe_i = v_i$ if the e_i denote the standard basis vectors. This implies that $Q(\mathbb{Z}^n) = \mathcal{L}'$. Let w_i be the rows of Q , thus $w_i = Q^T e_i$ and $Qw_i = e_i$. We define the subspace U as the span of w_{m+1}, \dots, w_n , thus $Q(U) = \langle e_{m+1}, \dots, e_n \rangle = \ker P$. Moreover, we define $t' := s^{-1}Q^T t$. Then $Qt' = s^{-1}t$. Summarizing,

$$sP(\mathcal{L}') = \mathcal{L}, \quad Q(\mathbb{Z}^n) = \mathcal{L}', \quad Q(U) = \ker P, \quad Qt' = s^{-1}t.$$

Let us verify that

$$\text{dist}(t, \mathcal{L}) = s \text{dist}(t' + U, \mathbb{Z}^n). \tag{6.1}$$

Indeed, $\text{dist}(t' + U, \mathbb{Z}^n) = \text{dist}(Q(t' + U), Q(\mathbb{Z}^n))$ by the orthogonality of Q . Moreover,

$$\text{dist}(Q(t') + Q(U), Q(\mathbb{Z}^n)) = \text{dist}(s^{-1}t + \ker P, \mathcal{L}') = \text{dist}(s^{-1}P(t), P(\mathcal{L}')) = s^{-1} \text{dist}(t, \mathcal{L})$$

where we used Corollary 2.6 in the second and $P(t) = t$ in the last equality.

Clearly, this defines a polynomial time reduction of CVP_γ to SLDP_γ that does not change the approximation factor γ .

Suppose the given instance of CVP has bit-length l . Theorem 1.5 implies that we can assume $n = O(m(\log m + \log l))$. We can bound l in terms of m by making a case distinction. If l is so large that $l \geq 2^{\Omega(m \log m)}$, then we apply the algorithm in [53], which

solves CVP exactly, in time $l2^{O(m \log m)}$ which is polynomially bounded in l in this case. Otherwise, if $l \leq 2^{O(m \log m)}$, we perform the above described reduction to SLDP. Then we have $m(\log m + \log l) \leq O(m^2 \log m)$ and hence we can upper bound $n = O(m^2 \log m)$. This completes the proof. ◀

Proof of Theorem 6.1. We compose the polynomial time reduction $\text{CVP}_{2\gamma}$ to $\text{SLDP}_{2\gamma}$ of Theorem 6.7 with the polynomial time reductions $\text{SLDP}_{2\gamma}$ to $\text{ROP}(G, \delta)_\gamma$ in Lemma 5.12. If m denotes the dimension of the given instance of CVP, then the dimension n of the constructed instance of ROP satisfies $m \leq n \leq O(m^2 \log m)$.

Using [29], we choose c such that $\text{CVP}_{2\gamma}$ is NP-hard with the approximation factor $2\gamma(m) = m^{c/\log \log m}$. Since $m \leq n \leq m^3$ for sufficiently large m , we have

$$\gamma(m) = \frac{1}{2}m^{c/\log \log m} \geq \frac{1}{2}n^{c/3 \log \log n} \geq n^{c'/\log \log n}$$

for a suitably chosen c' . Thus we see that $\text{ROP}(G, \delta)_\gamma$ is NP-hard for the approximation factor $n^{c'/\log \log n}$, which completes the proof. ◀

6.3 Proof of Theorem 1.5

Recall that $L \in \mathbb{Z}^{m \times n}$ is called *right-invertible* over \mathbb{Z} if there exists $R \in \mathbb{Z}^{n \times m}$ such that $LR = I_m$. This means that the lattice generated by the columns of L equals \mathbb{Z}^m , that is, $L(\mathbb{Z}^n) = \mathbb{Z}^m$.

▶ **Proposition 6.8.** *Suppose $\mathcal{L} \subset \mathbb{R}^m$ is a lattice generated by the columns of $G \in \mathbb{Z}^{m \times m}$ with $\det G \neq 0$. Then the following are equivalent for integers $n \geq m$ and $s \in \mathbb{Z}_{>0}$:*

(1) *There exists a lattice $\mathcal{L}' \subset \mathbb{R}^n$ generated by an orthonormal basis such that*

$$\mathcal{L} = sP(\mathcal{L}')$$

where $P : \mathbb{R}^n \rightarrow \mathbb{R}^m$ denotes the orthogonal projection onto the first m -coordinates.

(2) *There exists a matrix $L \in \mathbb{Z}^{m \times n}$ that is right-invertible over \mathbb{Z} , such that*

$$(GL)(GL)^T = s^2 I_m.$$

We need the following observation, which easily follows with Gram-Schmidt orthogonalization.

▶ **Lemma 6.9.** *Suppose $m \leq n$ and $X \in \mathbb{R}^{m \times n}$. Then X can be completed to an orthogonal matrix if and only if $XX^T = I_m$.*

Proof of Proposition 6.8. (1) \Rightarrow (2): Suppose $v_1, v_2, \dots, v_n \in \mathbb{R}^n$ is an orthonormal basis generating \mathcal{L}' and put $u_i := P(v_i)$. The matrix $U \in \mathbb{R}^{m \times n}$ with columns u_1, \dots, u_n has orthogonal rows, that is, $UU^T = I_m$. By assumption, the vectors su_i are in \mathcal{L} and hence can be written as integer linear combinations of the columns of G . This means that there exists an integer matrix $L \in \mathbb{Z}^{m \times n}$ such that $GL = sU$. The rows of this matrix are orthogonal, hence $(GL)(GL)^T = s^2 I_m$. It remains to show that L is right-invertible over \mathbb{Z} . Also by assumption, since the su_i generate \mathcal{L} , the columns of G can be written as integer linear combinations of u_i . Thus there exists an integer matrix $R \in \mathbb{Z}^{n \times m}$ such that $(GL)R = sUR = G$. Hence $LR = I_m$, since $\det G \neq 0$, so that L is indeed right-invertible.

(2) \Rightarrow (1): Suppose $s^2 I_m = (GL)(GL)^T$. Thus the matrix $X := \frac{1}{s}GL$ satisfies $XX^T = I_m$. By Lemma 6.9 we can complete X to an orthogonal matrix: thus there exists an orthogonal matrix $Y \in O_n$ whose first m rows constitute the matrix $\frac{1}{s}GL$. Therefore $sPY = GL$. Hence, if \mathcal{L}' denotes the lattice generated by the columns of Y , we have $sP(\mathcal{L}') \subset \mathcal{L}$. Equality holds since L is right-invertible over \mathbb{Z} . ◀

14:36 Complexity of Robust Orbit Problems for Torus Actions and the abc -Conjecture

Proposition 6.8 indicates a strategy for proving Theorem 1.5. We need two ingredients:

- A polynomial time algorithm for computing on input G an integer $n \geq m$ and a right invertible matrix $L \in \mathbb{Z}^{m \times n}$ such that $(GL)(GL)^T = s^2 I_m$.
- An efficient version of Lemma 6.9.

We prove now that both requirements can be satisfied. For the first step we will use the following result.

► **Theorem 6.10** (Efficient Waring decomposition for quadratic forms). *Assume $A \in \mathbb{Q}^{m \times m}$ is a positive definite, symmetric matrix of bit-length b . Then, in $\text{poly}(m, b)$ -time, we can compute N vectors $l_1, l_2, \dots, l_N \in \mathbb{Q}^m$, where $m \leq N \leq O(m(\log m + \log b))$, such that*

$$A = \sum_{i=1}^N l_i l_i^T.$$

► **Remark 6.11.** In 1932, Mordell [69] considered what he called the *Waring's problem for quadratic forms*, the problem of writing a given positive definite quadratic form $f(x) := x^T A x$ as sum of squares of linear forms over \mathbb{Q} . Note that this equivalent to writing the matrix A as in Theorem 6.10. Mordell proved that a Waring decomposition with $N = m + 3$ is always possible, and described a method for computing the linear forms. Unfortunately, his method relies on Lagrange's four squares theorem that every positive integer D can be written as the sum of four squares $D = a^2 + b^2 + c^2 + d^2$. Randomized polynomial time algorithms to compute a, b, c, d are available [77], but to the best of our knowledge, it is still an open problem whether a polynomial time deterministic algorithm exists. Instead, we will use the lemma below, which uses more squares but can easily be shown to run in deterministic polynomial time.

The proof of Theorem 6.10 requires some preparations.

► **Lemma 6.12.** *Given a positive integer D , we can in polynomial time compute $k = O(\log \log D)$ integers a_1, a_2, \dots, a_k such that*

$$D = a_1^2 + a_2^2 + \dots + a_k^2.$$

Proof. We first compute $a_1 := \lfloor \sqrt{D} \rfloor$. Replacing D with $D' := D - a_1^2$ and repeating the process, we compute integers a_1, a_2, \dots, a_k such that $D = a_1^2 + \dots + a_k^2$. Let us bound k : since $a_1^2 \leq D < (a_1 + 1)^2$, we have $D' = D - a_1^2 \leq (a_1 + 1)^2 - 1 - a_1^2 \leq 2\sqrt{D}$. This implies that if D_j is computed in the j -th step, then $D_j \leq 2\sqrt{D_{j-1}} \leq \dots \leq 2^{1+\frac{1}{2}+\dots+\frac{1}{2^{j-1}}} D^{2^{-j}} \leq 4D^{2^{-j}}$. We deduce that after $k := \log_2 \log_2 D$ steps we have $D_k \leq 8$. The algorithm terminates after at most 4 more steps. ◀

The following result can be found in [62, Algorithm 12.1].

► **Lemma 6.13** (Lagrange's method for congruence diagonalization). *Suppose $X \in \mathbb{Z}^{m \times m}$ is a symmetric matrix. Then we can compute in polynomial time a matrix $Q \in \mathbb{Z}^{m \times m}$ with $\det Q \neq 0$ such that*

$$QXQ^T = \text{diag}(d_1, d_2, \dots, d_m) \in \mathbb{Z}^{m \times m}$$

is diagonal with integer entries.

Proof of Theorem 6.10. By multiplying A with the square of the least common multiple of the denominators of the entries of A , we may assume without loss of generality that A is integral. We compute a matrix Q such that $QAQ^T = \text{diag}(d_1, d_2, \dots, d_m)$ by Lemma 6.13. Since A is positive definite, all d_i are positive. Using Lemma 6.12, for each $i = 1, 2, \dots, m$, we compute e_{ij} such that $d_i = \sum_{j=1}^{k_i} e_{ij}^2$. Denote by L the matrix

$$L := \begin{bmatrix} e_{11} & \dots & e_{1k_1} & & & \\ & & & e_{21} & \dots & e_{2k_2} \\ & & & & & \ddots \\ & & & & & & e_{m1} & \dots & e_{mk_m} \end{bmatrix} \in \mathbb{Z}^{m \times \sum k_i}.$$

Then $LL^T = \text{diag}(d_1, d_2, \dots, d_m) = QAQ^T$, so $A = (Q^{-1}L)(Q^{-1}L)^T$. Defining $l_i \in \mathbb{Q}^m$ to be the i -th column of $Q^{-1}L$, we have $A = \sum_{i=1}^N l_i l_i^T$ where $N := \sum k_i$ and this proves the claim. For the bound on N , we first note that since the algorithm in Lemma 6.13 runs in polynomial time, we have $\log d_i = (bm)^{O(1)}$. Lemma 6.12 then implies that $k_i = O(\log \log d_i) = O(\log(bm))$ and $N = O(m \log(bm))$. ◀

The efficient Waring decomposition is the first ingredient in our proof of Theorem 1.5. The second ingredient is an efficient version of Lemma 6.9.

► **Lemma 6.14.** Suppose $m \leq n$ and $X \in \mathbb{Q}^{m \times n}$ satisfies $XX^T = I_m$. Then, we can compute in polynomial time an orthogonal matrix $Y \in \mathbb{Q}^{n \times n}$ such that the first m rows of Y are the rows of X .

Proof. We construct Y as a product of reflections at rational hyperplanes.

First note that for any $v, w \in \mathbb{Q}^n$ of norm one, there is an orthogonal matrix $Y \in \mathbb{Q}^{n \times n}$ such that $Yv = w$. Indeed, w.l.o.g. $v \neq -w$, and take for Y the reflection at the hyperplane orthogonal to $v + w$, which is given by the linear map

$$Yx := 2 \frac{\langle x, v + w \rangle}{\|v + w\|^2} (v + w) - x.$$

Note that Y defines a rational orthogonal matrix and indeed $Yv = w$. Denote by e_i the standard basis vectors.

Given X as in the lemma, we compute an orthogonal matrix $Y_1 \in \mathbb{Q}^{n \times n}$ which maps e_1 to the first row of X . Since Y is orthogonal and $Y^2 = I_n$, the first row of Y equals the first row of X . We construct Y by iterating this process. ◀

► **Remark 6.15.** M. Hall [41, 42] gave necessary conditions for an integer matrix to be completable to a scalar multiple of an orthogonal matrix in the sense of Lemma 6.14.

Proof of Theorem 1.5. Given $G \in \mathbb{Z}^{m \times m}$ with $\det G \neq 0$, we compute $s := m\|G\|_{\max} + 1$. By Lemma 2.7 we have $\sigma_{\max}(G) < s$. The minimum eigenvalue of $G^{-1}G^{-T}$ satisfies

$$\lambda_{\min}(G^{-1}G^{-T}) = \sigma_{\min}^2(G^{-1}) = \frac{1}{\sigma_{\max}^2(G)} > \frac{1}{s^2}.$$

Consequently, the matrix $A := s^2(G^{-1}G^{-T}) - I_m$ is positive definite.

Using Theorem 6.10, we compute in polynomial time a number N and a matrix $L \in \mathbb{Q}^{m \times N}$ such that $A = LL^T$. Let $f \in \mathbb{Z}_{>0}$ denote the least common multiple of the denominators of the entries of L , so that $L' := fL$ is integral. Then $(sf)^2 G^{-1}G^{-T} - f^2 I_m = L'(L')^T$.

Using Lemma 6.12, we compute integers b_1, \dots, b_p such that $f^2 - 1 = b_1^2 + b_2^2 + \dots + b_p^2$. Since L is computed from A in polynomial time, the bit-length of f is $\text{poly}(b, m)$ so Lemma 6.12 implies $p = O(\log(bm))$. Considering the integer matrix $L'' = [b_1 I_m \mid b_2 I_m \mid \dots \mid b_p I_m \mid L']$ of format $m \times (pm + N)$, we get

$$(sf)^2 G^{-1} G^{-T} - I_m = (b_1^2 + b_2^2 + \dots + b_p^2) I_m + L'(L')^T = L''(L'')^T. \tag{6.2}$$

We now consider the matrix $X := [G \mid GL'']$. By Equation (6.2), we have $XX^T = (sf)^2 I_m$. Using Lemma 6.14 with input $(sf)^{-1} X$, we compute a rational, orthogonal matrix $Y \in \mathbb{Q}^{n \times n}$ such that the first m rows of Y are the rows of $(sf)^{-1} X$ where $n := m + (pm + N)$.

The columns v_1, \dots, v_n of Y form an orthonormal basis. Moreover, the orthogonal projections $P(v_i)$ onto the first m coordinates equal the columns of $(sf)^{-1} X$. Thus if we denote by \mathcal{L}' the lattice generated by v_1, \dots, v_n , then $sfP(\mathcal{L}')$ is the lattice spanned by the columns of $X = [G \mid GL'']$, which is equal to $\mathcal{L} = G(\mathbb{Z}^m)$, because L'' is an integer matrix. We conclude that $sfP(\mathcal{L}') = \mathcal{L}$. ◀

7 Orbit Problems and the Kempf-Ness Approach

In Section 1.9 we outlined a general approach for the Orbit Equality Problem, based on the Kempf-Ness theorem [55]. Here we analyze this approach for torus actions. We formulate a conjecture on the complexity of computing approximate solutions to unconstrained geometric programs. A positive answer to the conjecture leads to a numerical algorithm for the Orbit Equality Problem that runs in polynomial time, provided Separation Hypothesis 1.7 is true.

7.1 The general Kempf-Ness theorem

This theorem holds for any reductive group T (not just a torus) with a rational action on a finite dimensional vector space V . We denote by K a maximal compact subgroup of T and assume that V is endowed with a Hermitian inner product such that K acts by isometries. The inner product defines a norm and Euclidean metric on V . In the special case $T = (\mathbb{C}^\times)^d$ of a torus we have $K = (S^1)^d$.

It is well known [24, Thm. 2.3.6] that each orbit closure $\overline{\mathcal{O}_v}$ contains a unique closed orbit. Therefore, we have $\mathcal{O}_v \cap \mathcal{O}_w \neq \emptyset$ iff the orbit closures $\overline{\mathcal{O}_v}$ and $\overline{\mathcal{O}_w}$ share the same closed orbit. We therefore focus on closed orbits: let us call a vector $v \in V$ *polystable* if its orbit \mathcal{O}_v is closed. We denote by V^{ps} the set of polystable vectors in V . As in Section 2.5, we can endow the space V^{ps}/T of closed orbits with the quotient topology, which is Hausdorff. The Kempf-Ness Theorem [55] states that the space V^{ps}/T is homeomorphic to a “smaller object”, which is defined in analytic terms, namely the space $\text{Crit}(V)/K$ of K -orbits of the closed and K -invariant subset $\text{Crit}(V)$ of critical points of V .

The critical points are defined in terms of the *Kempf-Ness function*:

$$F_v : T \rightarrow \mathbb{R}, \quad F_v(g) := \log \|g \cdot v\| = \frac{1}{2} \log \|g \cdot v\|^2. \tag{7.1}$$

Let us denote the induced action of $\text{Lie}(T)$ on V by $x \cdot v := \left. \frac{d}{dt} \right|_{t=0} (e^{tx} \cdot v)$, for $x \in \text{Lie}(T)$ and $v \in V$. Note that $x \cdot v = v + M^T x$ for the action Equation (1.1) of a torus. One checks that the derivative of F_v at I is given by $D_I F_v x = \|v\|^{-2} \langle x \cdot v, v \rangle$. Thus we define $v \in V$ to be *critical* iff $\langle x \cdot v, v \rangle = 0$ for all $x \in \text{Lie}(T)$. By definition, the set $\text{Crit}(V)$ of critical vectors in V is a closed real algebraic set in V , cut out by real quadratic polynomials. Moreover, $\text{Crit}(V)$ is K -invariant, since we assume the Hermitian inner product to be K -invariant.

It is easy to see that closed orbits contain a critical point. The celebrated Kempf-Ness theorem [55] states the converse: the orbits of critical points are closed, thus $\text{Crit}(V) \subset V^{ps}$. Even more is known:

- **Theorem 7.1** (The Kempf-Ness theorem). (a) *The orbit \mathcal{O}_v is closed if and only if $\mathcal{O}_v \cap \text{Crit}(V) \neq \emptyset$.*
- (b) *The intersection $\mathcal{C}_{v^*} := \overline{\mathcal{O}_v} \cap \text{Crit}(V)$ is a single K -orbit, that we call the Kempf-Ness orbit of v .*
- (c) *We have $\overline{\mathcal{O}_v} \cap \overline{\mathcal{O}_w} \neq \emptyset$ if and only if $\mathcal{C}_{v^*} = \mathcal{C}_{w^*}$.*

The proof of Theorem 7.1 relies on convexity properties of the function F_v . One observes that $t \mapsto F_v(\gamma(t))$ is convex for all all curves of the form $\gamma(t) := \text{Exp}(tx) \cdot v$. A more conceptual view of the situation is as follows (see [17]). One can view F_v as a function on $T/K = \{Kg \mid g \in T\}$, since F_v is constant on the cosets Kg . Moreover, T/K is a symmetric space with respect to a K -invariant Riemannian metric, whose geodesics are exactly the curves arising from the γ . For this reason, one calls F_v a *geodesically convex* function on T/K . In the special case where $T = (\mathbb{C}^\times)^d$ is the torus, we have the isometry

$$\mathbb{R}^d \xrightarrow{\sim} T/K, \quad x \mapsto K \text{Exp}(x/2) \tag{7.2}$$

given by the exponential map. In general, the convexity implies that $v^* \in \overline{\mathcal{O}_v}$ is critical iff

$$\|v^*\| = \inf_{v' \in \mathcal{O}_v} \|v'\|.$$

► **Remark 7.2.** The second property in Theorem 7.1 expresses that the continuous map $\text{Crit}(V)/K \rightarrow V^{ps}/T$, induced by the inclusion, is a bijection. Equivalently, this map is a homeomorphism.⁸

7.2 Efficient approximation of the Kempf-Ness orbit

We return to the situation of the action of a torus T on $V = \mathbb{C}^n$. Let $\omega_1, \dots, \omega_n \in \mathbb{Z}^d$ denote the corresponding weights, i.e., the columns of the weight matrix M . We will always assume that the affine span of these weights is \mathbb{R}^d . We fix a vector $v \in V$ and assume for simplicity that $q_i := |v_i|^2 > 0$ for all i . Using the parametrization (7.2) and multiplying by 2 to simplify the presentation, the Kempf-Ness function (7.1) becomes

$$f: \mathbb{R}^d \rightarrow \mathbb{R}, \quad f(x) := 2 \log \|e^{x/2} \cdot v\| = \log \left(\sum_{i=1}^n q_i e^{\omega_i^T x} \right). \tag{7.3}$$

The gradient $\nabla f(x)$ and the Hessian of f at x are given by

$$\nabla f(x) = \frac{\sum_{i=1}^n q_i e^{\omega_i^T x} \omega_i}{\sum_{i=1}^n q_i e^{\omega_i^T x}}, \quad \nabla^2 f(x) = \frac{\sum_{i=1}^n q_i e^{\omega_i^T x} \omega_i \omega_i^T}{\sum_{i=1}^n q_i e^{\omega_i^T x}} - \nabla f(x) \nabla f(x)^T. \tag{7.4}$$

This shows that $\nabla f(x)$ lies in the interior of the *weight polytope*

$$P := \text{conv}(\omega_1, \dots, \omega_n).$$

⁸ Pass to projective spaces to see this.

Moreover, the Cauchy-Schwarz inequality implies for all nonzero $y \in \mathbb{R}^d$,

$$y^T \nabla^2 f(x) y = \frac{\sum_{i=1}^n q_i e^{\omega_i^T x} (y^T \omega_i)^2}{\sum_{i=1}^n q_i e^{\omega_i^T x}} - (y^T \nabla f(x))^2 > 0, \quad (7.5)$$

which directly proves that f is a strictly convex function.⁹

For the following equivalences, see [20, 16]. The function f has a finite infimum iff $0 \in P$. This is equivalent to $0 \notin \overline{\mathcal{O}_v}$. Moreover, the infimum f_* is attained iff 0 lies in the interior of P , which is equivalent to the orbit \mathcal{O}_v being closed. By the strict convexity, the minimum is attained at a unique point $x^* \in \mathbb{R}^d$. By Theorem 7.1, $v^* := e^{x^*} \cdot v$ defines the K -orbit \mathcal{C}_{v^*} .

There is a vast amount of literature on the problem of minimizing $f(x)$ using the ellipsoid or the interior point methods, see [20, 14, 74, 80, 83, 60] and the references therein. On input $\varepsilon > 0$, these algorithms return a point $x \in \mathbb{R}^d$ with $f(x) - f_* < \varepsilon$ in polynomial time. However, surprisingly, we are not aware of any result on computing a certified approximation to x^* in polynomial time, and currently this seems to be an open problem. We conjecture that this can be achieved in polynomial time:

► **Conjecture 7.3.** *There exists an algorithm that on input $M \in \mathbb{Z}^{d \times n}$ such that $0 \in \text{int}(P)$, a vector $v \in (\mathbb{Q}(i)^\times)^n$, and $\varepsilon \in \mathbb{Q}_{>0}$, computes $x \in \mathbb{Q}^d$ such that*

$$\|x - x^*\| < \varepsilon,$$

in time $\text{poly}(d, n, B, b, \log \frac{1}{\varepsilon})$, where B is the bit-length of M and b is the bit-length of v .

We will now explain why computing an approximation to x^* seems to be more challenging than approximately minimizing $f(x)$. In Example 7.4 below, we provide a family of Kempf-Ness functions f and points x such that $f(x) - f_*$ becomes doubly-exponentially small in the input bit-length, while the distance to the true minimizer x^* remains constant: $\|x - x^*\| = 1$. This implies that using known minimization algorithms in a black box will not be sufficient to solve Conjecture 7.3 and more specific algorithms appear to be required.

► **Example 7.4.** We consider the action of $T := (\mathbb{C}^\times)^2$ on $V := \mathbb{C}^4$ with the following weights: $\omega_1 := (1, 0)$, $\omega_2 := (-2, 0)$, $\omega_3 := (-N, 1)$, $\omega_4 := (-N, -1)$ where $N > 2$ is a positive integer. See Figure 4 for illustration. The Kempf-Ness function (7.3) for $v := (1, 1, 1, 1)$ reads

$$f(x_1, x_2) = \log \left(e^{x_1} + e^{-2x_1} + e^{-Nx_1+x_2} + e^{-Nx_1-x_2} \right).$$

We note that $f(x) \geq 0$ for every $x \in \mathbb{R}^d$ since at least one of the exponents $x_1, -2x_1$ is non-negative.

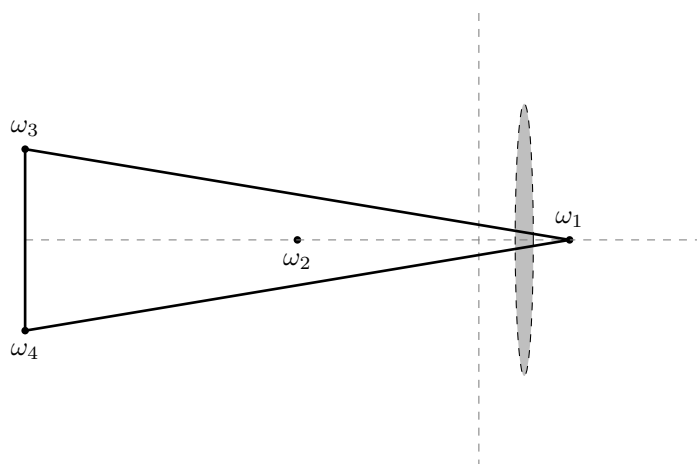
We now focus on the unique minimizer $x^* = (x_1^*, x_2^*)$ of the convex program $f_* := \min_x f(x)$, which is characterized by the property that the gradient vanishes: $\nabla f(x^*) = 0$. By Equation (7.4) this is equivalent to

$$e^{x_1^*} - 2e^{-2x_1^*} - Ne^{-Nx_1^*+x_2^*} - Ne^{-Nx_1^*-x_2^*} = 0 \quad \text{and} \quad e^{-Nx_1^*+x_2^*} - e^{-Nx_1^*-x_2^*} = 0. \quad (7.6)$$

The second equation implies $x_2^* = 0$. The first equation then reduces to

$$e^{x_1^*} - 2e^{-2x_1^*} - 2Ne^{-Nx_1^*} = 0, \quad (7.7)$$

⁹ To see this, we split $\sum_i q_i e^{\omega_i^T x} y^T \omega_i$ as $\sum_i q_i^{\frac{1}{2}} e^{\frac{1}{2} \omega_i^T x} y^T \omega_i \cdot q_i^{\frac{1}{2}} e^{\frac{1}{2} \omega_i^T x}$ and apply the Cauchy-Schwarz inequality. Since the vectors ω_i affinely span \mathbb{R}^d we have $y^T \omega_i \neq y^T \omega_j$ for some $i \neq j$. This shows that the vectors $(q_i^{\frac{1}{2}} e^{\frac{1}{2} \omega_i^T x} y^T \omega_i \mid i \in [n])$ and $(q_i^{\frac{1}{2}} e^{\frac{1}{2} \omega_i^T x} \mid i \in [n])$ cannot be proportional and hence the Cauchy-Schwarz inequality must be strict.



■ **Figure 5** An example of a torus action with weights $(1, 0), (-2, 0), (-N, 1), (-N, -1)$. The points in the shaded region all have small $f(x) - f_*$ but they can be far away from the optimal solution x^* .

which implies that x_1^* is positive and decreases with N . In the limit $N \rightarrow \infty$, the first equation degenerates to $e^{x_1^*} - 2e^{-2x_1^*} = 0$. We deduce that $e^{x_1^*} > \sqrt[3]{2}$ for every $N > 2$. In fact, we have:

$$\sqrt[3]{2} < e^{x_1^*} < \left(1 + \frac{N}{2^{N/3}}\right) \sqrt[3]{2}. \tag{7.8}$$

To see this, we write $e^{x_1^*} = (1 + \varepsilon)\sqrt[3]{2}$ where $\varepsilon > 0$. Then Equation (7.7) implies

$$(1 + \varepsilon)\sqrt[3]{2} - \frac{\sqrt[3]{2}}{(1 + \varepsilon)^2} = \frac{2N}{(1 + \varepsilon)^N 2^{N/3}}.$$

Multiplying this equation by $(1 + \varepsilon)^N / \sqrt[3]{2}$, we get $(1 + \varepsilon)^{N-2} (\varepsilon^2 + 3\varepsilon + 3) \varepsilon = 2N \times 2^{-(N+1)/3}$. Since $\varepsilon > 0$, we have $\varepsilon^2 + 3\varepsilon + 3 > 3$ and $3\varepsilon < 2N \times 2^{-(N+1)/3}$, which implies Equation (7.8).

We now consider the point $x := (x_1^*, 1)$, whose first coordinate agrees with the first coordinate of $x^* = (x_1^*, 0)$ but has a constant distance away from x^* in the second coordinate. We will show that $f(x) - f_*$ is exponentially small in N :

$$e^{f(x)-f_*} = 1 + \frac{e^{-Nx_1^*}(e + e^{-1} - 2)}{e^{f_*}} \leq 1 + e^{-Nx_1^*}(e + e^{-1} - 2)$$

where the inequality holds since $f_* \geq 0$. Taking logarithms we get

$$f(x) - f_* \leq \log(1 + e^{-Nx_1^*}(e + e^{-1} - 2)) \leq e^{-Nx_1^*}(e + e^{-1} - 2) < 2^{-N/3}(e + e^{-1} - 2)$$

since $e^{x_1^*} > \sqrt[3]{2}$. This shows that $f(x) - f_*$ is exponentially small in N and hence doubly-exponentially small in the input bit-length, and yet $\|x - x^*\| = 1$.

► **Remark 7.5.** In the setting of Example 7.4, consider the vectors $v := (1, 1, 1, 1)$ and $w := (1, 1, 2, 2)$, and denote by x^*, y^* the minimizers of the Kempf-Ness functions of v, w , respectively, i.e., $e^{x^*} = v^*$ and $e^{y^*} = w^*$. By Example 7.4, we have $x^* = (x_1^*, 0)$, where x_1^* satisfies Equation (7.8). By a similar reasoning, we also have $y^* = (y_1^*, 0)$, where y_1^* satisfies

$$\sqrt[3]{2} < e^{y_1^*} < \left(1 + \frac{2N}{2^{N/3}}\right) \sqrt[3]{2}.$$

We will now show that the coordinates of v^*, w^* are exponentially close in N . To this end, denote $\varepsilon_N := 2N \times 2^{-N/3}$. We have $|(v^*)_1 - (w^*)_1| = |e^{x_1^*} - e^{y_1^*}| < 2^{1/3} \varepsilon_N$. For the second coordinate we have $|(v^*)_2 - (w^*)_2| = |e^{-2x_1^*} - e^{-2y_1^*}| < 2^{-2/3}((1 + \varepsilon_N)^2 - 1) < 2^{4/3} \varepsilon_N$, using the crude bound $(1 + \varepsilon_N)^2 - 1 < 4\varepsilon_N$, which holds for $\varepsilon_N < 2$ and hence for N large enough. For the third and fourth coordinate we have $|(v^*)_3 - (w^*)_3| = |(v^*)_4 - (w^*)_4| = |e^{-Nx_1^*} - 2e^{-Ny_1^*}| < |e^{-Nx_1^*}| + 2|e^{-Ny_1^*}| < 3 \times 2^{-N/3} < \varepsilon_N$. Hence $\|v^* - w^*\|_\infty < 2^{4/3} \varepsilon_N$ and we conclude

$$\text{dist}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) \leq \|v^* - w^*\| \leq 2\|v^* - w^*\|_\infty < 2^{7/3} \varepsilon_N$$

where the second inequality follows from the inequality $\|\cdot\|_2 \leq \sqrt{n}\|\cdot\|_\infty$ which holds in the n -dimensional Euclidean space. This shows that the Euclidean distance between \mathcal{C}_{v^*} and \mathcal{C}_{w^*} can be exponentially small in N and doubly exponentially small in the bit-length of N . On the other hand, it is easy to lower bound the logarithmic distance: It is easy to verify that

$$H = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 2N & 0 & 1 & 1 \end{bmatrix}$$

is the matrix of rational invariants and $H \text{Log } v = (0, 0)$ and $H \text{Log } w = (0, 2 \log 2)$. Proposition 3.5 then implies

$$\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) \geq \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \frac{1}{\sigma_{\max}(H)} \Delta(H \text{Log } v, H \text{Log } w) \geq \frac{\log 2}{2N}$$

where in the last inequality we use Lemma 2.7 to upper bound $\sigma_{\max}(H) \leq 4N$.

More generally, if Separation Hypothesis 1.7 holds then it is true for general torus actions that $\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*})$ is at most singly exponentially small: If $\mathcal{C}_{v^*} \neq \mathcal{C}_{w^*}$, then, in general, $\mathcal{O}_v \neq \mathcal{O}_w$, hence, $\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) \geq \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \text{sep}_T(d, n, B, b)$. This is one of the main reasons why we work with the logarithmic distance instead of the Euclidean distance.

7.3 Deciding the equality of the Kempf-Ness orbits

We now show that, assuming Separation Hypothesis 1.7 and Conjecture 7.3, that for given $v, w \in (\mathbb{C}^\times)^n$, it is possible to decide in polynomial time whether the orbits \mathcal{O}_v for \mathcal{O}_w are equal. By Theorem 7.1, this is equivalent to the equality of the Kempf-Ness orbits \mathcal{C}_{v^*} and \mathcal{C}_{w^*} . We decide this by computing the approximate distance of the corresponding orbits \mathcal{C}_{v^*} and \mathcal{C}_{w^*} with sufficient accuracy. This has two ingredients: first we compute x' and y' that are ε -close approximations of x^* and y^* , respectively. This is possible by Conjecture 7.3. In a second step, we compute an approximation D of the distance $\delta := \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'})$ between $\mathcal{C}_{v'}$ and $\mathcal{C}_{w'}$ such that $\delta \leq D \leq \gamma\delta$. For this we use Theorem 5.2 (c), see Corollary 7.6 below. When choosing $\varepsilon = 2^{-\ell}$ sufficiently small, we show that $\mathcal{C}_{v^*} = \mathcal{C}_{w^*}$ iff $D < \gamma\varepsilon$, which allows to decide orbit equality. Separation Hypothesis 1.7 is needed to make sure that it is sufficient to choose ℓ as a polynomial in the input bit-length in order to obtain a polynomial time algorithm.

We now give the technical details. We recall that we restrict ourselves to torus actions where 0 lies in the interior of the convex hull P of the set of weights Ω . This condition guarantees that *all* orbits of vectors in $(\mathbb{C}^\times)^n$ are closed in \mathbb{C}^n ; on the other hand, when $0 \notin \text{int}(P)$, then no such orbit is closed, see [16].

We first state the following straightforward consequence of Theorem 5.2 (c).

► **Corollary 7.6.** *We assume the Separation Hypothesis 1.7. There is an exponentially bounded approximation factor $\gamma = \gamma(d, n, B, b)$ such that, given a T -action by a weight matrix $M \in \mathbb{Z}^{d \times n}$, and given vectors $v, w \in (\mathbb{Q}(i)^\times)^n$ and $x, y \in \mathbb{Q}^d$, we can compute in polynomial time a number $D \in \mathbb{Q}_{\geq 0}$ such that*

$$\delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) \leq D \leq \gamma \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}),$$

where $v' := e^x \cdot v$ and $w' := e^y \cdot w$.

Using the approximation factor γ of Corollary 7.6, we define the parameter

$$\varepsilon := \frac{1}{2\gamma} \text{sep}_T(d, n, B, b).$$

Assuming Conjecture 7.3, we can compute in polynomial time approximations

$$\|x - x^*\| < \frac{1}{2} \|M\|^{-1} \varepsilon, \quad \|y - y^*\| < \frac{1}{2} \|M\|^{-1} \varepsilon.$$

Recall that $v^* = e^{x^*} \cdot v$. We set $v' := e^x \cdot v$. Then $\text{Log } v^* = \text{Log } v + M^T x^*$ and $\text{Log } v' = \text{Log } v + M^T x$, hence

$$\delta_{\log}(v', v^*) = \Delta(\text{Log } v + M^T x, \text{Log } v + M^T x^*) = \|M^T(x - x^*)\| \leq \frac{1}{2} \varepsilon,$$

which implies $\delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{v^*}) \leq \frac{1}{2} \varepsilon$. Analogously, we get $\delta_{\log}(\mathcal{C}_{w'}, \mathcal{C}_{w^*}) \leq \frac{1}{2} \varepsilon$ for $w' := e^y \cdot w$. Using the triangle inequality, we obtain

$$\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) \leq \delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{v'}) + \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) + \delta_{\log}(\mathcal{C}_{w'}, \mathcal{C}_{w^*}) \leq \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) + \varepsilon.$$

By symmetry, this implies

$$|\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) - \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'})| < \varepsilon. \quad (7.9)$$

We now observe the following two implications:

$$\begin{aligned} \delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) = 0 &\implies \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) < \varepsilon \\ \delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) > 0 &\implies \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) > \gamma \varepsilon. \end{aligned} \quad (7.10)$$

The first implication is an obvious consequence of Equation (7.9). For the second implication, note that $\mathcal{C}_{v^*} \neq \mathcal{C}_{w^*}$ implies $\delta_{\log}(\mathcal{C}_{v^*}, \mathcal{C}_{w^*}) \geq \delta_{\log}(\mathcal{O}_v, \mathcal{O}_w) \geq \text{sep}_T$, since $\mathcal{C}_{v^*} \subset \mathcal{O}_v$ and $\mathcal{C}_{w^*} \subset \mathcal{O}_w$. Combined with Equation (7.9), this gives

$$\delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'}) > \text{sep}_T - \varepsilon = 2\gamma\varepsilon - \varepsilon \geq \gamma\varepsilon,$$

which proves Equation (7.10).

We use Corollary 7.6 to compute in polynomial time the number D such that $\delta \leq D \leq \gamma\delta$, where $\delta := \delta_{\log}(\mathcal{C}_{v'}, \mathcal{C}_{w'})$. Equation (7.10) implies now:

$$\begin{aligned} \mathcal{C}_{v^*} = \mathcal{C}_{w^*} &\implies D < \gamma\varepsilon \\ \mathcal{C}_{v^*} \neq \mathcal{C}_{w^*} &\implies D > \gamma\varepsilon. \end{aligned}$$

So the knowledge of D allows to decide whether $\mathcal{C}_{v^*} = \mathcal{C}_{w^*}$ and hence whether $\mathcal{O}_v = \mathcal{O}_w$.

References

- 1 Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.
- 2 Dorit Aharonov and Oded Regev. Lattice problems in $\text{NP} \cap \text{coNP}$. *J. ACM*, 52(5):749–765, September 2005. doi:10.1145/1089023.1089025.
- 3 Zeyuan Allen-Zhu, Ankit Garg, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Operator scaling via geodesically convex optimization, invariant theory and polynomial identity testing. In *STOC'18 – Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 172–181. ACM, New York, 2018. doi:10.1145/3188745.3188942.
- 4 Carlos Améndola, Kathlén Kohn, Philipp Reichenbach, and Anna Seigal. Invariant theory and scaling algorithms for maximum likelihood estimation. *SIAM J. Appl. Algebra Geom.*, 5(2):304–337, 2021. doi:10.1137/20M1328932.
- 5 Carlos Améndola, Kathlén Kohn, Philipp Reichenbach, and Anna Seigal. Toric invariant theory for maximum likelihood estimation in log-linear models. *Algebraic Statistics*, 12(2):187–211, 2021.
- 6 Michele Audin. *Torus actions on symplectic manifolds*, volume 93 of *Progress in Mathematics*. Birkhäuser Basel, 2012.
- 7 Alan Baker. Experiments on the abc-conjecture. *Publicationes Mathematicae*, 65, November 2004.
- 8 Alan Baker. Logarithmic forms and the abc-conjecture. In *Number Theory: Diophantine, Computational and Algebraic Aspects. Proceedings of the International Conference held in Eger, Hungary, July 29-August 2, 1996*, pages 37–44. De Gruyter, 2011. doi:10.1515/9783110809794.37.
- 9 Alan Baker and Gisbert Wüstholz. *Logarithmic Forms and Diophantine Geometry*. New Mathematical Monographs. Cambridge University Press, 2008. doi:10.1017/CB09780511542862.
- 10 Markus Bläser, Christian Ikenmeyer, Vladimir Lysikov, Anurag Pandey, and Frank-Olaf Schreyer. Variety membership testing, algebraic natural proofs, and geometric complexity theory. arXiv:1911.02534, 2020.
- 11 Enrico Bombieri and Walter Gubler. *Heights in Diophantine Geometry*. New Mathematical Monographs. Cambridge University Press, 2006. doi:10.1017/CB09780511542879.
- 12 Jonathan M. Borwein and Peter B. Borwein. *Pi and the AGM : a study in analytic number theory and computational complexity*. Canadian Mathematical Society series of monographs and advanced texts. Wiley, New York, 1987.
- 13 Jonathan M. Borwein and Peter B. Borwein. On the complexity of familiar functions and numbers. *SIAM Rev.*, 30(4):589–601, 1988. doi:10.1137/1030134.
- 14 Stephen Boyd, Seung-Jean Kim, Lieven Vandenbergh, and Arash Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, March 2007. doi:10.1007/s11081-007-9001-7.
- 15 Peter Bürgisser, Matthias Christandl, Ketan Mulmuley, and Michael Walter. Membership in moment polytopes is in NP and coNP. *SIAM J. Comput.*, 46(3):972–991, 2017. doi:10.1137/15M1048859.
- 16 Peter Bürgisser, Levent Doğan, Visu Makam, Michael Walter, and Avi Wigderson. Polynomial Time Algorithms in Invariant Theory for Torus Actions. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:30, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2021.32.
- 17 Peter Bürgisser, Cole Franks, Ankit Garg, Rafael Mendes de Oliveira, Michael Walter, and Avi Wigderson. Towards a theory of non-commutative optimization: geodesic first and second order methods for moment maps and polytopes. In *60th Annual IEEE Symposium on Foundations of Computer Science – FOCS 2019*, pages 845–861. IEEE Computer Soc., Los Alamitos, CA, 2019. arXiv:1910.12375.

- 18 Peter Bürgisser, Cole Franks, Ankit Garg, Rafael Oliveira, Michael Walter, and Avi Wigderson. Efficient algorithms for tensor scaling, quantum marginals, and moment polytopes. In *59th Annual IEEE Symposium on Foundations of Computer Science – FOCS 2018*, pages 883–897. IEEE Computer Soc., Los Alamitos, CA, 2018. doi:10.1109/FOCS.2018.00088.
- 19 Peter Bürgisser, Ankit Garg, Rafael Oliveira, Michael Walter, and Avi Wigderson. Alternating minimization, scaling algorithms, and the null-cone problem from invariant theory. In *9th Innovations in Theoretical Computer Science*, volume 94 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 24, 20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.
- 20 Peter Bürgisser, Yinan Li, Harold Nieuwboer, and Michael Walter. Interior-point methods for unconstrained geometric programming and scaling problems. arXiv:2008.12110, 2020.
- 21 Giuseppe C. Calafiore, Stephane Gaubert, and Corrado Possieri. Log-sum-exp neural networks and posynomial models for convex and log-log-convex data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(3):827–838, 2020. doi:10.1109/TNNLS.2019.2910417.
- 22 Giuseppe C. Calafiore, Stephane Gaubert, and Corrado Possieri. A universal approximation result for difference of log-sum-exp neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12):5603–5612, 2020. doi:10.1109/TNNLS.2020.2975051.
- 23 John H. Conway and Neil J. A. Sloane. Low-dimensional lattices v. integral coordinates for integral lattices. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 426(1871):211–232, 1989. URL: <http://www.jstor.org/stable/2398341>.
- 24 Harm Derksen and Gregor Kemper. *Computational Invariant Theory*. BV035421342 Encyclopaedia of Mathematical Sciences volume 130. Springer, Heidelberg ; New York ; Dordrecht ; London, second enlarged edition with two appendices by vladimir l. popov, and an addendum by norbert a. campo and vladimir l. popov edition, 2015.
- 25 Harm Derksen and Visu Makam. Polynomial degree bounds for matrix semi-invariants. *Adv. Math.*, 310:44–63, 2017. doi:10.1016/j.aim.2017.01.018.
- 26 Harm Derksen and Visu Makam. Algorithms for orbit closure separation for invariants and semi-invariants of matrices. *Algebra Number Theory*, 14(10):2791–2813, 2020. doi:10.2140/ant.2020.14.2791.
- 27 Harm Derksen and Visu Makam. Maximum likelihood estimation for matrix normal models via quiver representations. *SIAM Journal on Applied Algebra and Geometry*, 5(2):338–365, 2021.
- 28 Harm Derksen, Visu Makam, and Michael Walter. Maximum likelihood estimation for tensor normal models via castling transforms. In *Forum of Mathematics, Sigma*, volume 10, page e50. Cambridge University Press, 2022.
- 29 Irit Dinur, Guy Kindler, Ran Raz, and Shmuel Safra. Approximating CVP to within-almost polynomial factors is NP-hard. *Combinatorica*, 23(2):205–243, April 2003. doi:10.1007/s00493-003-0019-y.
- 30 Kousha Etessami, Alistair Stewart, and Mihalis Yannakakis. A note on the complexity of comparing succinctly represented integers, with an application to maximum probability parsing. *ACM Trans. Comput. Theory*, 6(2), May 2014. doi:10.1145/2601327.
- 31 Michael A. Forbes and Amir Shpilka. Explicit Noether normalization for simultaneous conjugation via polynomial identity testing. In *Approximation, randomization, and combinatorial optimization*, volume 8096 of *Lecture Notes in Comput. Sci.*, pages 527–542. Springer, Heidelberg, 2013. doi:10.1007/978-3-642-40328-6_37.
- 32 Cole Franks, Rafael Oliveira, Akshay Ramachandran, and Michael Walter. Near optimal sample complexity for matrix and tensor normal models via geodesic convexity. *arXiv preprint*, 2021. arXiv:2110.07583.
- 33 Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. In *57th Annual IEEE Symposium on Foundations of Computer Science – FOCS 2016*, pages 109–117. IEEE Computer Soc., Los Alamitos, CA, 2016. doi:10.1109/FOCS.2016.95.

- 34 Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. Operator scaling: theory and applications. *Found. Comput. Math.*, 20(2):223–290, 2020. doi:10.1007/s10208-019-09417-z.
- 35 Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. Association for Computing Machinery. doi:10.1145/1536414.1536440.
- 36 Dorian Goldfeld. Beyond the last theorem. *Math Horizons*, 4(1):26–34, 1996. doi:10.1080/10724117.1996.11974985.
- 37 Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski, editor, *Advances in Cryptology – CRYPTO '97*, pages 112–131, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.
- 38 Andrew Granville and Thomas Tucker. It's as easy as abc. *Notices of the American Mathematical Society*, 49, January 2002.
- 39 Victor Guillemin and Shlomo Sternberg. *Symplectic techniques in physics*. Cambridge Univ. Press, Cambridge u.a., 1. publ., reprint. edition, 1986.
- 40 Leonid Gurvits. Classical complexity and quantum entanglement. *J. Comput. Syst. Sci.*, 69(3):448–484, 2004. doi:10.1016/j.jcss.2004.06.003.
- 41 Marshall Hall. Integral matrices a for which $AA^T = mI$. *Number Theory and Algebra*, pages 119–134, 1977. URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84887022329&partnerID=40&md5=5f91330ec7e4ddf22586cd2717a71b3a>.
- 42 Marshall Hall. Combinatorial completions. In *Advances in Graph Theory*, volume 3 of *Annals of Discrete Mathematics*, pages 111–123. Elsevier, 1978. doi:10.1016/S0167-5060(08)70501-6.
- 43 Masaki Hamada and Hiroshi Hirai. Computing the nc-rank via discrete convex optimization on $\text{cat}(0)$ spaces. *SIAM Journal on Applied Algebra and Geometry*, 5(3):455–478, 2021. doi:10.1137/20M138836X.
- 44 Godfrey H. Hardy and Edward M. Wright. *An introduction to the theory of numbers*. Oxford University Press, Oxford, sixth edition, 2008. Revised by D. R. Heath-Brown and J. H. Silverman, With a foreword by Andrew Wiles.
- 45 Charles J. Himmelberg. Pseudo-metrizability of quotient spaces. *Fund. Math.*, pages 1–6, 1968.
- 46 Warren Hoburg, Philippe Kirschen, and Pieter Abbeel. Data fitting with geometric-programming-compatible softmax functions. *Optimization and Engineering*, 17(4):897–918, December 2016. doi:10.1007/s11081-016-9332-3.
- 47 Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Ntru: A ring-based public key cryptosystem. In Joe P. Buhler, editor, *Algorithmic Number Theory*, pages 267–288, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.
- 48 Jeffrey Hoffstein and Joseph Silverman. Optimizations for ntru. In Kazimierz Alster, Jerzy Urbanowicz, and Hugh C. Williams, editors, *Proceedings of the International Conference organized by the Stefan Banach International Mathematical Center Warsaw, Poland, September 11-15, 2000*, pages 77–88, Berlin, New York, 2001. De Gruyter. doi:doi:10.1515/9783110881035.77.
- 49 Gábor Ivanyos and Youming Qiao. *On the orbit closure intersection problems for matrix tuples under conjugation and left-right actions*, pages 4115–4126. Society for Industrial and Applied Mathematics, 2023. doi:10.1137/1.9781611977554.ch158.
- 50 Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Non-commutative Edmonds' problem and matrix semi-invariants. *Comput. Complexity*, 26(3):717–763, 2017. doi:10.1007/s00037-016-0143-x.
- 51 Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive non-commutative rank computation is in deterministic polynomial time. *Comput. Complexity*, 27(4):561–593, 2018. doi:10.1007/s00037-018-0165-7.
- 52 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

- 53 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. URL: <http://www.jstor.org/stable/3689974>.
- 54 Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Comput.*, 8(4):499–507, 1979. doi:10.1137/0208040.
- 55 George Kempf and Linda Ness. The length of vectors in representation spaces. In Knud Lønsted, editor, *Algebraic Geometry*, pages 233–243, Berlin, Heidelberg, 1979. Springer Berlin Heidelberg.
- 56 Pascal Koiran. Hilbert's Nullstellensatz is in the Polynomial Hierarchy. *Journal of Complexity*, 12(4):273–286, 1996. doi:10.1006/jcom.1996.0019.
- 57 Joseph P. S. Kung and Gian-Carlo Rota. The invariant theory of binary forms. *Bull. Amer. Math. Soc. (N.S.)*, 10(1):27–85, 1984. doi:10.1090/S0273-0979-1984-15188-7.
- 58 Greg Kuperberg. Knottedness is in NP, modulo GRH. *Advances in Mathematics*, 256:493–506, 2014. doi:10.1016/j.aim.2014.01.007.
- 59 Serge Lang. *Elliptic Curves: Diophantine Analysis*, volume 231 of *Grundlehren der mathematischen Wissenschaften*. Springer, 2013.
- 60 Jonathan Leake and Nisheeth K. Vishnoi. On the computability of continuous maximum entropy distributions with applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, pages 930–943, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3357713.3384302.
- 61 Arjen K. Lenstra, Hendrik W. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, December 1982. doi:10.1007/BF01457454.
- 62 Seymour Lipschutz. *Schaum's outline of theory and problems of linear algebra : [including 600 solved problems; completely solved in detail]*. Schaum's outline series. McGraw-Hill, New York u.a., 6th ed. edition, 1974.
- 63 Visu Makam and Avi Wigderson. Singular tuples of matrices is not a null cone (and the symmetries of algebraic varieties). *J. Reine Angew. Math.*, 780:79–131, 2021. doi:10.1515/crelle-2021-0044.
- 64 Petros Maragos, Vasileios Charisopoulos, and Emmanouil Theodosis. Tropical geometry and machine learning. *Proceedings of the IEEE*, 109(5):728–755, 2021. doi:10.1109/JPROC.2021.3065238.
- 65 Jerrold Marsden and Alan Weinstein. Reduction of symplectic manifolds with symmetry. *Reports on Mathematical Physics*, 5(1):121–130, 1974. doi:10.1016/0034-4877(74)90021-4.
- 66 E. M. Matveev. An explicit lower bound for a homogeneous rational linear form in logarithms of algebraic numbers. *Izvestiya: Mathematics*, 62(4):723, August 1998. doi:10.1070/IM1998v062n04ABEH000190.
- 67 Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems : a cryptographic perspective*. The Kluwer international series in engineering and computer science BV000632170 671. Kluwer Academic, Boston, 2002.
- 68 Gary L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317, 1976. doi:10.1016/S0022-0000(76)80043-8.
- 69 Louis J. Mordell. On the representation of a binary quadratic form as a sum of squares of linear forms. *Mathematische Zeitschrift*, 35(1-15):1432–1823, 1932. doi:10.1007/BF01186544.
- 70 Ketan Mulmuley. Geometric complexity theory V: Efficient algorithms for Noether normalization. *J. Amer. Math. Soc.*, 30(1):225–309, 2017. doi:10.1090/jams/864.
- 71 Ketan Mulmuley and Milind Sohoni. Geometric complexity theory I: An approach to the P vs. NP and related problems. *SIAM Journal on Computing*, 31(2):496–526, 2001.
- 72 David Mumford. *The red book of varieties and schemes*, volume 1358 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1999. doi:10.1007/b62130.

- 73 David Mumford, John Fogarty, and Frances Kirwan. *Geometric invariant theory*, volume 34 of *Ergebnisse der Mathematik und ihrer Grenzgebiete (2) [Results in Mathematics and Related Areas (2)]*. Springer-Verlag, Berlin, third edition, 1994. doi:10.1007/978-3-642-57916-5.
- 74 Arkadi Nemirovski and Uriel Rothblum. On complexity of matrix scaling. *Linear Algebra and its Applications*, 302-303:435–460, 1999. doi:10.1016/S0024-3795(99)00212-8.
- 75 Joseph Oesterlé. Nouvelles approches du théorème de Fermat. In *Séminaire Bourbaki : volume 1987/88, exposés 686-699*, number 161-162 in *Astérisque*. Société mathématique de France, 1988. talk:694. URL: http://www.numdam.org/item/SB_1987-1988__30__165_0/.
- 76 Victor Y. Pan and Zhao Q. Chen. The complexity of the matrix eigenproblem. In *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*, pages 507–516. ACM, New York, 1999. doi:10.1145/301250.301389.
- 77 Michael O. Rabin and Jeffery O. Shallit. Randomized algorithms in number theory. *Communications on Pure and Applied Mathematics*, 39(S1):S239–S256, 1986. doi:10.1002/cpa.3160390713.
- 78 J. Maurice Rojas. Counting Real Roots in Polynomial-Time via Diophantine Approximation. *Foundations of Computational Mathematics*, 24(2):639–681, Apr 2024. doi:10.1007/s10208-022-09599-z.
- 79 Alexander Schrijver. *Theory of linear and integer programming*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, Chichester u.a., reprinted edition, 1999.
- 80 Mohit Singh and Nisheeth K. Vishnoi. Entropy, optimization and counting. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing, STOC '14*, pages 50–59, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2591796.2591803.
- 81 Cameron L. Stewart and R. Tijdeman. On the Oesterlé-Masser conjecture. *Monatshefte für Mathematik*, 102(3):251–257, September 1986. doi:10.1007/BF01294603.
- 82 Cameron L. Stewart and Kunrui Yu. On the abc conjecture. *Mathematische Annalen*, 291(2):225–230, 1991. URL: <http://eudml.org/doc/164860>.
- 83 Damian Straszak and Nisheeth K. Vishnoi. Maximum entropy distributions: Bit complexity and stability. In Alina Beygelzimer and Daniel Hsu 0001, editors, *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, volume 99 of *Proceedings of Machine Learning Research*, pages 2861–2891. PMLR, 2019. URL: <http://proceedings.mlr.press/v99/straszak19a.html>.
- 84 Bernd Sturmfels. *Algorithms in Invariant Theory*. Texts & Monographs in Symbolic Computation. Springer, 2008. doi:10.1007/978-3-211-77417-5.
- 85 Alfred J. van der Poorten. On Baker’s inequality for linear forms in logarithms. *Mathematical Proceedings of the Cambridge Philosophical Society*, 80(2):233–248, 1976. doi:10.1017/S0305004100052877.
- 86 Michel Waldschmidt. *Diophantine approximation on linear algebraic groups: Transcendence properties of the exponential function in several variables*. Grundlehren der mathematischen Wissenschaften BV000000395 326. Springer, Berlin u.a., 2000.
- 87 Michel Waldschmidt. Open diophantine problems. *Mosc. Math. J.*, 4:245–305, 2004.
- 88 Michel Waldschmidt. Lecture on the abc conjecture and some of its consequences. In Pierre Cartier, A.D.R. Choudary, and Michel Waldschmidt, editors, *Mathematics in the 21st Century*, pages 211–230, Basel, 2015. Springer Basel.
- 89 Gisbert Wüstholz and Alan Baker. Logarithmic forms and group varieties. *Journal für die reine und angewandte Mathematik*, 442:19–62, 1993. URL: <http://eudml.org/doc/153550>.

Quantum Automating TC^0 -Frege Is LWE -Hard

Noel Arteche  

Lund University, Sweden

University of Copenhagen, Denmark

Gaia Carenini  

École Normale Supérieure (ENS-PSL), Paris, France

University of Cambridge, UK

Matthew Gray  

University of Oxford, UK

Abstract

We prove the first hardness results against efficient proof search by quantum algorithms. We show that under Learning with Errors (LWE), the standard lattice-based cryptographic assumption, no quantum algorithm can weakly automate TC^0 -Frege. This extends the line of results of Krajíček and Pudlák (*Information and Computation*, 1998), Bonet, Pitassi, and Raz (*FOCS*, 1997), and Bonet, Domingo, Gavaldà, Maciel, and Pitassi (*Computational Complexity*, 2004), who showed that Extended Frege, TC^0 -Frege and AC^0 -Frege, respectively, cannot be weakly automated by classical algorithms if either the RSA cryptosystem or the Diffie-Hellman key exchange protocol are secure. To the best of our knowledge, this is the first interaction between quantum computation and propositional proof search.

2012 ACM Subject Classification Theory of computation \rightarrow Proof complexity; Theory of computation \rightarrow Quantum complexity theory

Keywords and phrases automatability, post-quantum cryptography, feasible interpolation

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.15

Related Version *Full Version*: <https://eccc.weizmann.ac.il/report/2024/029/> [7]

Funding *Noel Arteche*: This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Acknowledgements The question of the quantum non-automatability of strong proof systems was suggested to us by three different people. We thank Vijay Ganesh for bringing it up during the Dagstuhl Seminar 22411 *Theory and Practice of SAT and Combinatorial Solving*. We thank Susanna F. de Rezende for bringing our attention to the problem later and for insightful comments and careful proofreading. We would also like to thank Ján Pich for pointing us to the problem and discussing many details with us. We are particularly grateful for him directing us to the work of Soltys and Cook on LA. We also thank Rahul Santhanam for his insights and conversations and Yanyi Liu for pointing us to the existence of the certificates of injectivity. We also thank María Luisa Bonet, Jonas Conneryd, Ronald de Wolf, Eli Goldin, Peter Hall, Russell Impagliazzo, Erfan Khaniki, Alex Lombardi, Daniele Micciancio, Angelos Pelecanos and Michael Soltys for useful comments, suggestions and pointers. A preliminary version of this work was presented at the poster session of QIP 2024. We are thankful to the anonymous reviewers and their suggestions. We are also grateful to the anonymous CCC reviewers for their comments and particularly for observing that some crucial axioms were missing from the definition of $\text{LA}_{\mathbb{Q}}$ in an earlier version of this work.

This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing at UC Berkeley during the spring of 2023 for the *Meta-Complexity* and *Extended Reunion: Satisfiability* programs.



© Noel Arteche, Gaia Carenini, and Matthew Gray;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 15; pp. 15:1–15:25
Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



1 Introduction

Originally, propositional proof complexity has been primarily concerned with proving lower bounds for the length of proofs in propositional proof systems, with the ultimate goal of settling whether $\text{NP} = \text{coNP}$ [21]. In parallel, a growing line of research has focused on the computational hardness of finding propositional proofs. Efficient proof search is formally captured by the notion of *automatability*, introduced by Bonet, Pitassi, and Raz [17]: a propositional proof system S is automatable if there exists an algorithm that given as input a tautology φ , outputs an S -proof of φ in time polynomial in the size of the shortest proof. By relating proofs and computation, automatability connects proof complexity to central areas of theoretical computer science such as automated theorem proving, SAT solving and combinatorial optimization [18], learning theory [44], and Kolmogorov complexity [36].

Except for very weak proof systems like Tree-like Resolution, automatable in quasipolynomial time [12], most natural systems appear to be non-automatable under standard hardness assumptions. Existing hardness results can be split into two broad categories. Work from the late 90s and early 00s showed that stronger proof systems are non-automatable under cryptographic assumptions, while more recent work has shown that weaker proof systems are non-automatable under the optimal assumption that $\text{P} \neq \text{NP}$.

The cryptography-based approach was initiated by the seminal work of Krajíček and Pudlák [37], who showed that Extended Frege is not automatable unless factoring can be solved efficiently, although the notion of automatability would only be defined slightly later by Bonet, Pitassi, and Raz [17], who showed that TC^0 -Frege is hard to automate unless Blum integers can be factored by polynomial-size circuits. Finally, Bonet, Domingo, Gavaldà, Maciel, and Pitassi [16] extended the existing result from TC^0 -Frege to AC^0 -Frege under the stronger assumption that Blum integers cannot be factored by subexponential-size circuits.

Building on a long line of work [19, 30, 46, 8, 4, 5, 38], the first NP -hardness result was shown in 2019, when Atserias and Müller [10] proved that Resolution is not automatable unless $\text{P} = \text{NP}$. This is optimal, as $\text{P} = \text{NP}$ implies the automatability of any proof system. Their proof uses a clever reduction from SAT that requires showing a specific lower bound for this system. The technique has since been adapted to other weak proof systems such as Regular and Ordered Resolution [14], k -DNF Resolution [24], Cutting Planes [25], Nullstellensatz and Polynomial Calculus [23], the OBDD proof system [29] and, more recently, even AC^0 -Frege [41].

Though the latter works prove non-automatability under the optimal hardness assumption, their strength is incomparable to the cryptography-based results. The NP -hardness results all rely on proving specific super-polynomial proof complexity lower bounds for each system, meaning this strategy fails for $\text{AC}^0[2]$ -Frege and systems above, for which no lower bounds are known. In contrast, the cryptographic hardness results work by ruling out *feasible interpolation* for these systems, a property which allows one to extract computational content from proofs. For a proof system S proving its own soundness (such as TC^0 -Frege), feasible interpolation is equivalent to the notion of *weak automatability* introduced by Atserias and Bonet [8], the latter meaning that no proof system simulating S is automatable. The question of whether weak systems such as Resolution are weakly automatable remains one of the major open problems in the field. In short, there exists a trade-off between the strength of the hardness assumption involved ($\text{P} \neq \text{NP}$ versus cryptographic) and the generality of the result (automatability versus weak automatability).

Our work is the first new contribution to the non-automatability of strong proof systems¹ in more than two decades. The early results [37, 17, 16] relied on the assumption that factoring is hard, which does not hold for quantum models of computation due to Shor's breakthrough algorithm [49]. This raises the question of whether a quantum machine could carry out proof search efficiently for some strong proof system. Grover's search algorithm [27] already provides a quadratic speed-up over brute-force proof search for any system. While this is not enough to achieve automatability, the possibility of more powerful algorithms motivates the interest in new conditional hardness results. The **NP**-hardness results outlined above imply that $\mathbf{NP} \not\subseteq \mathbf{BQP}$ suffices to rule out automatability for weak systems, but for stronger systems no widely believed assumption had yet been proven sufficient.

In this work, we formally define the notion of *quantum automatability* and show the first hardness results. We prove that \mathbf{TC}^0 -Frege is not quantum automatable unless lattice-based cryptography can be broken by polynomial-size quantum circuits. Our results follow from the relationship between automatability and feasible interpolation suitably generalized to the quantum setting. This means that we also rule out quantum feasible interpolation and weak quantum automatability under the same cryptographic assumptions.

Contributions

Our main contribution is proving the hardness of quantum automatability under the assumption that lattice-based cryptography is secure against quantum computers.

In 1996, Ajtai [2] gave the first worst-case to average-case reductions for lattice problems. In 1997, in joint work with Dwork [3], the worst-case hardness of such lattice problems was used to design public-key cryptosystems. Building on similar principles, the Learning with Errors (LWE) assumption of Regev [48] has become the standard post-quantum cryptographic assumption. The LWE assumption is simple to state, surprisingly versatile, and does not seem susceptible to the period-finding technique crucial to Shor's algorithm.

In this work we show that any quantum algorithm that automates \mathbf{TC}^0 -Frege can be used to break LWE.

► **Theorem** (Main theorem, informal). *If there exists a polynomial-time quantum algorithm that weakly automates \mathbf{TC}^0 -Frege, then LWE can be broken in polynomial time by a quantum machine.*

We then exploit the simulation of \mathbf{TC}^0 -Frege by \mathbf{AC}^0 -Frege proofs of subexponential size to extend the result to \mathbf{AC}^0 -Frege under a slightly stronger assumption, in the style of [16].

► **Corollary**. *If there exists a polynomial-time quantum algorithm that weakly automates \mathbf{AC}^0 -Frege, then LWE can be broken in subexponential time by a quantum machine.*

In order to properly state and prove these results, we first formally define the notion of quantum automatability for quantum Turing machines. Note that a quantum algorithm might provide a wrong answer with small probability, so we need to be careful in choosing the right definitions. We show that our definition is equivalent to a similar one over uniform quantum circuits, and we verify that it is robust by reproving Impagliazzo's observation that weak automatability implies feasible interpolation, suitably translated to the quantum setting.

¹ We use the terms *weak* and *strong* informally throughout the paper. Traditionally, a strong proof system is a system that proves its own soundness, though it is often also intended to be a system for which lower bounds are lacking. For our purposes, *strong* refers to anything simulating \mathbf{TC}^0 -Frege, for which both of the previous conditions apply.

Techniques

The overall structure of the proofs follows the strategy of the previous non-automatability results of Krajíček and Pudlák [37] and Bonet, Pitassi, and Raz [17], but the technical details are quite different due to certain complications arising from lattice-based cryptography. We outline below the main hurdles and the techniques used to overcome them.

Quantum feasible interpolation

Our result follows from conditionally ruling out feasible interpolation by quantum circuits. As observed by Impagliazzo, weak automatability implies feasible interpolation. We use this observation contrapositively. Suppose that a proof system can prove the injectivity of a candidate one-way function. In the presence of feasible interpolation, we are guaranteed the existence of small circuits capable of inverting the one-way function one bit at a time. If one believes in the security of the cryptographic object, one must conclude that the proof system does not admit feasible interpolation, and in turn that it is not weakly automatable.

For this strategy to work the candidate one-way function should fulfill two important conditions. First, its definition must be simple enough that the proof system can easily reason about it. For example, RSA requires modular exponentiation to be defined, which is conjectured not to be computable in \mathbf{TC}^0 . This forced Bonet, Pitassi, and Raz to use instead the Diffie-Hellman protocol. Second, the candidate one-way function must be injective. The rather technical reason for injectivity is that feasible interpolation allows one to carry out the inversion bit by bit, which does not guarantee retrieving a correct preimage if there are multiple ones.

A few injective one-way functions based on lattice geometry have been proposed throughout the literature, e.g., see [43, 26, 40]. However, we consider instead a simple scheme for worst-case lattice-based functions that closely resembles the one described by Micciancio [39]. Such a scheme has the advantage that its injectivity can be easily verified, and that its worst case one way-ness is guaranteed by the assumed hardness of Learning with Errors, which we will now discuss.

Learning with Errors and certificates of injectivity

We base our construction directly on the Learning with Errors assumption. The assumption is simple to define: roughly speaking, a vector x should be hard to recover after being multiplied by some public matrix A , and summed with some Gaussian noise, $Ax + \varepsilon$. While the most naive functions based on LWE are not necessarily injective, we can bound the magnitude of the error vectors to construct a family of functions where almost all of the functions are injective. For most matrices A , the corresponding function f_A in this family is worst-case one-way assuming the hardness of LWE [39].

However, the functions being injective and worst-case one-way is not sufficient, because their injectivity needs to be provable inside \mathbf{TC}^0 -Frege. Unlike with the Diffie-Hellman construction, where a single proof showed the injectivity of the protocol for all generators, here each injective f_A may require a tailored proof of injectivity. Fortunately, most of these f_A can have their injectivity certified by a left-inverse of A together with a short basis for the dual lattice of the q -ary lattice spanned by A . These short bases not only certify injectivity, but can also be used as trapdoors to invert the function [42]. Though we do not exploit this directly, one may think of the automating algorithm as extracting such trapdoors from proofs. Instead, we use these certificates to prove the injectivity of most f_A inside \mathbf{TC}^0 -Frege.

With these properties, we can show that feasible interpolation can be used to invert almost all f_A , which is sufficient to break LWE and its associated worst-case lattice problems.

Formal theories for linear algebra

The most technical component of the previous work on \mathbf{TC}^0 -Frege and \mathbf{AC}^0 -Frege was the formalization of many basic properties of arithmetic directly inside the propositional proof systems, which can be quite cumbersome. While we can borrow a large part of the existing formalization of Bonet, Pitassi, and Raz [17], putting it together to carry out arguments about lattice geometry would still be quite convoluted.

Instead, we follow the approach of Krajíček and Pudlák, who showed the injectivity of RSA in Extended Frege by reasoning in Buss's theory S_2^1 of bounded arithmetic. Universal theorems of this first-order theory translate into propositional tautologies with succinct proofs in Extended Frege. For \mathbf{TC}^0 -Frege and its sequent calculus formalism PTK, the corresponding first-order theory of bounded arithmetic is the two-sorted theory \mathbf{VTC}^0 introduced by Cook and Nguyen [20]. This theory is quite expressive and can reason even about analytic functions, as shown by Jeřábek [32]. However, since we are mostly interested in statements of matrix algebra, we use the more convenient formal theory LA for linear algebra introduced by Soltys and Cook [50].

The theory LA is quantifier-free and operates directly with matrices. It is strong enough to prove their ring properties, but weak enough to allow all theorems in LA to translate into propositional tautologies with short \mathbf{TC}^0 -Frege proofs. In order to handle all the concepts required in our arguments, we work over a conservative extension of LA over the rationals which we show still propositionally translates into \mathbf{TC}^0 -Frege.

Open problems

To the best of our knowledge, this is the first interaction between quantum computation and propositional proof search, and we believe further exploration of connections between the two fields is worthwhile. We outline below three open lines of research, ranging from the interaction between quantum computation and proof complexity to a classical problem in the theory of automatability.

Positive results?

While hardness of proof search in most natural proof systems is now conditionally ruled out under different assumptions, there exists a handful of systems for which no non-automatability results are known. This is the case for the $\text{Res}(\oplus)$, $\text{Res}(\log)$, Sherali-Adams and Sum-of-Squares proof systems. Could quantum algorithms automate any of these systems efficiently?

Even for proof systems where worst-case hardness is known, could quantum algorithms provide a significant speed-up over brute-force search? Clearly, Grover's algorithm already achieves a quadratic speed-up, but could this be pushed further in some cases?

Quantum proof complexity

Hardness results in automatability involve three key elements: the proof system, the hardness assumption and the model of computation for the automating algorithm. In this work we shifted the latter two to the quantum setting, by choosing a post-quantum cryptographic assumption and a quantum model of computation, but the proof systems considered remain classical.

What would it mean to have an inherently quantum proof system? In the same way that Extended Frege can be seen as \mathbf{P} /poly-Frege, could we define a proof system where lines are quantum circuits? This could open the door to a quantum analogue of the Cook-Reckhow

program, where showing lower bounds on quantum proof systems would be related to the question of whether $\text{QMA} = \text{coQMA}$. We note that an analogous approach exists in the field of parameterized complexity, starting with the work of Dantchev, Martin, and Szeider [22], who defined parameterized proof complexity as a program to gain evidence on the \mathbf{W} -hierarchy being different from \mathbf{FPT} . As an intermediate step, it would make sense to consider the case of randomized proof systems and the relationship between \mathbf{MA} and coMA , though this has proven to be challenging so far.

We remark that while Pudlák [45] already defined the notion of quantum derivation rules for propositional proof systems and defined the quantum Frege proof system, his approach is orthogonal to ours, in that those systems are still designed to derive propositional tautologies. In fact, he showed that classical Frege systems simulate quantum Frege systems, though classical Frege proofs cannot be extracted from quantum proofs by a classical algorithm unless factoring is in \mathbf{FP} .

Towards generic hardness assumptions

Like the original works on weak automatability, our proof requires *concrete* cryptographic assumptions. That is, we assume that some specific candidate one-way function or cryptographic protocol is secure. The reason is that in order to obtain the upper bounds on which to apply feasible interpolation we need concrete formulas to manipulate inside the different proof systems.

A major open problem in the theory of automatability is to disentangle these results from concrete families of candidate one-way functions. That is, can we prove that TC^0 -Frege is not (weakly) automatable under the assumption that, say, one-way functions exist? Even better, can one obtain \mathbf{NP} -hardness of automating strong proof systems without the need to prove lower bounds first, in a way different from the strategy of Atserias and Müller [10]? This seems to require conceptual breakthroughs.

Structure of the paper

The paper is structured as follows. Section 2 recalls the necessary concepts in proof complexity and lattice-based cryptography needed in the rest of the paper. Section 3 defines automatability for quantum Turing machines and uniform quantum circuits and proves the equivalence between both models to then reprove Impagliazzo’s observation on the relation between automatability and feasible interpolation, now in the quantum setting. Section 4 states and proves the main theorem of the paper. The section first presents a detailed overview of the main argument, while the subsections contain all the necessary technical work.

2 Preliminaries

We assume basic familiarity with computational complexity theory, propositional logic and quantum circuits. We review the main concepts needed from proof complexity and refer the reader to standard texts like [35] for further details. We also recall some relevant notions from linear algebra and lattice geometry useful in our arguments.

2.1 Proof complexity

Following Cook and Reckhow [21], a *propositional proof system* S for the language TAUT of propositional tautologies is a polynomial-time surjective function $S : \{0, 1\}^* \rightarrow \text{TAUT}$. We think of S as a proof checker that takes some proof $\pi \in \{0, 1\}^*$ and outputs $S(\pi) = \varphi$, the

theorem that π proves. Soundness follows from the fact that the range is exactly TAUT, and implicational completeness is guaranteed by the fact that S is surjective. One may alternatively define proof systems for refuting propositional contradictions. We move from one setup to the other depending on context.

We denote by $\text{size}_S(\varphi)$ the *size* of the smallest S -proof of φ plus the size of φ . We say that a proof system S is *polynomially bounded* if for every $\varphi \in \text{TAUT}$, $\text{size}_S(\varphi) \leq |\varphi|^{O(1)}$. We say that a proof system S *polynomially simulates* a system Q if for every $\varphi \in \text{TAUT}$, $\text{size}_S(\varphi) \leq \text{size}_Q(\varphi)^{O(1)}$. For a family $\{\varphi_n\}_{n \in \mathbb{N}}$ of propositional tautologies, we write $S \vdash \varphi_n$ whenever $\text{size}_S(\varphi_n) \leq |\varphi_n|^{O(1)}$. Finally, a proof system S is said to be *closed under restrictions* if whenever S proves a formula φ in size s , for every partial restriction ρ to the variables in φ , there exists a proof of the restricted formula $\varphi|_\rho$ in size $s^{O(1)}$.

The focus of this work is on a specific class of proof systems known as *Frege systems*. A Frege system is a finite set of axiom schemas and inference rules that are sound and implicationally complete for the language of propositional tautologies built from the Boolean connectives negation (\neg), conjunction (\wedge), and disjunction (\vee). A Frege proof is a sequence of formulas where each formula is obtained by either substitution of an axiom schema or by application of an inference rule on previously derived formulas. As long as the set of inference rules is finite, sound and implicationally complete, the specific choice of rules does not effect the size of the proofs up to polynomial factors, as all Frege systems polynomially simulate each other [35, Theorem 4.4.13].

We can make gradations between Frege systems by restricting the complexity of their proof lines. For a circuit class \mathcal{C} , the system \mathcal{C} -Frege is any Frege system where lines are restricted to be \mathcal{C} -circuits (see [31] for a formal definition). In this setup, a standard Frege system amounts to \mathbf{NC}^1 -Frege. We are mostly interested in the weaker systems \mathbf{AC}^0 -Frege and \mathbf{TC}^0 -Frege, where the proof lines are, respectively, circuits of constant-depth and unbounded fan-in, and threshold circuits of constant-depth and unbounded fan-in. A *threshold circuit* is a Boolean circuit where gates can be the usual \neg, \vee, \wedge as well as the threshold ones $\text{Th}_k(x_1, \dots, x_n)$, where Th_k is true if at least k of its inputs are true.

It is often convenient to consider an alternative formalism of \mathbf{TC}^0 -Frege in the style of Gentzen's sequent calculus. The *Propositional Threshold Calculus* PTK [20, Chapter X.4.1] is a version of the propositional sequent calculus where the cuts are restricted to threshold formulas of constant depth. We refer to [17, Section 2] for a complete rendering of the derivational rules of PTK.

2.2 Lattice geometry

We recall some basic definitions from lattice geometry. For a linearly independent set of n vectors $\mathcal{B} = \{b_1, \dots, b_n\} \subseteq \mathbb{R}^m$, which we often treat simply as an $m \times n$ matrix, the *lattice* over \mathcal{B} is defined to be the set of all integer linear combinations of vectors in \mathcal{B} ,

$$\mathcal{L}(\mathcal{B}) := \{x \in \mathbb{R}^m \mid \text{there is } a \in \mathbb{Z}^n \text{ such that } x = \mathcal{B}a\}.$$

When the vectors in \mathcal{B} belong in \mathbb{Z}_q^m for some modulus q , we can further define a *modular lattice* over \mathcal{B} , denoted $\mathcal{L}_q(\mathcal{B})$, to be the set of all integer linear combinations of the basis modulo q ,

$$\mathcal{L}_q(\mathcal{B}) := \{x \in \mathbb{Z}_q^m \mid \text{there is } a \in \mathbb{Z}^n \text{ such that } \mathcal{B}a \equiv x \pmod{q}\},$$

where the mod function is applied element-wise in the vector.

15:8 Quantum Automating TC^0 -Frege Is LWE -Hard

We define the length of a vector x in $\mathcal{L}_q(\mathcal{B})$ to be the Euclidean norm of the shortest vector in \mathbb{Z}^m that is congruent to x modulo q . Note that these shortest vectors will always fall in the domain $[-\lfloor q/2 \rfloor, \lfloor q/2 \rfloor]^m$.

A q -ary lattice $\Delta_q(\mathcal{B})$ can be thought of as an extension of a modular lattice back to \mathbb{Z}^m and is the set of all vectors $x \in \mathbb{Z}^m$ congruent to members of the modular lattice,

$$\Delta_q(\mathcal{B}) := \{x \in \mathbb{Z}^m \mid \text{there is } a \in \mathbb{Z}^n \text{ such that } \mathcal{B}a \equiv x \pmod{q}\}.$$

Note that because for all $x \in \{0, q\}^m$, $x \in \Delta_q(\mathcal{B})$, we have that all q -ary lattices have rank m .

From the definitions above it is clear that $\mathcal{L}_q(\mathcal{B}) \subseteq \Delta_q(\mathcal{B})$. Consequently a proof that no vector in $\Delta_q(\mathcal{B})$ has length less than ℓ also proves that no vector in $\mathcal{L}(\mathcal{B})$ has length less than ℓ .

Another important concept in lattice geometry is that of a *dual lattice*. Given a lattice $\mathcal{L}(\mathcal{B})$, its dual lattice $\mathcal{L}^*(\mathcal{B})$ is defined to be the set of vectors within the subspace spanned by \mathcal{B} whose inner product with any element in \mathcal{L} is an integer. Formally,

$$\mathcal{L}^*(\mathcal{B}) := \{y \in \mathbb{R}^m \mid \text{there is } z \in \mathbb{R}^n \text{ such that } y = \mathcal{B}z \text{ and for all } x \in \mathcal{L}(\mathcal{B}), \langle x, y \rangle \in \mathbb{Z}\},$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. The dual lattice is also a lattice, whose basis admits a closed form.

► **Lemma 1.** *For a basis $\mathcal{B} \in \mathbb{R}^{m \times n}$, $\mathcal{L}^*(\mathcal{B}) = \mathcal{L}(\mathcal{B}(\mathcal{B}^\top \mathcal{B})^{-1})$.*

Note that, if $\mathcal{B} \in \mathbb{Z}^{m \times n}$, it is easy to show that $\mathcal{B}(\mathcal{B}^\top \mathcal{B})^{-1} \in \mathbb{Q}^{m \times n}$, and, therefore, that any $x \in \mathcal{L}^*(\mathcal{B})$ belongs to \mathbb{Q}^m . This lemma is standard and can be found, for example, in [39].

Modular lattices and q -ary lattices are fairly different mathematical objects, but we can show that given a matrix $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$ such that $\text{rank}(\mathcal{B}) = n$, there exists a closed form for a matrix \mathcal{B}' such that $\mathcal{L}(\mathcal{B}') = \Delta_q(\mathcal{B})$.

► **Lemma 2 (Full-rank modular lattices have q -ary lattice bases).** *Let $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$ and define $C \in \{0, 1\}^{m \times m}$ to be the permutation matrix that swaps the appropriate rows so that the first n rows of $C\mathcal{B}$ are linearly independent. Let $\mathcal{B}_1 \in \mathbb{Z}^{n \times n}$ and $\mathcal{B}_2 \in \mathbb{Z}^{m-n \times n}$ be matrices such that $\mathcal{B} = [C\mathcal{B}_1 \mid C\mathcal{B}_2]^\top$. Then, for $\mathcal{B} \in \mathbb{Z}_q^{m \times n}$, if $\text{rank}(\mathcal{B}) = n$, $\Delta_q(\mathcal{B}) = \mathcal{L}(\mathcal{B}')$, where*

$$\mathcal{B}' = C \begin{bmatrix} I_n & 0 \\ (C\mathcal{B}_2)(C\mathcal{B}_1)^{-1} & qI_{m-n} \end{bmatrix} C^{-1},$$

and where the inverses M^{-1} are defined over the modular lattice \mathbb{Z}_q^m .

Note that we can combine this corollary with Lemma 1 to get a closed form for \mathcal{B}' such that $\Delta_q^*(\mathcal{B}) = \mathcal{L}(\mathcal{B}')$.

The i -th successive minimum of a lattice \mathcal{L} is $\lambda_i(\mathcal{L}) := \inf\{r \in \mathbb{Z} \mid \dim(\text{span}(\mathcal{L} \cap B(0, r))) \geq i\}$, where $B(0, r)$ is the ball of radius r around the origin. Roughly speaking, this means that $\lambda_i(\mathcal{L})$ is the length of the i -th smallest linearly independent vector in the lattice.

There exists an intimate relationship between a lattice and its dual, as captured by Banaszczyk's Transference Theorem.

► **Theorem 3 (Transference Theorem [11]).** *For any rank- n lattice $\mathcal{L} \subseteq \mathbb{Z}^m$, it holds that*

$$1 \leq \lambda_1(\mathcal{L}) \cdot \lambda_n(\mathcal{L}^*) \leq n.$$

Modular lattices $\mathcal{L}_q(\mathcal{B})$ are subsets of \mathbb{Z}_q^m , not \mathbb{Z}^m , and therefore the Transference Theorem does not directly apply. However we are able to leverage the fact that $\lambda_1(\Delta_q(\mathcal{B})) = \min(q, \lambda_1(\mathcal{L}_q(\mathcal{B})))$ to indirectly apply it through the q -ary lattice.

We recall useful properties of random lattices.

► **Lemma 4.** *For a randomly selected matrix $A \in \mathbb{Z}_q^{m \times n}$, we have that*

- (i) $\Pr_A[\text{rank}(A) = n] \geq n/q^{m-n+1}$;
- (ii) $\Pr_A[\lambda_1(\mathcal{L}_q(A)) < r \mid \text{rank}(A) = n] \leq (2r + 1)^m / q^{m-2n-1}$.

These properties are folklore. For the sake of completeness, we provide proofs in Appendix C of the full version of the paper [7].

2.3 Learning with Errors (LWE)

Learning with Errors (LWE) is a central problem of learning theory, introduced by Regev [48].

► **Assumption 5** (The Learning with Errors (LWE) assumption [48, 42]). *Let $m = n^{O(1)}$, $q \leq 2^{n^{O(1)}}$, let $s \sim \mathbb{Z}_q^n$ be a secret vector, $A \sim \mathbb{Z}_q^{m \times n}$, and $\varepsilon \in \mathbb{Z}_q^m$ a sample from the discrete Gaussian with standard deviation αq with $\alpha = o(1)$ and $\alpha \in [0, 1]$. The Learning with Errors assumption states that there is no quantum inverter M running in time $n^{O(1)}$ such that $M(A, As + \varepsilon)$ outputs s with noticeable probability over the choice of s, A, ε , and the internal randomness of M .*

The security of this assumption relies on the existence of worst-case to average-case reductions to fundamental lattice problems conjectured to be hard. In particular, as shown by Regev [48], breaking LWE implies solving the γ -GAPSVP problem for an approximation factor $\gamma = n^2$. Here, γ -GAPSVP refers to the γ -Approximate Shortest Vector Problem: given a lattice basis $\mathcal{B} \in \mathbb{Q}^{m \times n}$ and a distance threshold $r > 0$, decide whether $\lambda_1(\mathcal{L}(\mathcal{B})) \leq r$, or $\lambda_1(\mathcal{L}(\mathcal{B})) > \gamma r$, when one of those cases is promised to hold.

The belief that γ -GAPSVP is intractable is backed by the fact that the problem is **NP**-hard under randomized reductions when the approximation factor is constant [2, 42, 15]. However, for the range of γ in which the reduction to LWE works, no **NP**-hardness is known. Obtaining **NP**-hardness for polynomial approximation factors would imply the breakthrough consequence of basing cryptography on worst-case hardness assumptions. In turn, this would turn our non-automatability results into **NP**-hardness results. As appealing as this might be, it is unlikely. For $\gamma \geq \sqrt{n}$, the problem γ -GAPSVP is known to be in **NP** \cap **coNP** [1] and thus cannot be **NP**-hard unless **PH** collapses.

2.4 The formal theory LA

The theory LA is a quantifier-free theory introduced by Soltys and Cook [50] whose main objects are matrices. This is not technically speaking a first-order theory of bounded arithmetic like those used by Krajíček and Pudlák [37], but like them it admits a propositional translation into Frege systems.

The system LA operates over three sorts: *indices* (intended to be natural numbers), *field elements* (over some abstract field \mathbb{F}), and *matrices* (with entries over \mathbb{F}). Variables for these three sorts are usually denoted i, j, k, \dots for indices, a, b, c, \dots for field elements, and A, B, C, \dots for matrices. We sometimes use lower-case letters v, w, \dots for vectors, which are seen as a special case of matrices.

The language of LA consists of the following constant, predicate and function symbols, over the three different sorts:

15:10 Quantum Automating TC^0 -Frege Is LWE -Hard

- Index sort: $0_{\text{index}}, 1_{\text{index}}, +_{\text{index}}, \cdot_{\text{index}}, -_{\text{index}}, \text{div}, \text{rem}, \text{cond}_{\text{index}}, \leq_{\text{index}}, =_{\text{index}}$
- Field sort: $0_{\text{field}}, 1_{\text{field}}, +_{\text{field}}, \cdot_{\text{field}}, -_{\text{field}}, ^{-1}, r, c, e, \Sigma, \text{cond}_{\text{field}}, =_{\text{field}}$
- Matrix sort: $=_{\text{matrix}}$

The meaning of the symbols is the standard one, except for $-_{\text{index}}$ that denotes the cutoff subtraction ($i - j = 0$ if $i < j$) and for a^{-1} , denoting the inverse of a field element a , with $0^{-1} = 0$. For operations over matrices, $r(A)$ and $c(A)$ are, respectively, the number of rows and columns in A , $e(A, i, j)$ is the field element $A_{i,j}$ (with $e(A, i, j) = 0$ if either $i = 0$, $j = 0$, $i > r(A)$ or $j > c(A)$) and ΣA is the sum of the elements in A . The function symbol $\text{cond}(\alpha, t_1, t_2)$ is interpreted to mean that if α holds, then the returned value should be t_1 , else t_2 , where α is a formula all of whose atomic subformulas have the form $m \leq n$ or $m = n$, where m and n are of the index sort, and t_1, t_2 are terms either both of index sort or both of field sort.

The language of LA can be enriched with the following defined terms: index maximum (max), matrix sum ($+$, when sizes of the matrices are compatible), scalar product (\cdot), matrix transpose (AT), zero (0) and identity matrices (I), matrix trace (tr), dot product ($\langle _, _ \rangle$), and matrix product (\cdot). See [50, Section 2.1] for details on the definitions of these terms. In general, whenever it is clear from context, we drop the subscripts indicating the sort and we use standard linear algebra notation for the sake of readability.

The theory then consists of several groups of axioms fixing the meaning of these symbols. These are rather lengthy to state, so we relegate them to Appendix B, where we also include several theorems derived by Soltys and Cook inside LA.

Observe that the theory is field-independent, but whenever we fix the field to be either finite or \mathbb{Q} , LA has the robust property that every theorem translates into a family of propositional formulas with short TC^0 -Frege proofs. This is the main property of LA that we shall exploit.

3 Quantum automatability and feasible interpolation

Following Bonet, Pitassi, and Raz [17], we say that a propositional proof system S is *automatable in time t* if there exists a deterministic Turing machine A that on input a formula φ outputs an S -proof of φ , if one exists, in time $t(\text{size}_S(\varphi))$. We now consider the possibility of replacing A by a probabilistic or quantum Turing machine. The main issue in the definition is now that the output of the machine may be erroneous, albeit with small probability. Note, however, that if a machine were to output an incorrect proof, we would be able to easily detect this, since we can verify the proofs in polynomial time. We may thus assume that when yielding an incorrect proof, the machine will restart and find another one. Hence, instead of asking for the error-probability of the machine to be bounded, we ask for the expected running time to be bounded. The following definition captures this idea.

► **Definition 6** (Quantum and randomized automatability). *Let S be a propositional proof system and let $t : \mathbb{N} \rightarrow \mathbb{N}$ be a time-constructible function. We say that S is quantum (respectively, random) automatable in time t or simply quantumatable in time t if there exists a quantum Turing machine (respectively, a randomized Turing machine) that on input a formula φ outputs an S -proof of φ , if one exists, in expected time $t(\text{size}_S(\varphi))$.*

In what follows, we assume t to be a polynomial and talk simply about a system being *automatable* or *quantum automatable*, without reference to t . Since quantum circuits are often more convenient than quantum Turing machines, we also define automatability in the circuit setting.

► **Definition 7** (Circuit automatability). *Let S be a propositional proof system. We say that S is circuit-automatable if there exists a constant c and a uniform multi-output circuit family $\{C_{n,s}\}_{n,s \in \mathbb{N}}$ of size $(n+s)^c$ such that $C_{n,s}$ takes as input a formula φ of size n and outputs an S -proof of size s^c if a proof of size s exists, and is allowed to output any string otherwise.*

The generalization to randomized and quantum circuits is now immediate.

► **Definition 8.** *Let S be a propositional proof system. We say S is quantum circuit-automatable if there exists a constant c and a uniform multi-output quantum circuit family $\{C_{n,s}\}_{n,s \in \mathbb{N}}$ of size $(n+s)^c$ such that $C_{n,s}$ takes as input a formula φ of size n , and outputs an S -proof of size s^c with probability at least $2/3$ if a proof of size s exists, and is allowed to output any string otherwise. We say that S is random circuit-automatable if the circuit is classical but also takes as input a sequence r of random bits and, for at least $2/3s$ of the choices for r , $C_{n,s}(\varphi, r)$ outputs an S -proof of size s^c if a proof of size s exists, and is allowed to output any string otherwise.*

In fact, the machine-based and circuit-based definitions are equivalent.

► **Proposition 9.** *Let S be a propositional proof system. The following equivalences hold:*

- (i) *the system S is automatable if and only if it is circuit-automatable;*
- (ii) *the system S is random automatable if and only if it is random circuit-automatable;*
- (iii) *the system S is quantum automatable if and only if it is quantum circuit-automatable.*

We defer the rather simple proof to Appendix B in the full version of the paper [7].

Even if a proof system is not automatable, one might still hope for an algorithm that finds some proof efficiently, even if it is in a different proof system. We say that a proof system S is *weakly automatable* if there exists another proof system Q and an algorithm A that given a formula φ , outputs a Q -proof of φ in time $\text{size}_S(\varphi)^{O(1)}$. The concept was introduced by Atserias and Bonet [8], who further showed that this is equivalent to S being simulated by a system Q that is itself automatable. Despite the fact that weak automatability has been conditionally ruled out for Resolution under hardness assumptions for certain two-player games [9, 28, 13], establishing whether weak proof systems – such as Resolution – are weakly automatable under more standard hardness conjectures remains one of the main open problems in the area. It is straightforward to extend the notion of weak automatability to the quantum setting.

Weak automatability is closely related to feasible interpolation. We recall this connection in its classical form and then move to the quantum setting.

► **Definition 10** (Feasible interpolation [34, 46]). *We say that a proof system S has the feasible interpolation property if there exists a polynomial-time computable function I such that for every tautological split formula $\varphi(x, y, z) = \alpha(x, z) \vee \beta(z, y)$, whenever a proof π in S derives φ in size s , $I(\pi)$ produces an interpolant circuit C_φ of size $s^{O(1)}$ that takes as input an assignment ρ to the z -variables and such that*

$$C_\varphi(\rho) = \begin{cases} 0 & \text{only if } \alpha(x, \rho) \text{ is a tautology} \\ 1 & \text{only if } \beta(\rho, y) \text{ is a tautology} \end{cases}$$

indicating which side of the conjunction is tautological.

Bonet, Pitassi, and Raz attribute the following crucial observation relating (weak) automatability and feasible interpolation to Impagliazzo. We refer to it as *Impagliazzo's observation*.

15:12 Quantum Automating TC^0 -Frege Is LWE -Hard

► **Proposition 11** (Impagliazzo’s observation [17, Thm. 1.1]). *If a proof system is weakly automatable and closed under restrictions, then it admits feasible interpolation.*

Impagliazzo’s observation is useful contrapositively: to rule out (weak) automatability it suffices to rule out feasible interpolation, as done in the previous works [37, 17]. We outline this strategy further in Section 4, where we instantiate it together with our cryptographic assumption.

To use feasible interpolation in our setting, we suitably adapt the definition to the quantum world.

► **Definition 12** (Quantum feasible interpolation). *We say that a proof system S has the quantum feasible interpolation property if there exists a polynomial-time computable function I such that, for every tautological split formula $\varphi(x, y, z) = \alpha(x, z) \vee \beta(z, y)$, whenever a proof π derives φ in S in size s , $I(\pi)$ prints the description of a quantum interpolant circuit C_φ of size $s^{O(1)}$ as in Definition 10. If the circuit is instead randomized, we call this property random feasible interpolation.*

Interestingly, feasible interpolation is not affected by moving from classical automatability to randomized automatability. This is essentially folklore, but we reprove it for the sake of completeness.

► **Proposition 13.** *If a proof system S is weakly random automatable and closed under restrictions, then it has feasible interpolation by deterministic Boolean circuits.*

Proof. The proof is essentially the same as the original proof in [17], except for having to take randomness into account. Suppose R is a probabilistic automating algorithm for S . By Proposition 9.(ii), we can instead think of a family of randomized circuits $\{C_{n,s}\}_{n,s \in \mathbb{N}}$ that, for some fixed constant c , outputs proofs of size s^c when a proof of size s exists. Furthermore, let d be the constant in the exponent that bounds the blow-up in size happening in the closure under restrictions. Given a split formula $\varphi = \alpha \vee \beta$, we want to obtain an interpolant circuit C_φ .

Use the automating algorithm to find some proof of φ . Let s_0 be the size of such a proof. We first show that it is possible to extract a polynomial-size randomized circuit that computes the interpolant with one-sided error. Consider the circuit that takes as input the restriction ρ together with some random bits and proceeds to compute $C_{|\alpha|, s_0^d}(\alpha_{\upharpoonright \rho}, r)$. If this circuit finds a proof of $\alpha_{\upharpoonright \rho}$ and it is checked to be correct, we output 0; else, we output 1. We claim that for at least $2/3$ choices of r , this circuit is a correct interpolant (and, in fact, whenever it outputs 0, it is always correct). First, note that if we output 0 it is because a proof of $\alpha_{\upharpoonright \rho}$ was found, in which case it is correct to say that $\alpha_{\upharpoonright \rho}$ is a tautology. Otherwise, we will always output 1. The only problematic case is when the circuit outputs 1 while $\neg\beta_{\upharpoonright \rho}$ is satisfiable. If such was the case, then let σ be a satisfying assignment to the z -variables such that $\neg\beta_{\upharpoonright \rho, \sigma}$ is satisfied. Since S can prove φ in size s_0 and S is closed under restrictions, we know that S can prove $\varphi_{\upharpoonright \rho, \sigma}$ in size s_0^d , and this proof must clearly be deriving $\alpha_{\upharpoonright \rho, \sigma} = \alpha_{\upharpoonright \rho}$. Since $s_0^{cd} \geq s_0^d$, for a “good” choice of r the circuit $C_{|\alpha|, s_0^d}(\alpha_{\upharpoonright \rho}, r)$ would have found such a proof, so the only reason why we could have output 1 is that we chose a bad r . But this of course only happens with probability at most $1/3$. So this randomized circuit interpolates φ , makes only one-sided error, and has size polynomial in the size of the shortest proof.

We now replicate the strategy used in Adleman’s theorem ($\text{BPP} \subseteq \text{P}/\text{poly}$) to show that in fact randomness is not needed in the circuit. One can follow here the standard argument as presented, for example, by Arora and Barak [6, Thm. 7.15]: given the interpolant circuit F_φ , perform error reduction and then argue that there must be a string of random bits that is “good” for all inputs of the same size. The circuit no longer makes mistakes and computes f_φ as desired. ◀

► **Remark 14** (Constructive feasible interpolation). Our definition of feasible interpolation deviates from the one given in standard texts like that of Krajíček [35], and follows instead the one given by Pudlák [46], who imposes the condition that the interpolant circuit must be constructed from the given proof in polynomial time. Note that even if we adopted the non-constructive definition, the kind of feasible interpolation obtained by the construction above achieves this property anyway.

The constructivity requirement is useful to obtain a sort of converse of Impagliazzo's observation: if a propositional proof system has uniform polynomial-size proofs of its reflection principle, then it is weakly automatable (see [46, Prop. 3.6]).

Since randomness does not buy us anything when it comes to proof search, all hardness results immediately transfer to the randomized setting. In particular, for every proof system S simulating \mathbf{TC}^0 -Frege, S is not weakly random automatable unless Blum integers can be factored by polynomial-size randomized circuits. For weak proof systems where automatability is known to be \mathbf{NP} -hard, the systems cannot be automatable unless $\mathbf{NP} \subseteq \mathbf{BPP}$.

When moving to the quantum setting, unfortunately, we do not know of any way to get a deterministic circuit for the interpolant. Instead, we have the following natural version of Impagliazzo's observation.

► **Proposition 15.** *If a proof system is quantum automatable and closed under restrictions, then it admits feasible interpolation by quantum circuits.*

Proof. The proof follows the argument in Proposition 13, except we can no longer apply the final step to get rid of quantumness. The interpolant now is a quantum circuit, since it is simulating the quantum circuit $C_{|\alpha\rangle, s_0^d}(\alpha|_\rho)$. ◀

4 \mathbf{TC}^0 -Frege is hard to quantum automate

The quantum version of Impagliazzo's observation (Proposition 15) is the main tool needed for our hardness results, which we are now ready to state formally.

► **Theorem 16** (Main theorem). *If there exists a polynomial-time quantum algorithm that weakly automates \mathbf{TC}^0 -Frege, then the LWE assumption (Assumption 5) is broken by a uniform family of polynomial-size quantum circuits. Furthermore, if the weak automating algorithm is classical, the LWE assumption is broken by a uniform family of polynomial-size Boolean circuits.*

We can then extend the result to \mathbf{AC}^0 -Frege under a stronger assumption. This is done by applying the fact that \mathbf{TC}^0 -Frege proofs can be translated into \mathbf{AC}^0 -Frege proofs of subexponential size (see, for example, Theorems 2.5.6 and 18.7.3 in [35] or the original work on the non-automatability of \mathbf{AC}^0 -Frege [16]).

► **Corollary 17.** *If there exists a polynomial-time (quantum) algorithm that weakly automates \mathbf{AC}^0 -Frege, then the LWE assumption is broken by a uniform family of (quantum) circuits of size $2^{n^{o(1)}}$.*

We devote the rest of the paper to formally proving Theorem 16.

Suppose $h : \{0, 1\}^n \rightarrow \{0, 1\}^n$ is an injective and secure one-way function. Let x, y and z denote variables ranging over $\{0, 1\}^n$ and assume that \mathbf{TC}^0 -Frege is able to state and refute efficiently the following unsatisfiable formula,

$$(h(x) = z \wedge x_1 = 0) \wedge (h(y) = z \wedge y_1 = 1),$$

where x_1, y_1 are respectively the first bit of x and y . The unsatisfiability follows precisely from the fact that h is injective, and hence every output has a unique preimage.

15:14 Quantum Automating \mathbf{TC}^0 -Frege Is \mathbf{LWE} -Hard

If \mathbf{TC}^0 -Frege admits feasible interpolation, we are guaranteed the existence of a small circuit $C(z)$ such that

$$C(z) = \begin{cases} 0 & \text{if } h(x) = z \wedge x_1 = 0 \text{ is unsatisfiable} \\ 1 & \text{if } h(y) = z \wedge y_1 = 1 \text{ is unsatisfiable} \end{cases}$$

meaning that C is able to invert one bit of z . Since every output has a unique preimage, we can iterate the process to get the entire input string. This contradicts the assumption that h is one-way.

In order to instantiate the proof strategy to rule out quantum feasible interpolation, we now need a candidate one-way function that is injective and conjectured to be post-quantum secure and for which injectivity can be proven inside the proof system. Unfortunately, to the best of our knowledge, no such candidate function is currently known, or not with enough security guarantees². Alternatively, we may use other cryptographic objects that do achieve some form of injectivity, such as bit commitments, but the formalization of the latter does not seem simpler than the approach we follow instead. We now explain how we avoid this issue.

The most reliable post-quantum cryptographic assumptions have their security based on worst-case reductions to lattice problems conjectured to be hard. This is the case of the Learning with Errors framework [48], on which we base the security of the following class of candidate one-way functions. For these functions, as well as the basic properties of them that we employ, we follow the treatment of Micciancio [39]. We include the details for the proof complexity readers, who may not be familiar with these constructions.

► **Definition 18** (The candidate functions f_A). *Let $m = n^{O(1)}$, $q \leq 2^{n^{O(1)}}$, and $c = \alpha q / \sqrt{n}$, where $\alpha \in [0, 1]$. For every matrix $A \in \mathbb{Z}_q^{m \times n}$, we define the function $f_A : \mathbb{Z}_q^n \times \{\varepsilon \in \mathbb{Z}_q^m : |\varepsilon| \leq 10c\sqrt{mn}\} \rightarrow \mathbb{Z}_q^m$ as*

$$f_A(s, \varepsilon) := (As + \varepsilon) \bmod q.$$

At this point, we would like to show inside \mathbf{TC}^0 -Frege that the conjunction

$$(f_A(x) = z \wedge x_1 = 0) \wedge (f_A(y) = z \wedge y_1 = 1) \tag{1}$$

is a contradiction, where A is represented by free variables and x_1 and y_1 refer to the first bits of x and y . Unfortunately, the problem concerning injectivity mentioned above remains. The formula is not necessarily a contradiction, since for some choices of A , the function f_A is not injective. We can show, however, that with high probability over the choice of A , the function f_A will satisfy two conditions that imply injectivity. Namely, A will be full rank and the shortest vector in the q -ary lattice spanned by A will be large enough.

The following proposition, which captures this idea, is standard. We reprove it here for the sake of completeness, since we shall formalize part of it inside the proof systems later.

► **Proposition 19.** *Let $n \in \mathbb{N}$, $m = n \log n$ and $q \geq n^5$. With high probability over the choice of $A \in \mathbb{Z}_q^{m \times n}$, $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$. Furthermore, when these hold, the function f_A is injective.*

² In a previous version of this work we formalized the injectivity of several group-based post-quantum cryptographic assumptions, such as MOBS [47], as well as variants of supersingular isogeny-based Diffie-Hellman protocols, which unfortunately all happen to be now broken more or less efficiently.

Proof. From Lemma 4.i we get that $\Pr_{A \sim \mathcal{U}(\mathbb{Z}_q^{m \times n})}[\text{rank}(A) < n] \leq n/q^{m-n+1}$. By Lemma 4.ii we can see that

$$\Pr_{A \sim \mathcal{U}(\mathbb{Z}_q^{m \times n})}[\lambda_1(\mathcal{L}(A)) \leq 20c\sqrt{mn} \mid \text{rank}(A) = n] \leq \frac{(40c\sqrt{mn} + 1)^m}{q^{m-2n-1}}.$$

The probability that a random A does not satisfy the conditions in the statement is at most the sum of the two probabilities above, which are both negligible for our choice of m and q .

For injectivity, suppose for contradiction that there exist $x, x', \varepsilon, \varepsilon'$, with either $x \neq x'$ or $\varepsilon \neq \varepsilon'$, causing a collision $f_A(x, \varepsilon) = Ax + \varepsilon = Ax' + \varepsilon' = f_A(x', \varepsilon')$. We have two cases.

- (a) If $\varepsilon = \varepsilon'$, then the collision happens if and only if $\text{rank}(A) < n$, which contradicts the assumption.
- (b) Suppose that $\varepsilon \neq \varepsilon'$. We have that $\varepsilon - \varepsilon' = A(x' - x)$. Since the norm of $\varepsilon - \varepsilon'$ is at most $20c\sqrt{nm}$, by transitivity we have that the length of $A(x' - x)$ is bounded by the same quantity. However, the latter belongs to the lattice and therefore we obtain a contradiction. \blacktriangleleft

Luckily for us, these two conditions are succinctly certifiable! Indeed, to certify that the matrix A is full rank we may provide a left-inverse A_L^{-1} such that $A_L^{-1}A = I_n$. Unfortunately, we cannot guarantee that all injective f_A have simple certificates of the second property, $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$. Nevertheless, we show in Section 4.2 that almost all of them do. These certificates take the form of sets $W = \{w_1, \dots, w_m\} \subseteq \Delta_q^*(A)$ of short linearly independent vectors in the dual of the q -ary lattice. We prove – using the left inequality of Banaszczyk’s Transference Theorem – that such a set suffices to certify the second property, and then show – using the right side of Banaszczyk’s Transference Theorem – that the certificate W exists with high probability.

► **Definition 20** (Certificate of injectivity). *A certificate of injectivity for the function f_A , with $A \in \mathbb{Z}_q^{m \times n}$, is a pair (A_L^{-1}, W) such that A_L^{-1} is a left-inverse so that $A_L^{-1}A = I_n$, and $W = \{w_1, \dots, w_m\} \subseteq \Delta_q^*(A)$ is a set of m linearly independent vectors such that $\max_{i \in [m]} \|w_i\| < 1/20c\sqrt{nm}$.*

The relation between injectivity and these certificates is made formal as follows.

- **Proposition 21.** *Let $n \in \mathbb{N}$, $m = n \log n$, $q = n^5$, and $A \in \mathbb{Z}_q^{m \times n}$. The following hold:*
- (i) *if there is a certificate of injectivity (A_L^{-1}, W) for f_A , then f_A is injective;*
 - (ii) *if $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$, then there exists a certificate of injectivity for f_A ;*
 - (iii) *with high probability over the choice of A , $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$.*

Observe that given a certificate (A_L^{-1}, W) , verifying its correctness is a rather simple task: it is sufficient check that $A_L^{-1}A = I_n$, to verify that W is a set of linearly independent vectors in $\Delta_q^*(A)$, and finally to ensure that the vectors in W are small enough.

Let us return to the propositional system. We denote by $\text{INJ}(f_A)$ the propositional formula encoding that f_A is injective. From this formula, \mathbf{TC}^0 -Frege can derive that (1) is a contradiction. However, $\text{INJ}(f_A)$ is false if we leave A as free variables. We instead prove $\text{INJ}(f_{A_0})$ for concrete injective f_{A_0} , where A_0 is hardwired. The concrete f_{A_0} for which we do it are the ones that admit a certificate of injectivity.

Essentially, we formalize inside \mathbf{TC}^0 -Frege that a certificate of injectivity implies injectivity. That is,

$$\mathbf{TC}^0\text{-Frege} \vdash \text{CERT}(C_A) \rightarrow \text{INJ}(f_A), \quad (2)$$

where $\text{CERT}(C_A)$ encodes that C_A is a correct certificate for f_A . Here C_A and A are free variables. This implication is precisely Proposition 21.i above. The proof inside the system is carried out in Section 4.3.

Now, given a concrete certificate C_{A_0} for f_{A_0} , the formula $\text{CERT}(C_{A_0})$ is derivable inside TC^0 -Frege, which amounts to the system verifying the certificate's correctness. From this, TC^0 -Frege proves $\text{INJ}(f_{A_0})$.

The rest of this section completes the missing parts in the proof. Section 4.1 sketches the known fact that f_A is worst-case one-way based on the hardness of Learning with Errors, while Section 4.2 proves Proposition 21 showing the existence of certificates. We remark that the arguments and techniques are standard in cryptography and readers familiar with the area might want to skip them. We include them for the sake of completeness and to cater to the proof complexity reader that may have never come across these ideas before, and we refer to standard texts like [39] for further details. Finally, Section 4.3 formalizes the certificate-to-injectivity implication above inside the theory $\text{LA}_{\mathbb{Q}}$, which propositionally translates into TC^0 -Frege. Section 4.4 reconstructs the final argument.

4.1 Security of f_A

The functions in $\{f_A\}_{A \in \mathbb{Z}_q^{m \times n}}$ very closely resemble the standard Learning with Errors functions, the only difference being that we have set a maximum value on the magnitude of the error vectors and allowed these to be chosen as a uniform part of the input (instead of being sampled from a Gaussian distribution). We now observe that inverting these functions allows us to invert LWE with high probability over the choice of the error vector.

► **Lemma 22** ([39, Section 3.2]). *Suppose there exists an algorithm B taking as input $A \in \mathbb{Z}_q^{m \times n}$ and a string z and outputting a preimage in $f_A^{-1}(z)$ with probability p . Then, LWE can be broken with probability $0.99p$ over the choice of the error vector ε and the internal randomness of B .*

Proof. It suffices to show that with high probability the error vectors in the standard Learning with Errors functions are bounded as in our definition of f_A , and thus the same inverter for f_A will also work for most of the original LWE instances. This follows from a standard Gaussian tail bound. Thus, if we are able to invert f_A on all outputs with probability p , then we are able to invert its corresponding LWE function with probability, say, $0.99p$ over the choice of ε . ◀

Note that it is in fact possible to invert with all but negligible probability, since finding a vector whose norm is far above the expectation with high probability requires that several independently sampled coordinates all return values much larger than the expected one. For simplicity, we use this weaker result which suffices for our applications.

4.2 Existence of certificates of injectivity: Proof of Proposition 21

This section proves the three statements of Proposition 21.

► **Proposition 21.i** (Correctness of certificates). *If there is a certificate of injectivity (A_L^{-1}, W) for f_A , then $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{nm}$, and thus the function f_A is injective.*

Proof. As discussed in the proof of Proposition 19, f_A is injective if and only if both $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20c\sqrt{mn}$. By elementary linear algebra, $\text{rank}(A) = n$ if and only if there exists a A_L^{-1} . As previously observed we know that $\lambda_1(\Delta_q(A)) = \min(q, \lambda_1(\mathcal{L}(A)))$ and since $20c\sqrt{mn} \leq q/m$, therefore it suffices to show that the existence of W as described above implies that $\lambda_1(\Delta_q(A)) > 20c\sqrt{mn}$.

Because it is a q -ary lattice we know that $\text{rank}(\Delta_q(A)) = m$. By rearranging the left inequality of the Transference Theorem for rank- m lattices, we get that $\lambda_1(\Delta_q(A)) \geq 1/\lambda_m(\Delta_q^*(A))$. By the definition of W , we conclude that $\lambda_m(\mathcal{L}^*) < 1/20c\sqrt{nm}$, which implies that $\lambda_1(\mathcal{L}_q(A)) = \lambda_1(\Delta_q(A)) > 20c\sqrt{nm}$.

Injectivity of f_A now immediately follows from the argument in Proposition 19. \blacktriangleleft

► **Proposition 21.ii** (Conditional existence of certificates). *If $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$, then there exists a certificate of injectivity for f_A .*

Proof. Since we assumed that $\text{rank}(A) = n$, there exists a left inverse A_L^{-1} for A . It therefore suffices to show that if $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm}$, then there exists a set of vectors W satisfying the conditions above.

By the right inequality of the Transference Theorem for rank- m lattices, we can obtain that $\lambda_m(\Delta_q^*(A)) \leq m/\lambda_1(\Delta_q(A))$. Since $\lambda_1(\mathcal{L}_q(A)) > 20mc\sqrt{nm} \leq q$, and $\lambda_1(\Delta_q(A)) = \min(q, \lambda_1(\mathcal{L}_q(A))) = \lambda_1(\mathcal{L}_q(A))$ there must exist a set of m linearly independent vectors in $\Delta_q^*(A)$, such that $\max_{i \in [m]} \|w_i\| < 1/20c\sqrt{nm}$. \blacktriangleleft

► **Proposition 21.iii** (Existence of certificates with high probability). *Let $n \in \mathbb{N}$, $m = n \log n$, $q \geq n^5$, $c \leq \sqrt{nm}/40$ and $A \in \mathbb{Z}_q^{m \times n}$ be sampled uniformly at random. The probability that $\text{rank}(A) = n$ and $\lambda_1(\mathcal{L}_q(A)) > 20cm\sqrt{nm}$ is at least*

$$1 - \frac{n}{q^{m-n+1}} - m^{3m-(m-2n-1)\log_m q}.$$

This probability is at least exponentially close to 1 for our choice of q and m .

Proof. For the following equations we define E_{short} to be the event that $\lambda_1(\mathcal{L}_q(A)) \leq 20cm\sqrt{nm}$. We have that

$$\begin{aligned} \Pr_A[\text{rank}(A) \neq n \vee E_{\text{short}}] &= \Pr_A[\text{rank}(A) \neq n] + \Pr_A[E_{\text{short}} \wedge \text{rank}(A) = n] \\ &\leq \Pr_A[\text{rank}(A) \neq n] + \Pr_A[E_{\text{short}} \mid \text{rank}(A) = n]. \end{aligned}$$

By Lemma 4.i, we know that $\Pr_A[\text{rank}(A) \neq n] \leq n/q^{m-n+1}$, and by the second point of Lemma 4.ii, we have that

$$\begin{aligned} \Pr[E_{\text{short}} \mid \text{rank}(A) = n] &\leq \frac{(40mc\sqrt{mn})^m}{q^{m-2n-1}} \leq \frac{(m^2n)^m}{q^{m-2n-1}} \\ &\leq \frac{m^{3m}}{m^{(m-2n-1)\log_m q}} = m^{3m-(m-2n-1)\log_m q}. \blacktriangleleft \end{aligned}$$

4.3 Formalization

At this point, the only thing left is the formalization of the implication $\text{CERT}(C_A) \rightarrow \text{INJ}(f_A)$ inside the propositional system. Since this is rather cumbersome, we work instead in the more convenient theory **LA** of linear algebra of Soltys and Cook [50]. The theory, however, is field-independent, which means we cannot state or prove properties about the ordering of the rationals, which is needed in our arguments. Furthermore, we sometimes use the fact that certain matrices are over the integers, so we must be able to identify certain elements as integers. To solve this, we introduce a conservative extension of the theory, called **LA $_{\mathbb{Q}}$** , which assumes the underlying field to be \mathbb{Q} .

4.3.1 The conservative extension $\text{LA}_{\mathbb{Q}}$

On top of the existing symbols of the language of LA , we have two new predicate symbols int and $<_{\mathbb{Q}}$. The int predicate, applied to a field element q , written $\text{int}(q)$, is supposed to be true whenever the rational q is an integer.

The symbol $<_{\mathbb{Q}}$, which we overload onto $<$ in what follows, is intended to represent the usual ordering relation over the rationals. For convenience, we also add the symbol $x \leq y$ together with an axiom imposing that its meaning is $x < y \vee x = y$. Recall that equality of field elements was a symbol in the base theory LA , which already equipped it with its corresponding axioms.

We now extend the axiom-set of LA with axioms for the new symbols. Recall that the original axioms of LA are listed in Appendix B.

Axioms for int

- (Int₁) $\text{int}(0)$
- (Int₂) $\text{int}(1)$
- (Int₃) $\text{int}(-1)$
- (Int₄) $\text{int}(x) \wedge \text{int}(y) \rightarrow \text{int}(x + y)$
- (Int₅) $\text{int}(x) \wedge \text{int}(y) \rightarrow \text{int}(x \cdot y)$
- (Int₆) $\text{int}(x) \wedge 0 < x \rightarrow 1 \leq x$

Axioms for $<_{\mathbb{Q}}$

- (Ord₁) $x \leq y \leftrightarrow (x < y \vee x = y)$
- (Ord₂) $\neg(x < x)$

$$\text{(Ord}_3\text{)} \quad x < y \rightarrow \neg(x = y)$$

$$\text{(Ord}_4\text{)} \quad x < y \wedge y < z \rightarrow x < z$$

$$\text{(Ord}_5\text{)} \quad \neg(x = y) \rightarrow x < y \vee y < x$$

$$\text{(Ord}_6\text{)} \quad x \leq y \wedge z \leq w \rightarrow x + z \leq y + w$$

$$\text{(Ord}_7\text{)} \quad 0 \leq x \wedge 0 \leq y \rightarrow 0 \leq x \cdot y$$

$$\text{(Ord}_8\text{)} \quad 0 \leq x \cdot x$$

$$\text{(Ord}_9\text{)} \quad 0 \leq x \wedge y < 0 \rightarrow x \cdot y \leq 0$$

$$\text{(Ord}_{10}\text{)} \quad a, b, c, d \geq 0 \wedge a < b \wedge c < d \rightarrow ac < bd$$

The axioms for the ordering symbols are essentially the axioms of a strict total order (Ord₂-Ord₅), together with an axiom connecting \leq and $<$ (Ord₁). We then ensure the compatibility of the operations with the ordering relation, (Ord₆-Ord₁₀). Our axioms are not necessarily minimal, since we are interested in convenience rather than succinctness.

The axioms for int are more ad hoc and it might seem that they are not enough to fix the correct interpretation of the symbol. Indeed, the axioms for int only really force that addition and multiplication are closed under this predicate and that every integer in the standard model can be argued to be an integer in $\text{LA}_{\mathbb{Q}}$, but they do not identify \mathbb{Z} as a substructure of \mathbb{Q} . The reason this is not an issue is that we are only interested in $\text{LA}_{\mathbb{Q}}$ for its propositional translation. For our purposes, these axioms are the only ones we need to prove the required claims about lattices, and once we translate to the propositional setting, the symbols will take the standard intended interpretation.

Crucially, theorems of $\text{LA}_{\mathbb{Q}}$ admit succinct TC^0 -Frege proofs. This requires extending the propositional translation of Soltys and Cook to the new symbols and axioms. We do this in detail in Appendix A.2 of the full version [7]. In this version, Appendix A contains a sketch the construction and the statement of the result.

4.3.2 Formalization of the proofs

We are ready to present the formal proofs needed inside our theory. In what follows, LA_{\dots} stands for the corresponding axiom in Appendix B.

First, we observe that under our new axioms, $\text{LA}_{\mathbb{Q}}$ can argue that the inner product of a vector with itself is non-negative. Recall that the inner product operator $\langle u, v \rangle$ is a defined term in LA , namely $u \cdot v^{\top}$.

► **Lemma 23.** *Provably in $\text{LA}_{\mathbb{Q}}$, for every $v \in \mathbb{Q}^n$, $0 \leq \langle v, v \rangle$.*

Proof. Unfolding the definition of the term $\langle v, v \rangle$ in $\text{LA}_{\mathbb{Q}}$, $\langle v, v \rangle = \sum_{i=1}^n v_i \cdot v_i$, so by axiom (Ord₈) each term in the sum satisfies $v_i \cdot v_i \geq 0$, and by repeatedly applying axiom (Ord₆), the entire sum can be proven to be non-negative. ◀

We now formalize inside $\text{LA}_{\mathbb{Q}}$ the classical Cauchy-Schwartz inequality.

► **Lemma 24** (Cauchy-Schwartz in $\text{LA}_{\mathbb{Q}}$). *The theory $\text{LA}_{\mathbb{Q}}$ proves that for every $u, v \in \mathbb{Q}^n$, $\langle u, v \rangle^2 \leq \langle u, u \rangle \cdot \langle v, v \rangle$.*

Proof. We first show that $\text{LA}_{\mathbb{Q}}$ can derive the following equality,

$$\frac{1}{\langle v, v \rangle} \langle (\langle v, v \rangle u - \langle u, v \rangle v), (\langle v, v \rangle u - \langle u, v \rangle v) \rangle = \langle u, u \rangle \langle v, v \rangle - \langle u, v \rangle^2, \quad (3)$$

where $\frac{1}{\langle v, v \rangle}$ can be explicitly referred to in $\text{LA}_{\mathbb{Q}}$ as $\langle v, v \rangle^{-1}$.

We do this explicitly by deriving the following chain of equalities,

$$\frac{1}{\langle v, v \rangle} \langle (\langle v, v \rangle u - \langle u, v \rangle v), (\langle v, v \rangle u - \langle u, v \rangle v) \rangle = \quad (\text{LA7.b})$$

$$\frac{1}{\langle v, v \rangle} \langle (\langle v, v \rangle u - \langle u, v \rangle v), (\langle v, v \rangle u) \rangle + \langle (\langle v, v \rangle u - \langle u, v \rangle v), (-\langle u, v \rangle v) \rangle = \quad (\text{LA7.a-c})$$

$$\begin{aligned} \frac{1}{\langle v, v \rangle} (\langle v, v \rangle^2 \langle u, u \rangle - \langle v, v \rangle \langle v, u \rangle^2) &= \quad (\text{LA7.c}) \\ \langle v, v \rangle \langle u, u \rangle - \langle v, u \rangle^2. & \end{aligned}$$

Observe now that from Lemma 23 above, we know that $\langle (\langle v, v \rangle u - \langle u, v \rangle v), (\langle v, v \rangle u - \langle u, v \rangle v) \rangle$ is non-negative, and it is also clear that $0 \leq \frac{1}{\langle v, v \rangle}$: we know again that $\langle v, v \rangle \geq 0$, and $\langle v, v \rangle \langle v, v \rangle^{-1}$ is either 0 or 1, by LA3.d; if it is 0, we are done, and if it is 1, then by axiom (Ord₉) we can get a contradiction.

Thus, the left-hand side of Equation (3) is positive and $\text{LA}_{\mathbb{Q}}$ can derive this fact. Then, by the transitivity axiom (Ord₄), the right-hand side of Equation (3) is non-negative as well. Rearranging the inequality using (Ord₆), the inequality follows. ◀

The other technical component needed in the final proof is a weakening of the lower bound in Banaszczyk's Transference Theorem (see Theorem 3). Informally, we need to prove that for every $A \in \mathbb{Q}^{m \times n}$, every non-zero vector $v \in \mathcal{L}(A)$ and any set of linearly independent vectors $W = \{w_1, \dots, w_n\} \subseteq \mathcal{L}^*(A)$, $\langle v, v \rangle \cdot \langle w_i, w_i \rangle \geq 1$ for some $i \in [n]$.

In order for $\text{LA}_{\mathbb{Q}}$ to process the conditions of the theorem, we provide certificate-like objects ensuring all the different hypotheses. For example, when we quantify over a vector v belonging to a lattice $\mathcal{L}(A)$, we provide the vector of coefficients c_v such that $Ac_v = v$. Note as well that when we quantify over matrices with elements in \mathbb{Z} , we are using the int predicate under the hood to enforce the entries to be integers. As a final remark, recall that we do not have existential quantifiers in LA, but whenever we do some existential quantification in the following lemmas we are quantifying over small finite domains, meaning we can write everything as a small disjunction.

► **Lemma 25** (Banaszczyk's left inequality in $\text{LA}_{\mathbb{Q}}$). *The theory $\text{LA}_{\mathbb{Q}}$ proves the following implication. Let $A \in \mathbb{Z}^{m \times n}$, $B \in \mathbb{Q}^{n \times n}$, $v \in \mathbb{Q}^n$, $c_v \in \mathbb{Z}^m$, $W = [w_1 | \dots | w_n] \in \mathbb{Q}^{m \times n}$, $c_W = [c_{w_1} | \dots | c_{w_n}] \in \mathbb{Z}^{m \times n}$, $W' \in \mathbb{Q}^{m \times n}$ fulfilling the following conditions:*

1. *the vector v is non-zero, $v \neq 0_n$;*
2. *the vector v belongs to the lattice $\mathcal{L}(A)$, $v = Ac_v$;*

15:20 Quantum Automating TC^0 -Frege Is LWE -Hard

3. the vectors in W belong to the dual lattice³ $\mathcal{L}^*(A)$, $w_i = ABc_{w_i}$ for all $i \in [n]$;
 4. $(A^\top A)B = I_n$;
 5. the vectors in W are linearly independent, $W'W^\top = I_n$.
- Then, for some $i \in [n]$, $\langle v, v \rangle \cdot \langle w_i, w_i \rangle \geq 1$.

Proof. The proof has two steps. First, we show that for all $i \in [n]$, $\langle v, w_i \rangle \in \mathbb{Z}$. To do this we use the following chain of equalities, where w is some arbitrary column w_i of W , and where the comments on the side refer to either axioms of $\text{LA}_{\mathbb{Q}}$ or the assumptions in the statement of the lemma:

$$\begin{aligned}
 \langle v, w \rangle &= \langle Ac_v, ABc_w \rangle && \text{(by ass. 2 and 3)} \\
 &= c_v^\top A^\top ABc_w && \text{(by def. of dot product)} \\
 &= c_v^\top (A^\top A)Bc_w && \text{(by associativity, LA5.i)} \\
 &= c_v^\top c_w. && \text{(by ass. 4)}
 \end{aligned}$$

By assumption, the entries in both c_v and c_w have integer entries, so by the closure under integer multiplication and addition (Int_4 and Int_5) we have that $c_v^\top c_w$ is an integer, and thus we deduce that for all $i \in [n]$, $\langle v, w_i \rangle \in \mathbb{Z}$.

In the second step of the proof, we show that there is $i \in [n]$ such that $\langle v, w_i \rangle \neq 0$. We consider the vector $s := W^\top v$. Note that by definition, the j -th entry of s is $\langle w_j, v \rangle$. We can multiply both sides by the same matrix W' , leading to $W's = W'W^\top v$. Using associativity (LA5.i), assumption (5) and properties of the identity matrix (LA5.f), we get that $W's = v$. Suppose that for all $i \in [n]$, $\langle v, w_i \rangle = 0$. Then, by definition, $s = 0_n$. We can easily derive (using LA3.a, LA3.c and LA3.i) that $W's = 0$ and therefore $v = 0$. This contradicts assumption (1).

Finally, let i denote the particular index for which we have now derived that simultaneously $\langle v, w_i \rangle \in \mathbb{Z}$ and $\langle v, w_i \rangle \neq 0$. By axiom (Ord_8), $\langle v, w_i \rangle^2 \geq 0$. Furthermore, it is easy to derive already in LA that for any field elements a and b , if $a \neq 0$ and $b \neq 0$, then $ab \neq 0$ (this follows immediately from axioms LA3.a-d). Thus, by axiom (Ord_1), $\langle v, w_i \rangle^2 > 0$. Recall now that by axiom (Int_6) of $\text{LA}_{\mathbb{Q}}$, every non-zero positive integer is greater or equal than 1, so $\langle v, w_i \rangle^2 \geq 1$. Then, the Cauchy-Schwartz inequality from Lemma 24 gives us

$$1 \leq \langle v, w_i \rangle^2 \leq \langle v, v \rangle \cdot \langle w_i, w_i \rangle,$$

which together with transitivity (axiom Ord_4 together with Ord_1) yields the desired $1 \leq \langle v, v \rangle \cdot \langle w_i, w_i \rangle$. \blacktriangleleft

We are now ready to prove in $\text{LA}_{\mathbb{Q}}$ that a correct certificate of injectivity implies the injectivity of f_A . Informally, we aim to prove that given a certificate of injectivity as in Definition 20, the function f_A is injective. As before, we need to provide some additional objects together with the certificate to make sure $\text{LA}_{\mathbb{Q}}$ can reason about this conditional implication and carry out the verification of the certificate.

► Lemma 26 (Certificate-implies-injectivity in $\text{LA}_{\mathbb{Q}}$). *Let $A \in \mathbb{Z}^{m \times n}$, $B \in \mathbb{Q}^{n \times n}$, $v_1 \in \mathbb{Q}^n$, $c_{v_1} \in \mathbb{Z}^m$, $v_2 \in \mathbb{Q}^n$, $c_{v_2} \in \mathbb{Z}^m$, $\varepsilon_1 \in \mathbb{Q}^n$, $\varepsilon_2 \in \mathbb{Q}^n$, $W = [w_1 | \dots | w_n] \in \mathbb{Q}^{m \times n}$, $c_W = [c_{w_1} | \dots | c_{w_n}] \in \mathbb{Z}^{m \times n}$, $W' \in \mathbb{Q}^{m \times n}$ fulfilling the following conditions:*

1. the vector v_1 belongs to the lattice $\mathcal{L}(A)$, $v_1 = Ac_{v_1}$;
2. the vector v_2 belongs to the lattice $\mathcal{L}(A)$, $v_2 = Ac_{v_2}$;

³ This dual lattice, in fact, admits a closed form for its base, as in Lemma 1. In particular, B can be seen as $(A^\top A)^{-1}$.

3. the vectors v_1 and v_2 are distinct, $v_1 \neq v_2$;
4. the vectors in W belong to the dual lattice $\mathcal{L}^*(A)$, $w_i = ABc_{w_i}$ for all $i \in [n]$;
5. $(A^\top A)B = I_n$;
6. the vectors in W are linearly independent, $W'W^\top = I_n$;
7. $\langle w_i, w_i \rangle < 1/400c^2nm$ for all $i \in [n]$;
8. $\langle \varepsilon_2 - \varepsilon_1, \varepsilon_2 - \varepsilon_1 \rangle < 400c^2nm$.

Then, $Av_1 + \varepsilon_1 \neq Av_2 + \varepsilon_2$.

Proof. The proof proceeds by contradiction. Suppose that $Av_1 + \varepsilon_1 = Av_2 + \varepsilon_2$, meaning that a collision exists in the range of f_A . By simple algebraic manipulations in $\mathbf{LA}_\mathbb{Q}$, we derive that $A(v_1 - v_2) = \varepsilon_2 - \varepsilon_1$. Let $v := A(v_1 - v_2)$ and $\varepsilon := \varepsilon_2 - \varepsilon_1$, so that we have $v = \varepsilon$.

Observe now that assumptions (7) and (8), together with axiom (Ord₁₀) and (LA3.d) give us $\langle w_i, w_i \rangle \langle \varepsilon, \varepsilon \rangle < 1$ for every $i \in [n]$. With the existing assumptions of the theorem we can in fact apply Lemma 25 to v , getting that there exists i such that $\langle v, v \rangle \langle w_i, w_i \rangle \geq 1$. Since $v = \varepsilon$, we get $\langle \varepsilon, \varepsilon \rangle \langle w_i, w_i \rangle \geq 1$, but this means

$$1 \leq \langle \varepsilon, \varepsilon \rangle \langle w_i, w_i \rangle < 1.$$

We remark that while technically the transitivity axiom (Ord₄) is stated for strict orders, the existing set of axioms immediately implies the “mixed” version, namely that for field elements a , b and c , if $a \leq b$ and $b < c$, then $a < c$. Thus we now have $1 < 1$, but this contradicts axiom (Ord₂). ◀

4.4 Proof of Theorem 16

We are ready to put all the pieces together.

Proof of Theorem 16. Suppose that \mathbf{TC}^0 -Frege is weakly quantum automatable, that is, suppose that S is a quantum automatable proof system simulating \mathbf{TC}^0 -Frege. Let Q be the quantum algorithm automating S . We describe a quantum algorithm Q' that takes as input a matrix A defining a function f_A as in Definition 18 and an output z of this function and succeeds in finding a preimage of z with high probability.

For a specific input matrix A_0 , consider the formula $\mathbf{CERT}(C_A) \rightarrow \mathbf{INJ}(f_A)$, where C and A are free variables. In Lemma 26 this implication was proven inside $\mathbf{LA}_\mathbb{Q}$, and by the propositional translation for $\mathbf{LA}_\mathbb{Q}$ in Theorem 28 we get an efficient proof inside \mathbf{TC}^0 -Frege, and thus also in S . Craft now the formula $\mathbf{INJ}(f_{A_0})$ for the particular A_0 received as input. By Proposition 21, for most f_{A_0} there exists a certificate of injectivity C_{A_0} such that $\mathbf{CERT}(C_{A_0})$ is true and, in fact, has no free variables. Consider this certificate as a partial restriction and apply it to the implication above. Since \mathbf{TC}^0 -Frege is closed under restrictions, there must be a polynomial-size proof of $\mathbf{INJ}(f_{A_0})$, and so S also proves this efficiently. Recall that, as noted in Remark 14, Impagliazzo’s observation guarantees that under the existence of an automating algorithm we get constructive feasible interpolation, so from the proof of $\mathbf{INJ}(f_{A_0})$ we can get a circuit that breaks one bit of the given output. By iterating this process we can recover the entire preimage. This procedure works as long as f_{A_0} is injective and admits a certificate of injectivity, but by Proposition 21 this is the case with overwhelming probability. Then, by Lemma 22, we break LWE and get the desired conclusion. ◀

The proof above is phrased from the starting assumption of a weak automating algorithm rather than feasible interpolation. The reason is that, intuitively, feasible interpolation alone does not seem to immediately break the cryptographic assumption: for every fixed matrix

A , feasible interpolation only seems to guarantee the existence (with high probability) of a circuit breaking f_A , but this circuit seems to essentially depend on A . By starting the argument from an automating algorithm, we have a uniform way of finding the proofs of injectivity for each particular f_A to then construct the corresponding interpolating circuit.

While we find this more intuitive, we can still phrase the argument directly in terms of interpolation (and hence rule out this too under the same assumption). It suffices to argue that TC^0 -Frege can refute the contradictory formulas

$$(f_A(x) = z \wedge x_i = 0) \wedge ((f_A(y) = z \wedge y_i = 1) \wedge \text{CERT}(C_A)),$$

where x_i and y_i refer to the i -th respective bits, and $\text{CERT}(C_A)$ is the certificate predicate, as in Equation (2). Observe that this is still a split formula, since the variables encoding the certificate C_A appear only on the right-hand side. That the proof system can show this is a contradiction follows immediately from the fact that it can prove the implication in Equation (2). More importantly, the refutation of this formula is uniform and known, with A as free variables, meaning we can extract the interpolants directly. It is not hard to see that interpolating on this formula we can still break the same functions that we would break with the aid of an automating algorithm. This remark is due to Impagliazzo. Thus, the following corollary also follows from our formalization.

► **Corollary 27.** *If TC^0 -Frege admits feasible interpolation by (quantum) circuits, then the LWE assumption can be broken by a uniform family of polynomial-size (quantum) circuits.*

References

- 1 Dorit Aharonov and Oded Regev. Lattice problems in $\text{NP} \cap \text{coNP}$. *Journal of the ACM (JACM)*, 52(5):749–765, 2005.
- 2 Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, 1996.
- 3 Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293, 1997.
- 4 Michael Alekhnovich, Sam Buss, Shlomo Moran, and Toniann Pitassi. Minimum propositional proof length is NP -hard to linearly approximate. In *Mathematical Foundations of Computer Science 1998: 23rd International Symposium*, pages 176–184. Springer, 2006.
- 5 Michael Alekhnovich and Alexander A Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM Journal on Computing*, 38(4):1347–1363, 2008.
- 6 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
- 7 Noel Arteche, Gaia Carenini, and Matthew Gray. Quantum automating TC^0 -Frege is LWE -hard. *Electronic Colloquium on Computational Complexity (ECCC)*, 2024. URL: <https://eccc.weizmann.ac.il/report/2024/029/>.
- 8 Albert Atserias and María Luisa Bonet. On the automatizability of resolution and related propositional proof systems. *Information and Computation*, 189(2):182–201, 2004.
- 9 Albert Atserias and Elitza Maneva. Mean-payoff games and propositional proofs. *Information and Computation*, 209(4):664–691, 2011.
- 10 Albert Atserias and Moritz Müller. Automating resolution is NP -hard. *Journal of the ACM (JACM)*, 67(5):1–17, 2020.
- 11 Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296:625–635, 1993.
- 12 P. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 274–282, 1996.

- 13 Arnold Beckmann, Pavel Pudlák, and Neil Thapen. Parity games and propositional proofs. *ACM Transactions on Computational Logic (TOCL)*, 15(2):1–30, 2014.
- 14 Zoë Bell. Automating regular or ordered resolution is **NP**-hard. *Electronic Colloquium on Computational Complexity (ECCC)*, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/105/>.
- 15 Huck Bennett and Chris Peikert. Hardness of the (Approximate) Shortest Vector Problem: A Simple Proof via Reed-Solomon Codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, volume 275, pages 37:1–37:20, 2023.
- 16 María Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth Frege proofs. *computational complexity*, 13:47–68, 2004.
- 17 María Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for Frege systems. *SIAM Journal on Computing*, 29(6):1939–1967, 2000.
- 18 Sam Buss and Jakob Nordström. Proof complexity and SAT solving. *Handbook of Satisfiability*, 336:233–350, 2021.
- 19 Samuel R Buss. On Gödel’s theorems on lengths of proofs II: Lower bounds for recognizing k symbol provability. In *Feasible mathematics II*, pages 57–90. Springer, 1995.
- 20 Stephen Cook and Phuong Nguyen. *Logical Foundations of Proof Complexity*. Cambridge University Press, 2010.
- 21 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Logic, Automata, and Computational Complexity*, 1979.
- 22 Stefan Dantchev, Barnaby Martin, and Stefan Szeider. Parameterized proof complexity. *Computational Complexity*, 20:51–85, 2011.
- 23 Susanna F. de Rezende, Mika Göös, Jakob Nordström, Toniann Pitassi, Robert Robere, and Dmitry Sokolov. Automating algebraic proof systems is **NP**-hard. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 209–222, 2021.
- 24 Michal Garlík. Failure of feasible disjunction property for k -DNF resolution and **NP**-hardness of automating it, 2020. [arXiv:2003.10230](https://arxiv.org/abs/2003.10230).
- 25 Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is **NP**-hard. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 68–77, 2020.
- 26 Rishab Goyal, Venkata Koppula, Satyanarayana Vusirikala, and Brent Waters. On perfect correctness in (lockable) obfuscation. In *Theory of Cryptography Conference*, pages 229–259. Springer, 2020.
- 27 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.
- 28 Lei Huang and Toniann Pitassi. Automatizability and simple stochastic games. In *International Colloquium on Automata, Languages, and Programming*, pages 605–617. Springer, 2011.
- 29 Dmitry Itsykson and Artur Riazanov. Automating OBDD proofs is **NP**-hard. In *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, 2022.
- 30 Kazuo Iwama. Complexity of finding short resolution proofs. In *Mathematical Foundations of Computer Science 1997*, pages 309–318. Springer Berlin Heidelberg, 1997.
- 31 Emil Jeřábek. *Weak pigeonhole principle, and randomized computation*. PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2005.
- 32 Emil Jeřábek. Elementary analytic functions in VTC^0 . *Annals of Pure and Applied Logic*, 174(6):103269, 2023.
- 33 Jan Krajíček. *Bounded Arithmetic, Propositional Logic and Complexity Theory*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1995.
- 34 Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(2):457–486, 1997.

- 35 Jan Krajíček. *Proof Complexity*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2019.
- 36 Jan Krajíček. Information in propositional proofs and algorithmic proof search. *The Journal of Symbolic Logic*, 87(2):852–869, 2022.
- 37 Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for \mathbf{S}_2^1 and \mathbf{EF} . *Information and Computation*, 140(1):82–94, 1998.
- 38 Ian Mertz, Toniann Pitassi, and Yuanhao Wei. Short proofs are hard to find. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, 2019.
- 39 Daniele Micciancio. *The Geometry of Lattice Cryptography*, pages 185–210. Springer Berlin Heidelberg, 2011.
- 40 Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 700–718. Springer, 2012.
- 41 Theodoros Papamakarios. Depth d Frege systems are not automatable unless $\mathbf{P} = \mathbf{NP}$. *Electronic Colloquium on Computational Complexity (ECCC)*, 2023. URL: <https://eccc.weizmann.ac.il/report/2023/121/>.
- 42 Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016.
- 43 Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 187–196, 2008.
- 44 Ján Pich and Rahul Santhanam. Learning Algorithms Versus Automatability of Frege Systems. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229, pages 101:1–101:20, 2022.
- 45 Pavel Pudlák. Quantum deduction rules. *Annals of Pure and Applied Logic*, 157(1):16–29, 2009.
- 46 Pavel Pudlák. On reducibility and symmetry of disjoint \mathbf{NP} pairs. *Theoretical Computer Science*, 295(1):323–339, 2003. Mathematical Foundations of Computer Science.
- 47 Nael Rahman and Vladimir Shpilrain. MOBS (Matrices Over Bit strings) public key exchange, 2021. [arXiv:2106.01116](https://arxiv.org/abs/2106.01116).
- 48 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.
- 49 P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.
- 50 Michael Soltys and Stephen Cook. The proof complexity of linear algebra. *Annals of Pure and Applied Logic*, 130(1):277–323, 2004. Papers presented at the 2002 IEEE Symposium on Logic in Computer Science (LICS).

A The propositional translation for $\mathbf{LA}_{\mathbb{Q}}$

We sketch the construction for the propositional translation, and defer the details to the full version. The translation is analogous to the usual propositional translations used elsewhere in bounded arithmetic (see, for example, [20, 33]). Let φ be a formula of $\mathbf{LA}_{\mathbb{Q}}$ and let σ be an object assignment that assigns a natural number to each free index variable occurring in φ and to each term of the form $r(A)$ and $c(A)$ occurring in φ , and substitutes every function and predicate symbol by the corresponding \mathbf{TC}^0 circuit of the appropriate size. We denote by $\|\varphi\|_{\sigma}$ the propositional formula obtained by carrying out this translation process.

► **Theorem 28** (Propositional translation for $\mathbf{LA}_{\mathbb{Q}}$). *For every theorem φ of $\mathbf{LA}_{\mathbb{Q}}$ and every object assignment σ , the propositional formula $\|\varphi\|_{\sigma}$ admits polynomial-size \mathbf{TC}^0 -Frege proofs.*

For the proof, see Theorem A.2 in the full version of the paper [7].

B Axioms and basic theorems of LA

1. Equality axioms

- $x = x$
- $x = y \rightarrow y = x$
- $(x = y \wedge y = z) \rightarrow x = z$
- $\bigwedge_i^n (x_i = y_i) \rightarrow f(\bar{x}) = f(\bar{y})$
- $i_1 = j_1, i_2 = j_2, i_1 \leq i_2 \rightarrow j_1 \leq j_2$.

2. Axioms for indices

- $i + 0 = i$
- $i + (j + 1) = (i + j) + 1$
- $i \cdot (j + 1) = (i \cdot j) + i$
- $i + 1 = j + 1 \rightarrow i = j$
- $i + 1 \neq 0$
- $i \leq i + j$
- $i \leq j, j \leq i$
- $i \leq j, i + k = j \rightarrow (j - i = k)$
- $i \leq j, i + k = j \rightarrow (i \not\leq j \rightarrow j - i = 0)$
- $j \neq 0 \rightarrow \text{rem}(i, j) < j$
- $j \neq 0 \rightarrow i = j \cdot \text{div}(i, j) + \text{rem}(i, j)$
- $\alpha \rightarrow \text{cond}(\alpha, i, j) = i$
- $\neg\alpha \rightarrow \text{cond}(\alpha, i, j) = j$

3. Axioms for field elements

- $0 \neq 1 \wedge a + 0 = a$
- $a + (-a) = 0$
- $1 \cdot a = a$
- $a \neq 0 \rightarrow a \cdot (a^{-1}) = 1$
- $a + b = b + a$
- $a \cdot b = b \cdot a$
- $a + (b + c) = (a + b) + c$
- $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
- $a \cdot (b + c) = a \cdot b + a \cdot c$
- $\alpha \rightarrow \text{cond}(\alpha, a, b) = a$
- $\neg\alpha \rightarrow \text{cond}(\alpha, a, b) = b$

4. Axioms for matrices

- $(i = 0 \vee r(A) < i \vee j = 0 \vee c(A) < j) \rightarrow e(A, i, j) = 0$
- $r(A) = 1, c(A) = 1 \rightarrow \Sigma(A) = e(A, 1, 1)$
- $c(A) = 1 \rightarrow \sigma(A) = \sigma(A^\top)$
- $r(A) = 0 \vee c(A) = 0 \rightarrow \Sigma(A) = 0$

5. Theorems for ring properties

- $\max(i, j) = \max(j, i)$
- $\max(i, \max(j, k)) = \max(\max(i, j), k)$
- $\max(i, \max(j, k)) = \max(\max(i, j), \max(i, k))$
- $A + 0 = A$
- $A + (-1)A = 0$
- $AI = A$ and $IA = A$
- $A + B = B + A$
- $A + (B + C) = (A + B) + C$
- $A(BC) = (AB)C$
- $A(B + C) = AB + CA$
- $(B + C)A = BA + CA$
- $\Sigma 0 = 0_{\text{field}}$
- $\Sigma(cA) = c\Sigma(A)$
- $\Sigma(A + B) = \Sigma(A) + \Sigma(B)$
- $\Sigma(A) = \Sigma(A^\top)$

6. Theorems for module properties

- $(a + b)A = aA + bA$
- $a(A + B) = aA + aB$
- $(ab)A = a(bA)$

7. Theorems for inner product

- $A \cdot B = B \cdot A$
- $A \cdot (B + C) = A \cdot B + A \cdot C$
- $aA \cdot B = a(A \cdot B)$

8. Miscellaneous theorems

- $a(AB) = (aA)B \wedge (aA)B = A(aB)$
- $(AB)^\top = B^\top A^\top$
- $I^\top = I$
- $0^\top = 0$
- $(A^\top)^\top = A$

A Strong Direct Sum Theorem for Distributional Query Complexity

Guy Blanc

Department of Computer Science, Stanford University, CA, USA

Caleb Koch

Department of Computer Science, Stanford University, CA, USA

Carmen Strassle

Department of Computer Science, Stanford University, CA, USA

Li-Yang Tan

Department of Computer Science, Stanford University, CA, USA

Abstract

Consider the expected query complexity of computing the k -fold direct product $f^{\otimes k}$ of a function f to error ε with respect to a distribution μ^k . One strategy is to sequentially compute each of the k copies to error ε/k with respect to μ and apply the union bound. We prove a *strong direct sum theorem* showing that this naive strategy is essentially optimal. In particular, computing a direct product necessitates a blowup in both query complexity and error.

Strong direct sum theorems contrast with results that only show a blowup in query complexity or error but not both. There has been a long line of such results for distributional query complexity, dating back to (Impagliazzo, Raz, Wigderson 1994) and (Nisan, Rudich, Saks 1994), but a strong direct sum theorem that holds for all functions in the standard query model had been elusive.

A key idea in our work is the first use of the Hardcore Theorem (Impagliazzo 1995) in the context of query complexity. We prove a new *resilience lemma* that accompanies it, showing that the hardcore of $f^{\otimes k}$ is likely to remain dense under arbitrary partitions of the input space.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Query complexity, direct product theorem, hardcore theorem

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.16

Funding The authors are supported by NSF awards 1942123, 2211237, 2224246 and a Google Research Scholar award. Guy is also supported by a Jane Street Graduate Research Fellowship, Caleb by an NDSEG fellowship, and Carmen by a Stanford Computer Science Distinguished Fellowship.

1 Introduction

The *direct sum problem* seeks to understand the ways in which the complexity of solving k independent instances of a computational task scales with k . This problem and its variants such as the *XOR problem*, where one only seeks to compute the XOR of the k output values, have a long history in complexity theory. Research on them dates back to Strassen [33] and they have since been studied in all major computational models including boolean circuits [36, 26, 12, 14, 17, 29, 15, 11], communication protocols [16, 30, 22, 25, 35, 21, 31, 18, 20, 4, 8, 37], as well as classical [16, 27, 30, 22, 10, 5, 6, 9, 13] and quantum query complexity [2, 22, 32, 31, 1, 24].



© Guy Blanc, Caleb Koch, Carmen Strassle, and Li-Yang Tan;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 16; pp. 16:1–16:30

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



1.1 This work

We focus on classical query complexity, and specifically distributional query complexity. Distributional complexity, also known as average-case complexity, is a basic notion applicable to all models of computation. Direct sum theorems and XOR lemmas for distributional complexity, in addition to being statements of independent interest, have found applications in areas ranging from derandomization [36, 28, 17] to streaming [3] and property testing [7].

Let $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ be a boolean function, μ be a distribution over $\{\pm 1\}^n$, and consider the task of computing the k -fold direct product $f^{\otimes k}(X^{(1)}, \dots, X^{(k)}) := (f(X^{(1)}), \dots, f(X^{(k)}))$ of f to error ε with respect to μ^k . One strategy is to sequentially compute each $f(X^{(i)})$ to error ε/k with respect to μ and apply the union bound. Writing $\overline{\text{Depth}}^\mu(f, \varepsilon)$ to denote the minimum expected depth of any decision tree that computes f to error ε w.r.t. μ , this shows that:

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \leq k \cdot \overline{\text{Depth}}^\mu(f, \frac{\varepsilon}{k}).$$

Our main result is that this naive strategy is essentially optimal for all functions and distributions:

► **Theorem 1** (Strong direct sum theorem for distributional query complexity; special case of Theorem 2). *For every function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, integer $k \in \mathbb{N}$, and $\varepsilon < 1$,*

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \tilde{\Omega}(\varepsilon^2 k) \cdot \overline{\text{Depth}}^\mu(f, \Theta(\frac{\varepsilon}{k})).$$

Such direct sum theorems are termed *strong*, referring to the fact that they show that computing a direct product necessitates a blowup in *both* the computational resources of interest – in our case, query complexity – *and* error. Strong direct sum theorems contrast with standard ones, which only show a blowup in computational resource, and also with direct *product* theorems, which focus on the blowup in error. We give a detailed overview of prior work in Section 3, mentioning for now that while standard direct sum and direct product theorems for distributional query complexity have long been known, a strong direct sum theorem had been elusive. Prior to our work, it was even open whether $\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, 1.01\varepsilon) \geq 1.01 \cdot \overline{\text{Depth}}^\mu(f, \varepsilon)$ holds. Indeed, the problem is known to be quite subtle, as a striking counterexample of Shaltiel [30] shows that a strong direct sum theorem is badly false if one considers *worst-case* instead of expected query complexity.

A strong XOR lemma

We also obtain a strong XOR lemma (Theorem 3) as a corollary of a simple equivalence between direct sum theorems and XOR lemmas for query complexity. One direction is immediate, since the k -fold XOR $f^{\oplus k}(X^{(1)}, \dots, X^{(k)}) := f(X^{(1)}) \oplus \dots \oplus f(X^{(k)})$ can only be easier to compute than the k -fold direct product. For the query model the converse also holds: a direct sum theorem implies an XOR lemma with analogous parameters.

2 Broader context: Comparison with the randomized setting

Direct sum theorems are also well-studied in the setting of *randomized* query complexity. Recall that the ε -error randomized query complexity of f , denoted $\overline{R}(f, \varepsilon)$, is the minimum expected depth of any randomized decision tree that computes f with error at most ε for all

inputs. By Yao’s minimax principle, direct sum theorems for distributional query complexity imply analogous ones for randomized query complexity. However, as we now elaborate, such theorems are substantially more difficult to prove in the distributional setting.

2.1 A simple and near-optimal strong direct sum theorem for $\overline{\mathbf{R}}$

For randomized query complexity, proving a strong direct sum theorem with near-optimal parameters requires only two observations.

Observation #1

The first is that a standard direct sum theorem, one without error amplification, easily holds in the distributional setting, and hence the randomized setting as well by Yao’s principle:

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \Omega(k) \cdot \overline{\text{Depth}}^{\mu}(f, \varepsilon) \quad \text{and therefore} \quad \overline{\mathbf{R}}(f^{\otimes k}, \varepsilon) \geq \Omega(k) \cdot \overline{\mathbf{R}}(f, \varepsilon). \quad (1)$$

The idea is that given a decision tree T of average depth q that computes $f^{\otimes k}$ with error ε , one can extract a decision tree of average depth q/k that computes f to error ε : place the input in a random block $i \sim [k]$, fill the remaining blocks with independent random draws from μ , and return the i th bit of T ’s output. It is straightforward to show that this reduces the average depth of T by a factor of k while preserving its error.

Observation #2

The second observation is standard error reduction of randomized algorithms by repetition, which in particular implies:

$$\overline{\mathbf{R}}(f, \frac{\varepsilon}{k}) \leq O(\log k) \cdot \overline{\mathbf{R}}(f, \varepsilon). \quad (2)$$

Combining Equations (1) and (2) yields a strong direct sum theorem

$$\overline{\mathbf{R}}(f^{\otimes k}, \varepsilon) \geq \Omega\left(\frac{k}{\log k}\right) \cdot \overline{\mathbf{R}}(f, \frac{\varepsilon}{k})$$

that is within a $O(\log k)$ factor of optimal.

Blais–Brody

Using more sophisticated techniques, Blais and Brody [6] were recently able to remove this $O(\log k)$ factor and obtain an optimal strong direct sum theorem for $\overline{\mathbf{R}}$. Building on their work, Brody, Kim, Lerduptipongporn, and Srinivasulu [9] then obtained an optimal strong XOR lemma for $\overline{\mathbf{R}}$.

2.2 Error reduction fails in the distributional setting

While the crux of [6] and [9]’s works is the removal of a $O(\log k)$ factor, the situation is very different in the distributional setting. As mentioned, prior to our work even a direct sum theorem where both factor-of- $\tilde{\Omega}(k)$ blowups in Theorem 1 are replaced by 1.01 was not known to hold.

With regards to the argument above, it is Observation #2 that breaks in the distributional setting – not only does error reduction by repetition break, the distributional analogue of Equation (2) is simply false. This points to a fundamental difference between distributional

16:4 A Strong Direct Sum Theorem for Distributional Query Complexity

and randomized complexity: while generic error reduction of randomized algorithms is possible in all reasonable models of computation, the analogous statement for distributional complexity is badly false in all reasonable models of computation. For the query model specifically, in Appendix C we give an easy proof of the following:

► **Fact 1.** *For any $n \in \mathbb{N}$ and μ being the uniform distribution over $\{\pm 1\}^n$, there is a function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ such that $\overline{\text{Depth}}^\mu(f, \frac{1}{4}) = 0$ and yet $\overline{\text{Depth}}^\mu(f, \frac{1}{8}) \geq \Omega(n)$.*

2.3 A brief summary of our approach

We revisit Observation #1 and show how the very same extraction strategy can in fact yield a tree with error $\Theta(\varepsilon/k)$, instead of ε , at the expense of only a slight increase in depth. A key technical ingredient in our analysis is Impagliazzo’s *Hardcore Theorem* [14]. For intuition as to why this theorem may be relevant for us, we note that it is tightly connected to the notion of *boosting* from learning theory – they are, in some sense, dual to each other [23]. And boosting is, of course, a form of error reduction, albeit one that is more intricate than error reduction by repetition.

See Section 5 for a detailed overview of our approach, including a discussion of why Impagliazzo’s *Hardcore Theorem*, as is, does not suffice, thereby necessitating our new “resilience lemma” that accompanies it.

3 Prior Work

We now place Theorem 1 within the context of prior work on direct sum and product theorems for distributional query complexity. This is a fairly large body of work that dates back to the 1990s.

3.1 Standard direct sum and product theorems

Standard direct sum theorems

As we sketched in Section 2.1, a simple argument shows that

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \Omega(k) \cdot \overline{\text{Depth}}^\mu(f, \varepsilon). \quad (3)$$

This along with an application of Markov’s inequality yields:

$$\text{Depth}^{\mu^k}(f^{\otimes k}, \varepsilon - \varepsilon') \geq \Omega(\varepsilon'k) \cdot \text{Depth}^\mu(f, \varepsilon), \quad (4)$$

where $\text{Depth}^\mu(\cdot, \cdot)$ is the analogue of $\overline{\text{Depth}}^\mu(\cdot, \cdot)$ for worst-case instead of expected query complexity. (The details of these arguments are spelt out in [19, 5].)

Note that the error budget is the same on both sides of Equation (3) and the error budget on the RHS of Equation (4) is *larger* than that of the LHS. In a strong direct sum theorem one seeks a lower bound even when the error budget on the RHS is much smaller than that of the LHS, ideally by a multiplicative factor of k to match the naive upper bound.

A direct product theorem

Impagliazzo, Raz, and Wigderson [16] proved a direct product theorem which focuses on the blowup in error. They showed that:

$$\text{Depth}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \text{Depth}^\mu(f, \frac{\varepsilon}{k}). \quad (5)$$

While this result has the sought-for factor of k difference between the error budgets on the LHS and RHS, it comes at the price of there no longer being any blowup in depth.

3.2 Progress and barriers towards a strong direct sum theorem

These results naturally point to the problem of proving a unifying strong direct sum theorem. We now survey efforts at such a best-of-both-worlds result over the years.

Decision forests

Nisan, Rudich, and Saks [27] proved the following strengthening of [16]’s result. While [16] gives an upper bound on the success probability of a *single* depth- d decision tree for $f^{\otimes k}$, [27] showed that the same bound holds even for *decision forests* where one gets to construct a different depth- d tree for each of the k copies of f .

Since one can always stack the k many depth- d trees in a decision forest to obtain a single tree of depth kd , [27]’s result establishes a special case of a strong direct sum theorem under a structural assumption on the tree for $f^{\otimes k}$. See Figure 3 in Appendix A for an illustration of the stacked decision tree that one gets from a decision forest.

Fair decision trees

Building on the techniques of [27], Shaltiel [30] proved a strong direct sum theorem under a different structural assumption on the tree for $f^{\otimes k}$. He considered decision trees of depth kd that are “fair” in the sense that every path queries each of the k blocks of variables at most d times. ([30] actually proved a strong XOR lemma for fair decision trees, which implies a strong direct sum theorem for such trees.) See Figure 4 in Appendix A for an illustration of a fair decision tree.

Shaltiel’s counterexample for worst-case query complexity

These results of [27] and [30] could be viewed as evidence in favor of a general strong direct sum theorem, one that does not impose any structural assumptions on the tree for $f^{\otimes k}$. However, in the same paper Shaltiel also presented an illuminating example: he constructed a function, which we call *Shal*, and a distribution μ such that for all $k \in \mathbb{N}$,

$$\text{Depth}^{\mu^k}(\text{Shal}^{\otimes k}, \varepsilon) \leq O(\text{Depth}^{\mu}(\text{Shal}, \frac{\varepsilon}{k})). \quad (6)$$

This shows, surprisingly, that for *worst-case* query complexity, the factor-of- $\Omega(k)$ blowup in query complexity that one seeks in a strong direct sum theorem is not always necessary, and in fact sometimes even a constant factor suffices.

Shaltiel’s counterexample vs. Theorem 1

This counterexample for worst-case query complexity should be contrasted with our main result, Theorem 1, which shows that a strong direct sum theorem holds for *expected* query complexity. Indeed, the starting point of our work was the encouraging observation that Shaltiel’s function does in fact satisfy a strong direct sum theorem if one instead considers expected query complexity. That is, for any $\varepsilon < 1$ and sufficiently large k ,

$$\overline{\text{Depth}}^{\mu^k}(\text{Shal}^{\otimes k}, \varepsilon) \geq \Omega(k) \cdot \overline{\text{Depth}}^{\mu}(\text{Shal}, \frac{\varepsilon}{k}).$$

This is a simple observation but appears to have been overlooked. As we now overview, subsequent work considered other ways of sidestepping Shaltiel’s counterexample.

3.3 Results in light of Shaltiel's counterexample

A strong direct sum theorem for the OR function

Klauck, Špalek, and de Wolf [22] sidestepped Shaltiel's counterexample by considering a *specific* function (and distribution): motivated by applications to time-space tradeoffs, they proved a strong direct sum theorem for the OR function and with μ being its canonical hard distribution. Using this, they also showed, for all functions f a lower bound on $f^{\otimes k}$'s query complexity in terms of f 's block sensitivity. This stands in contrast to a strong direct sum theorem where one seeks a lower bound on $f^{\otimes k}$'s query complexity in terms of f 's query complexity.

A phase transition in Shaltiel's counterexample

The precise parameters of Shaltiel's counterexample are:

$$\text{Depth}^{\mu^k}(\text{Shal}^{\otimes k}, e^{-\Theta(\delta k)}) \leq C\delta k \cdot \text{Depth}^{\mu}(\text{Shal}, \delta)$$

for all sufficiently large constants C . Importantly, the multiplicative factor on the RHS is only δk instead of k , and therefore becomes a constant if the initial hardness parameter is $\delta = \varepsilon/k$ (thereby yielding Equation (6)).

Drucker [10] showed that there is a “phase transition” in Shaltiel's counterexample in the following sense: for all functions f and a sufficiently small constant $c > 0$,

$$\text{Depth}^{\mu^k}(f^{\otimes k}, 1 - e^{-\Theta(\delta k)}) \geq c\delta k \cdot \text{Depth}^{\mu}(f, \delta). \quad (7)$$

Therefore, while [30] showed the existence of a function Shal such that its k -fold direct product can be computed to surprisingly low error if the depth budget is $C\delta k \cdot \text{Depth}^{\mu}(f, \delta)$ for a sufficiently *large* constant C , [10] showed that for all functions f , this stops being the case if the depth budget is instead $c\delta k \cdot \text{Depth}^{\mu}(f, \delta)$ for a sufficiently *small* constant c .

Query complexity with aborts

Blais and Brody [6] showed that Shaltiel's counterexample can be sidestepped in a different way. En route to proving their strong direct sum theorem for randomized query complexity (discussed in Section 2), they considered decision trees $T : \{\pm 1\}^n \rightarrow \{\pm 1, \perp\}$ that are allowed to output \perp (“abort”) on certain inputs, and where the error of T in computing a function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ is measured with respect to $T^{-1}(\{\pm 1\})$. In other words, T 's output on x is considered correct if $T(x) = \perp$.

Writing $\text{Depth}_{\text{Pr}[\perp] \leq \frac{1}{3}}^{\mu}(f, \varepsilon)$ to denote the minimum depth of any decision tree for f that aborts with probability at most $1/3$ and otherwise errs with probability at most ε (both w.r.t. μ), [6] proved that

$$\text{Depth}_{\text{Pr}[\perp] \leq \frac{1}{3}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \Omega(k) \cdot \text{Depth}_{\text{Pr}[\perp] \leq \frac{1}{3}}^{\mu}(f, \frac{\varepsilon}{k}). \quad (8)$$

Even though the error budget on non-aborts is only ε/k on the RHS, the fact that the tree is allowed to abort with probability $1/3$ means that it is deemed correct on a $1/3$ fraction of inputs “for free”. A decision tree that aborts with probability $1/3$ and otherwise errs with probability ε/k can therefore be much smaller than one that never aborts and errs with probability ε/k , and indeed, it is easy to construct examples witnessing the maximally large separation:

$$n = \text{Depth}^{\mu}(f, \frac{\varepsilon}{k}) \gg \text{Depth}_{\text{Pr}[\perp] \leq \frac{1}{3}}^{\mu}(f, \frac{\varepsilon}{k}) = 1.$$

Building on [6], Brody, Kim, Lerdpattipongporn, and Srinivasulu [9] proved a strong XOR lemma for this model of query complexity with aborts, achieving analogous parameters.

A strong XOR lemma assuming hardness against all depths

A standard strong XOR lemma states that if f is hard against decision trees of certain fixed depth d , then $f^{\otimes k}$ is much harder against decision trees of depth $\Omega(dk)$. Recent work of Hoza [13] shows that Shaltiel’s counterexample can be sidestepped if one allows for the stronger assumption that f ’s hardness “scales nicely” with d . (See the paper for the precise statement of the resulting strong XOR lemma.)

3.4 Summary

Summarizing, prior work on direct sum and product theorems for distributional query complexity either: focused on the blowup in error [16] or query complexity [19, 5] but not both; considered restrictions (fair decision trees [30]) or variants (decision forests [27]; allowing for aborts [6, 9]) of the query model; focused on specific functions (the OR function [22]); or imposed additional hardness assumptions about the function [13]. Theorem 1, on the other hand, gives a strong direct sum theorem that holds for all functions in the standard query model. See Table 1.

■ **Table 1** Direct sum and product theorems for distributional query complexity.

Reference	Error Amplification	Query Amplification	Query model/ Assumption
[19, 5]	×	✓	Standard query model
[16]	✓	×	Standard query model
[27]	✓	✓	Decision forests
[30]	✓	✓	Fair decision trees
[22]	✓	✓	$f = \text{OR}$
[6, 9]	✓	✓	Decision trees with aborts
[13]	✓	✓	Hardness against all depths
Theorem 1	✓	✓	Standard query model

4 Formal statements of our results and their tightness

Theorem 1 is a special case of the following result:

► **Theorem 2** (Strong direct sum theorem). *For every function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, $k \in \mathbb{N}$, and $\gamma, \delta \in (0, 1)$, we have that*

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, 1 - e^{-\Theta(\delta k)} - \gamma) \geq \Omega\left(\frac{\gamma^2 k}{\log(1/\delta)}\right) \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

16:8 A Strong Direct Sum Theorem for Distributional Query Complexity

We in fact prove a strong *threshold* direct sum theorem which further generalizes Theorem 2: while a direct sum theorem shows that $f^{\otimes k}$ is hard to compute, i.e. it is hard to get *all* k copies of f correct, a threshold direct sum theorem shows that it is hard even to get *most* of the k copies of f correct. See Theorem 22.

By the equivalence between strong direct sum theorems and strong XOR lemmas (Claim 35), we also get:

► **Theorem 3** (Strong XOR lemma). *For every function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ and distribution μ over $\{\pm 1\}^n$, $k \in \mathbb{N}$, and $\gamma, \delta \in (0, 1)$, we have that*

$$\overline{\text{Depth}}^{\mu^k} \left(f^{\oplus k}, \frac{1}{2}(1 - e^{-\Theta(\delta k)} - \gamma) \right) \geq \Omega \left(\frac{\gamma^2 k}{\log(1/\delta)} \right) \cdot \overline{\text{Depth}}^\mu(f, \delta).$$

4.1 Tightness

Theorem 2 amplifies an initial hardness parameter of $\delta = \Theta(1/k)$ to $1 - \gamma$ for any small constant γ with a near-optimal overhead of

$$\Omega \left(\frac{\gamma^2 k}{\log(1/\delta)} \right) = \Omega \left(\frac{k}{\log k} \right).$$

However, due to the polynomial dependence on γ , we cannot achieve a final hardness parameter that is exponentially close to 1 as a function of k . We show that this is unavoidable since at least a linear dependence on γ is necessary:

▷ **Claim 4** (Linear dependence on γ is necessary). Let $\text{Par} : \{\pm 1\}^n \rightarrow \{\pm 1\}$ be the parity function and μ be the uniform distribution over $\{\pm 1\}^n$. Then for all γ ,

$$\overline{\text{Depth}}^{\mu^k}(\text{Par}^{\otimes k}, 1 - \gamma) \leq O(\gamma k) \cdot \overline{\text{Depth}}^\mu(\text{Par}, \frac{1}{4}).$$

The same example shows that a linear dependence on γ is likewise necessary in the setting of XOR lemmas. Determining the optimal polynomial dependence on γ in both settings, as well as the necessity of the $\log(1/\delta)$ factor, are concrete avenues for future work.

5 Technical Overview for Theorem 2

5.1 Hardcore measures and the Hardcore Theorem

At the heart of our proof is the notion of a *hardcore measure* and *Impagliazzo's Hardcore Theorem* [14], both adapted to the setting of query complexity.

► **Definition 5** (Hardcore measure for query complexity). *We say that $H : \{\pm 1\}^n \rightarrow [0, 1]$ is a (γ, d) -hardcore measure for $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ w.r.t. μ of density δ if:*

1. H 's density is δ : $\mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})] = \delta$.
2. d -query algorithms achieve correlation at most γ with f on H :

$$\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})T(\mathbf{x})H(\mathbf{x})] \leq \gamma \mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})] = \gamma\delta.$$

for all decision trees T whose expected depth w.r.t. μ is at most d .

► **Theorem 6** (Hardcore Theorem for query complexity). *For every function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, and $\gamma, \delta > 0$, there exists a (γ, d) -hardcore measure H for f of density $\delta/2$ w.r.t. μ where*

$$d = \Theta \left(\frac{\gamma^2}{\log(1/\delta)} \right) \overline{\text{Depth}}^\mu(f, \delta).$$

The Hardcore Theorem was originally proved, and remains most commonly used, in the setting of circuit complexity where it has long been recognized as a powerful result. (See e.g. [34], where it is described as “one of the bits of magic of complexity theory”.) We show in Appendix B that its proof extends readily to the setting of query complexity to establish Theorem 6. Despite its importance in circuit complexity and its straightforward extension to query complexity, our work appears to be the first to consider its applicability in the latter setting.

► **Remark 7.** For intuition regarding Definition 5, note that if $H : \{\pm 1\}^n \rightarrow \{0, 1\}$ is the indicator of a set, the two properties simplify to: $\Pr_{\mathbf{x} \sim \mu} [\mathbf{x} \in H] = \delta$ and $\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})T(\mathbf{x}) \mid \mathbf{x} \in H] \leq \gamma$.

5.2 Two key quantities: hardcore density and hardcore advantage at a leaf

Setup

For the remainder of this section, we fix a function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, and initial hardness parameter δ (which we think of as small, close to 0). Let $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ be a decision tree that seeks to compute $f^{\otimes k}$ w.r.t. μ^k . Our goal is to show that T 's error must be large, close to 1, unless its depth is sufficient large.

Definitions of the hardcore density and hardcore advantage at a leaf

Let $H : \{\pm 1\}^n \rightarrow [0, 1]$ be a (γ, d) -hardcore measure for f of density δ w.r.t. μ given by Theorem 6. Each leaf ℓ of T corresponds to a tuple of restrictions (π_1, \dots, π_k) to each of the k blocks of inputs. We will be interested in understanding, for a random block $i \in [k]$, the extent to which the restricted function H_{π_i} retains the two defining properties of a hardcore measure: high density and strong hardness. We therefore define:

► **Definition 8** (Hardcore density at ℓ). For $i \in [k]$, the hardcore density at ℓ in the i th block is the quantity:

$$\text{Dens}_H(\ell, i) := \mathbb{E}_{\mathbf{X} \sim \mu^k} [H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell].$$

The total hardcore density at ℓ is the quantity $\text{Dens}_H(\ell) := \sum_{i=1}^k \text{Dens}_H(\ell, i)$.

See Figure 1 for an illustration of Definition 8.

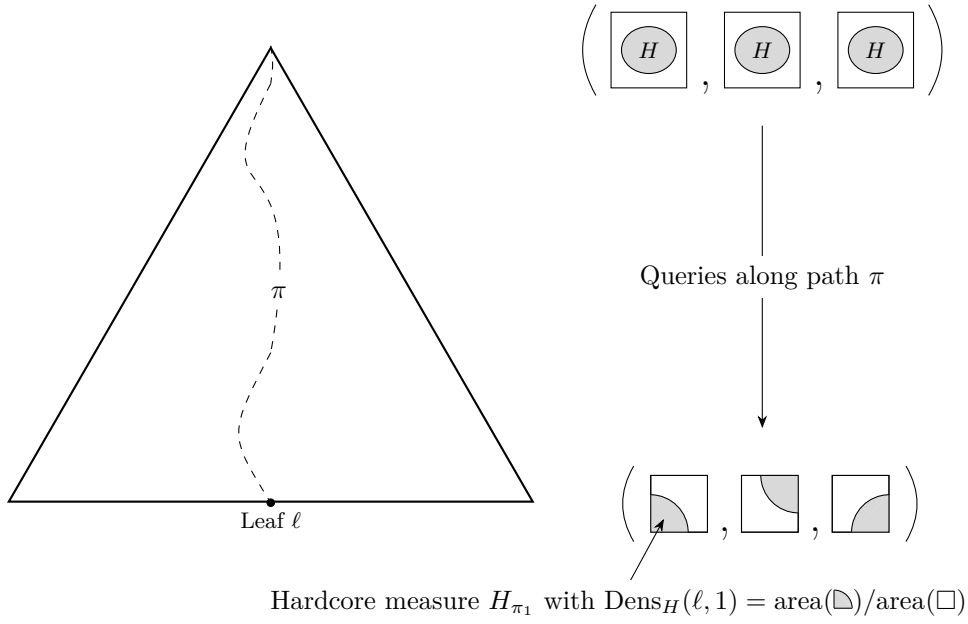
► **Definition 9** (Hardcore advantage at ℓ). For $i \in [k]$, the hardcore advantage at ℓ in the i th block is the quantity:

$$\text{Adv}_H(\ell, i) := \left| \mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)})T(\mathbf{X})_i H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right|.$$

The total hardcore advantage at ℓ is the quantity $\text{Adv}_H(\ell) := \sum_{i=1}^k \text{Adv}_H(\ell, i)$.

Intuitively, leaves for which $\text{Dens}_H(\ell)$ is large and $\text{Adv}_H(\ell)$ is small contribute significantly to error of T . Lemma 10 below formalizes this:

16:10 A Strong Direct Sum Theorem for Distributional Query Complexity



■ **Figure 1** Illustration of a hardcore density. The tree $T : (\{\pm 1\}^n)^3 \rightarrow \{\pm 1\}^3$ seeks to compute a function $f^{\otimes 3}$. The tuple of squares at the top of the figure illustrates the set of all inputs to the function while the strings in the support of the hardcore measure are shaded gray. The tuple at the bottom of the figure illustrates the set of inputs reaching the leaf ℓ . Each block is the subcube consistent with the path π and the shaded region denotes the fragment of H which is contained in the corresponding subcube.

Notation

Canonical distribution over leaves. We write $\mu^k(T)$ to denote the distribution over leaves of T where:

$$\Pr_{\ell \sim \mu^k(T)}[\ell = \ell] = \Pr_{\mathbf{X} \sim \mu^k}[\mathbf{X} \text{ reaches } \ell].$$

► **Lemma 10** (Accuracy in terms of hardcore density and advantage at leaves).

$$\Pr_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) = f^{\otimes k}(\mathbf{X})] \leq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\exp \left(- \frac{\text{Dens}_H(\ell) - \text{Adv}_H(\ell)}{4} \right) \right].$$

5.3 Expected total hardcore density and advantage

Lemma 10 motivates understanding the random variables $\text{Dens}_H(\ell)$ and $\text{Adv}_H(\ell)$ for $\ell \sim \mu^k(T)$. We begin by bounding their expectations:

▷ **Claim 11** (Expected total hardcore density). If H is a hardcore measure of density δ then

$$\mathbb{E}_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell)] = \delta k.$$

Claim 11 is a statement about density preservation. It says that H 's expected density at a random leaf $\ell \sim \mu^k(T)$ and in a random block $i \sim [k]$ is equal to H 's initial density:

$$\mathbb{E}_{\substack{\ell \sim \mu^k(T) \\ i \sim [k]}} \left[\underbrace{\mathbb{E}_{\mathbf{X} \sim \mu^k} [H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell]}_{\text{Dens}_H(\ell, i)} \right] = \delta = \mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{X})].$$

▷ **Claim 12 (Expected total hardcore advantage).** If H is a (γ, d) -hardcore measure for f of density δ w.r.t. μ and the expected depth of T is at most dk , then

$$\mathbb{E}_{\ell \sim \mu^k(T)} [\text{Adv}_H(\ell)] \leq \gamma \mathbb{E}_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell)].$$

Claim 12 is a statement about depth amplification. By definition, H being a (γ, d) -hardcore measure for f means that

$$\underbrace{\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x}) T_{\text{small}}(\mathbf{x}) H(\mathbf{x})]}_{\text{Hardcore advantage}} \leq \gamma \underbrace{\mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})]}_{\text{Hardcore density}}$$

for every tree $T_{\text{small}} : \{\pm 1\}^n \rightarrow \{\pm 1\}$ of expected depth d . Claim 12 says that

$$\underbrace{\sum_{i=1}^k \mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)}) T_{\text{large}}(\mathbf{X})_i H(\mathbf{X}^{(i)})]}_{\text{Total hardcore advantage}} \leq \gamma \underbrace{\sum_{i=1}^k \mathbb{E}_{\mathbf{X} \sim \mu^k} [H(\mathbf{X}^{(i)})]}_{\text{Total hardcore density}}$$

for every tree $T_{\text{large}} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ of expected depth dk . Crucially, the depth of T_{large} is allowed to be a factor of k larger than that of T_{small} , and yet the ratio of hardcore advantage to hardcore density remains the same (γ in both cases).

5.3.1 Done if Jensen went the other way

For intuition as to why Claim 11 and Claim 12 are relevant yet insufficient for us, note that if it were the case that $\mathbb{E}[\exp(-Z)] \leq \exp(-\mathbb{E}[Z])$, which unfortunately is the opposite of what Jensen's inequality gives, we would have the strong bound on the accuracy of T that we seek:

$$\begin{aligned} \Pr_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) = f^{\otimes k}(\mathbf{X})] &\leq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\exp \left(- \frac{\text{Dens}_H(\ell) - \text{Adv}_H(\ell)}{4} \right) \right] && \text{(Lemma 10)} \\ &\stackrel{\text{“} \leq \text{”}}{\leq} \exp \left(- \mathbb{E}_{\ell \sim \mu^k(T)} \left[\frac{\text{Dens}_H(\ell) - \text{Adv}_H(\ell)}{4} \right] \right) && \text{(Wrong direction of Jensen)} \\ &\leq \exp \left(- \frac{\delta k - \gamma \delta k}{4} \right) && \text{(Claim 11 and Claim 12)} \\ &\leq \exp(-\Theta(\delta k)). \end{aligned}$$

For an actual proof, we need to develop a more refined understanding of the distribution of $\text{Dens}_H(\ell)$ beyond just its expectation. (As it turns out, this along with the bound on $\mathbb{E}[\text{Adv}_H(\ell)]$ given by Claim 12 suffices.)

5.4 A resilience lemma for hardcore measures

An illustrative bad case to rule out

Suppose T were such that it achieved $\mathbb{E}[\text{Dens}_H(\ell)] = \delta k$ by having a δ -fraction of leaves with $\text{Dens}_H(\ell) = k$ and the remaining $1 - \delta$ fraction with $\text{Dens}_H(\ell) = 0$. If this were the case then “all the hardness” would be concentrated on a small δ fraction of leaves, and the best lower bound that we would be able to guarantee on error of T with respect to $f^{\otimes k}$ would only be δ . This is our starting assumption on the hardness of f , and so no error amplification has occurred.

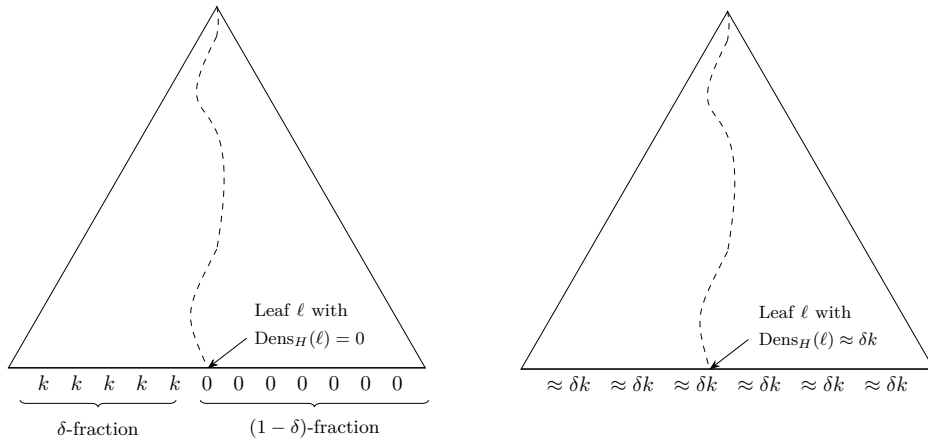
The resilience lemma

We rule out cases like this by showing that T must achieve $\mathbb{E}[\text{Dens}_H(\ell)] = \delta k$ by having the vast majority of its leaves with $\text{Dens}_H(\ell) = \Omega(\delta k)$, i.e. that $\text{Dens}_H(\ell)$ is tightly concentrated around its expectation:

► **Lemma 13** (Resilience lemma). *For any hardcore measure H of density δ w.r.t. μ and tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$,*

$$\Pr_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell) \leq \delta k / 2] \leq e^{-\delta k / 8}.$$

Similarly, $\Pr_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell) \geq 2\delta k] \leq e^{-\delta k / 3}.$



(a) An illustration of the bad case where Dens_H is anti-concentrated away from its mean of δk . (b) An illustration of the good case where Dens_H is concentrated around its mean of δk .

■ **Figure 2** An illustration of our resilience lemma (Lemma 13). This lemma shows that all trees resemble the one on the right, with $\text{Dens}_H(\ell)$ tightly concentrated around its mean of δk . This allows us to rule out bad trees such as those on the left where all of the hardness is concentrated on a small fraction of the leaves.

Comparing Lemma 13 to Claim 12, we see that Claim 12 is a statement about density preservation *in expectation* whereas Lemma 13 is a statement about density preservation *with high probability*. It says that H 's density at a random leaf $\ell \sim \mu^k(T)$ and in a random

block $i \sim [k]$ remains, with high probability, roughly the same as that of H 's initial density – this is why we call Lemma 13 a resilience lemma. (Our proof of Lemma 13 in fact shows that the H 's density remains resilient under arbitrary partitions of $(\{\pm 1\}^n)^k$, not just those induced by a decision tree.)

With Lemma 13 in hand, the intuition sketched in Section 5.3.1 can be made formal.

6 Discussion and Future Work

Our main results are a strong direct sum theorem and a strong XOR lemma for distributional query complexity, showing that if f is somewhat hard to approximate with depth- d decision trees, then $f^{\otimes k}$ and $f^{\oplus k}$ are both *much harder* to approximate, even with decision trees of *much larger* depth. These results hold for expected query complexity, and they circumvent a counterexample of Shaltiel showing that such statements are badly false for worst-case query complexity. We view our work as confirming a remark Shaltiel made in his paper, that his counterexample “seems to exploit defects in the formulation of the problem rather than show that our general intuition for direct product assertions is false.”

Shaltiel's counterexample applies to many other models including boolean circuits and communication protocols. A broad avenue for future work is to understand how this counterexample can be similarly circumvented in these models by working with more fine-grained notions of computation cost. Consider for example boolean circuit complexity and Yao's XOR lemma [36], which states that if f is mildly hard to approximate with size- s circuits w.r.t. μ , then $f^{\oplus k}$ is extremely hard to approximate with size- s' circuits w.r.t. μ^k . A well-known downside of this important result is that it only holds for $s' \ll s$. Indeed, Shaltiel's counterexample shows that it cannot hold for $s' \gg s$, at least not for the standard notion of circuit size. Extrapolating from our work, can we prove a strong XOR lemma for boolean circuits by considering a notion of the “expected size” of a circuit $C : \{\pm 1\}^n \rightarrow \{\pm 1\}$ with respect to a distribution μ over $\{\pm 1\}^n$? A natural approach is to consider the standard notion of the expected runtime of a Turing machine with respect to a distribution over inputs and have the Cook–Levin theorem guide us towards an appropriate analogue for circuit size.

On a more technical level, a crucial ingredient in our work is the first use of Impagliazzo's Hardcore Theorem within the context of query complexity (and indeed, to our knowledge, the first use of it outside of circuit complexity). Could this powerful theorem be useful for other problems in query complexity, possibly when used in conjunction with our new resilience lemma?

7 Preliminaries

We use $[n]$ to denote the set $\{1, 2, \dots, n\}$ and **bold font** (e.g. $\mathbf{x} \sim \mathcal{D}$) to denote random variables. For any distribution μ , we use μ^k to denote k -fold the product distribution $\mu \times \dots \times \mu$.

For any function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, we use $f^{\otimes k} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ to denote its k -fold direct product,

$$f^{\otimes k}(X^{(1)}, \dots, X^{(k)}) := (f(X^{(1)}), \dots, f(X^{(k)})).$$

Similarly, we use $f^{\oplus k} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}$ to denote its k -fold direct sum,

$$f^{\oplus k}(X^{(1)}, \dots, X^{(k)}) := \prod_{i \in [k]} f(X^{(i)}).$$

16:14 A Strong Direct Sum Theorem for Distributional Query Complexity

► **Definition 14** (Bernoulli distribution). For any $\delta \in [0, 1]$, we write $\text{Ber}(\delta)$ to denote the distribution of z where $z = 1$ with probability δ and 0 otherwise.

► **Definition 15** (Binomial distribution). For any $k \in \mathbb{N}$, $\delta \in [0, 1]$, we write $\text{Bin}(k, \delta)$ to denote the sum of k independent random variables drawn from $\text{Ber}(\delta)$.

► **Fact 2** (Chernoff bound). Let z_1, \dots, z_k be independent and each bounded within $[0, 1]$ and $Z := \sum_{i \in [k]} z_i$. For any threshold $t \leq \mu := \mathbb{E}[Z]$,

$$\Pr[Z \leq t] \leq \exp\left(-\frac{(\mu-t)^2}{2\mu}\right).$$

Similar bounds for the probability Z exceeds its mean hold. For example,

$$\Pr[Z \geq 2\mu] \leq \exp\left(-\frac{\mu}{3}\right).$$

Furthermore, the above bounds also hold for any random variable Y satisfying $\mathbb{E}[e^{\lambda Y}] \leq \mathbb{E}[e^{\lambda Z}]$ for all $\lambda \in \mathbb{R}$.

7.1 Randomized vs. deterministic decision trees

We will prove all of our results with respect to the expected depth of a *randomized* decision tree. In this subsection, we formally define *deterministic* and *randomized* decision trees and prove that our results easily extend to the deterministic setting.

► **Definition 16** (Deterministic decision tree). A *deterministic decision tree*, $T : \{\pm 1\}^n \rightarrow \{\pm 1\}$, is a binary tree with two types of nodes: Internal nodes each query some x_i for $i \in [n]$ and have two children whereas leaf nodes are labeled by a bit $b \in \{\pm 1\}$ and have no children. On input $x \in \{\pm 1\}^n$, $T(x)$ is computed as follows: We proceed through T starting at the root. Whenever at an internal node that queries the i^{th} coordinate, we proceed to the left child if $x_i = -1$ and right child if $x_i = +1$. Once we reach a leaf, we output the label of that leaf.

► **Definition 17** (Randomized decision tree). A *randomized decision tree*, $\mathcal{T} : \{\pm 1\}^n \rightarrow \{\pm 1\}$, is distribution over deterministic decision trees. On input $x \in \{\pm 1\}^n$, it first draws $\mathbf{T} \sim \mathcal{T}$ and then outputs $\mathbf{T}(x)$.

► **Definition 18** (Expected depth). For any deterministic decision tree $T : \{\pm 1\}^n \rightarrow \{\pm 1\}$ and distribution μ on $\{\pm 1\}^n$, we use $\overline{\text{Depth}}^\mu(T)$ to denote the expected depth of T , which is the expected number of coordinates T queries on a random input $\mathbf{x} \sim \mu$. Similarly, for a randomized decision tree \mathcal{T} , $\overline{\text{Depth}}^\mu(\mathcal{T}) := \mathbb{E}_{\mathbf{T} \sim \mathcal{T}}[\overline{\text{Depth}}^\mu(\mathbf{T})]$.

We write $\overline{\text{Depth}}^\mu(\cdot, \cdot)$ to denote the minimum expected depth of *any* decision tree, including randomized decision trees. Thus, all of our main results, as written, hold for *randomized* decision trees; however, equivalent statements are true if we restrict ourselves to only *deterministic* decision trees, with only a small change in constants, as easily seen from the following claim.

▷ **Claim 19.** For any $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ , and constant ε ,

$$\overline{\text{Depth}}^\mu(f, 2\varepsilon) \leq \overline{\text{Depth}}_{\text{det}}^\mu(f, 2\varepsilon) \leq 2 \cdot \overline{\text{Depth}}^\mu(f, \varepsilon).$$

Proof. The left-most inequality follows immediately from that fact that any deterministic decision tree is also a randomized decision tree. For the second inequality, given any randomized decision tree \mathcal{T} with error ε and expected depth d , we'll construct a deterministic

decision tree T with error at most 2ε and expected depth at most $2d$. First, we decompose the expected error of \mathcal{T} :

$$\varepsilon = \Pr_{\mathbf{x} \sim \mu} [\mathcal{T}(\mathbf{x}) \neq f(\mathbf{x})] = \mathbb{E}_{\mathbf{T} \sim \mathcal{T}} \left[\Pr_{\mathbf{x} \sim \mu} [\mathbf{T}(\mathbf{x}) \neq f(\mathbf{x})] \right].$$

Applying Markov's inequality, if we sample $\mathbf{T} \sim \mathcal{T}$, with probability at least $1/2$, it has error at most ε . Similarly, since the expected depth of \mathcal{T} is d , with probability at least $1/2$, \mathbf{T} will have expected depth at most $2d$. By union bound, there is a nonzero probability that we choose a single (deterministic) tree with error at most 2ε and expected depth at most $2d$. \triangleleft

Claim 19 immediately allows direct sum theorems for *randomized* decision trees to also apply to *deterministic* decision trees.

► **Corollary 20** (Randomized direct sum theorems imply deterministic ones). *Suppose that a randomized direct sum theorem of the following form holds. For a function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, $k \in \mathbb{N}$, and constants ε, δ, M ,*

$$\overline{\text{Depth}}^{\mu^{\otimes k}}(f^{\otimes k}, \varepsilon) \geq M \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

Then,

$$\overline{\text{Depth}}_{\text{det}}^{\mu^{\otimes k}}(f^{\otimes k}, \varepsilon) \geq \overline{\text{Depth}}^{\mu^{\otimes k}}(f^{\otimes k}, \varepsilon) \geq M \cdot \overline{\text{Depth}}^{\mu}(f, \delta) \geq \frac{M}{2} \cdot \overline{\text{Depth}}_{\text{det}}^{\mu}(f, 2\delta).$$

With Corollary 20 in mind, the remainder of this paper will only consider *randomized* decision trees.

8 Proof of Theorem 2

The purpose of this section is to prove our direct sum theorem (Theorem 2) showing that if f is hard to compute, then $f^{\otimes k}$ is even harder to compute. We will in fact prove a *threshold* direct sum theorem, showing that it is hard even to get *most* of the k copies correct. To formalize this, we generalize the notation $\overline{\text{Depth}}(\cdot, \cdot)$ to take in an additional threshold parameter t specifying how many blocks we allow to be wrong.

► **Definition 21.** *For any function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, error ε , and threshold $t \in \mathbb{N}$, we use $\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon, t)$ to denote the minimum expected depth of a tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ satisfying*

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}) - f^{\otimes k}(\mathbf{X})\|_0 > t] \leq \varepsilon.$$

► **Theorem 22** (A strong threshold direct sum theorem for query complexity). *For every function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, $k \in \mathbb{N}$, and $\gamma, \delta \in (0, 1)$,*

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, 1 - e^{-\Omega(\delta k)} - \gamma, \Omega(\delta k)) \geq \Omega\left(\frac{\gamma^2 k}{\log(1/\delta)}\right) \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

Note that the threshold of $\Omega(\delta k)$ is within a constant factor of optimal, as repeating an algorithm that errs δ fraction of the time k times will lead to an average of δk mistakes. Theorem 22 implies our standard strong direct sum theorem (Theorem 2) because

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon, t) \geq \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon, 0) = \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon)$$

for any $t \geq 0$ and $\varepsilon > 0$.

8.1 The structure of this section

By the Hardcore Theorem (Theorem 6), proving Theorem 22 reduces to proving:

► **Theorem 23** (Hardness of $f^{\otimes k}$ in terms of a hardcore measure for f). *Suppose that $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ has an (γ, d) -hardcore measure w.r.t. μ of density δ . Then, for any $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ with $\overline{\text{Depth}}^{\mu^k}(T) \leq kd$,*

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}) - f^{\otimes k}(\mathbf{X})\|_0 \leq \frac{\delta k}{10}] \leq e^{-\frac{\delta k}{10}} + 10\gamma.$$

This section is therefore devoted to proving Theorem 23. As discussed in Section 5, our proof tracks two key quantities: we will analyze how *hardcore density* (Definition 8) and *hardcore advantage* (Definition 9) are distributed over the leaves of T . This proof will be broken into three steps:

1. In Section 8.2, we prove Claim 12 and Lemma 13, which aim to understand the distributions of the hardcore density and hardcore advantage of a random leaf of T .
2. In Section 8.3, we derive an expression for the probability T makes surprisingly few mistakes as a function of the hardcore density and hardcore advantage at each leaf. This generalizes Lemma 10.
3. In Section 8.4, we combine the above to prove Theorem 23.

8.2 How hardcore density and advantage distribute over the leaves

We begin with proving our resilience lemma for hardcore density. Roughly speaking, this will say that for any tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$, the hardcore density of a random leaf concentrates around δk . We recall the definition of hardcore density.

► **Definition 24** (Hardcore density at ℓ , Definition 8 restated). *For any tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$, hardcore measure $H : \{\pm 1\}^n \rightarrow [0, 1]$, distribution μ on $\{\pm 1\}^n$, $i \in [k]$, and leaf ℓ of T , the hardcore density at ℓ in the i th block is the quantity:*

$$\text{Dens}_H(\ell, i) := \mathbb{E}_{\mathbf{X} \sim \mu^k} [H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell].$$

The total hardcore density at ℓ is the quantity $\text{Dens}_H(\ell) := \sum_{i=1}^k \text{Dens}_H(\ell, i)$.

The distribution over leaves in the resilience lemma is the canonical distribution.

► **Definition 25** (Canonical distribution). *For any tree T and distribution μ over T 's domain, we write $\mu(T)$ to denote the distribution over leaves of T where:*

$$\Pr_{\ell \sim \mu^k(T)} [\ell = \ell] = \Pr_{\mathbf{X} \sim \mu} [\mathbf{X} \text{ reaches } \ell].$$

► **Lemma 26** (Resilience lemma, generalization of Lemma 13). *For any $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$, hardcore measure H , distribution μ , and convex $\Phi : \mathbb{R} \rightarrow \mathbb{R}$,*

$$\mathbb{E}_{\ell \sim \mu^k(T)} [\Phi(\text{Dens}_H(\ell))] \leq \mathbb{E}_{\mathbf{z} \sim \text{Bin}(k, \delta)} [\Phi(\mathbf{z})]$$

where $\delta := \mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})]$ is the density of H w.r.t. μ .

Proof. Draw $\mathbf{X} \sim \mu^k$. Then, for each $i \in [k]$, independently draw $z_i \sim \text{Ber}(H(\mathbf{X}^{(i)}))$. Note that, for any leaf ℓ and $i \in [k]$,

$$\text{Dens}_H(\ell, i) := \mathbb{E}_{\mathbf{X} \sim \mu^k} [H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] = \mathbb{E}_{\mathbf{X} \sim \mu^k} [z_i \mid \mathbf{X} \text{ reaches } \ell].$$

By the above equality and definition $\text{Dens}_H(\ell) = \sum_{i \in [k]} \text{Dens}_H(\ell, i)$,

$$\begin{aligned} \mathbb{E}_{\ell \sim \mu^k(T)} [\Phi(\text{Dens}_H(\ell))] &= \mathbb{E}_{\ell \sim \mu^k(T)} \left[\Phi \left(\mathbb{E}_{\mathbf{X} \sim \mu^k} \left[\sum_{i \in [k]} z_i \mid \mathbf{X} \text{ reaches } \ell \right] \right) \right] \\ &\leq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\mathbb{E}_{\mathbf{X} \sim \mu^k} \left[\Phi \left(\sum_{i \in [k]} z_i \mid \mathbf{X} \text{ reaches } \ell \right) \right] \right] \\ &\hspace{20em} \text{(Jensen's inequality)} \\ &= \mathbb{E}_{\mathbf{X} \sim \mu^k} \left[\Phi \left(\sum_{i \in [k]} z_i \right) \right]. \hspace{5em} \text{(Law of total expectation)} \end{aligned}$$

Note that the last line holds precisely for the distribution $\mu^k(T)$ defined in Definition 25, which is why we use that distribution.

Since \mathbf{X} is drawn from a product distribution, and z_i depends on only the i^{th} coordinate of \mathbf{X} , z_1, \dots, z_k are independent. Furthermore, each has mean $\mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})] = \delta$. Therefore, $\sum_{i \in [k]} z_i$ is distributed according to $\text{Bin}(k, \delta)$. ◀

A couple of remarks about the above Lemma: First, it implies that $\text{Dens}_H(\ell)$ concentrates around δk . Since $z \mapsto e^{\lambda z}$ is convex for any $\lambda \in \mathbb{R}$, Lemma 26 implies that the moment generating function of $\text{Dens}_H(\ell)$ is dominated by that of $\text{Bin}(k, \delta)$. This means that Chernoff bounds that hold for $\text{Bin}(k, \delta)$ also hold for $\text{Dens}_H(\ell)$. In particular, the statement of Lemma 13 is a consequence of the Chernoff bound given in Fact 2.

Second, the proof of Lemma 26 does not make heavy use of the decision tree structure of T . It only uses that the leaves of T partition $(\{\pm 1\}^n)^k$, and so may find uses for other models that partition the domain.

Depth amplification for hardcore advantage

While the resilience lemma gives a fairly fine-grained understanding of how hardcore density distributes among the leaves, our guarantee for hardcore advantage are more coarse – that its expectation over the leaves is bounded.

► **Definition 27** (Hardcore advantage at ℓ , Definition 9 restated). *For any tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$, hardcore measure $H : \{\pm 1\}^n \rightarrow [0, 1]$, distribution μ on $\{\pm 1\}^n$, $i \in [k]$, and leaf ℓ of T , the hardcore advantage at ℓ in the i th block is the quantity:*

$$\text{Adv}_H(\ell, i) := \left| \mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)}) T(\mathbf{X})_i H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right|. \quad (9)$$

The total hardcore advantage at ℓ is the quantity $\text{Adv}_H(\ell) := \sum_{i=1}^k \text{Adv}_H(\ell, i)$.

► **Lemma 28** (Expected total hardcore advantage, Claim 12 restated). *If H is a (γ, d) -hardcore measure for f of density δ w.r.t. μ and the expected depth of T is at most dk , then*

$$\mathbb{E}_{\ell \sim \mu^k(T)} [\text{Adv}_H(\ell)] \leq \gamma \mathbb{E}_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell)] = \gamma \delta k.$$

16:18 A Strong Direct Sum Theorem for Distributional Query Complexity

Proof of Lemma 28. By contrapositive. Suppose there exists $T_{\text{large}} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ making dk queries on average w.r.t. μ^k for which,

$$\mathbb{E}_{\ell \sim \mu^k(T)} [\text{Adv}_H(\ell)] > \gamma \mathbb{E}_{\ell \sim \mu^k(T)} [\text{Dens}_H(\ell)] = \gamma \delta k.$$

Then, we'll show there exists $T_{\text{small}} : \{\pm 1\}^n \rightarrow \{\pm 1\}$ making d queries on average w.r.t. μ for which

$$\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x}) T_{\text{small}}(\mathbf{x}) H(\mathbf{x})] > \gamma \cdot \mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})] = \gamma \delta.$$

Before constructing T_{small} , we observe that we can assume, without loss of generality, that for every leaf ℓ of T , that we can remove the absolute value from Equation (9); i.e. that

$$\text{Adv}_H(\ell, i) = \mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)}) T(\mathbf{X})_i H(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell]$$

Otherwise, we could modify this leaf by flipping the label of $T(\mathbf{X})_i$ whenever \mathbf{X} reaches a leaf where the above quantity is negative. This does not change the hardcore advantage, so this new T still satisfies our assumption.

T_{small} will be a randomized algorithm. Upon receiving the input $x \in \{\pm 1\}^n$, it samples $\mathbf{X} \sim \mu^k$ and $\mathbf{i} \sim \text{Unif}([k])$, and then constructs $\mathbf{X}(x, \mathbf{i})$ by inserting x into the \mathbf{i}^{th} block of \mathbf{X} ,

$$(\mathbf{X}(x, \mathbf{i}))^{(j)} = \begin{cases} \mathbf{X}^{(j)} & \text{if } j \neq i \\ x & \text{if } j = i. \end{cases}$$

Then, $T_{\text{small}}(x)$ outputs $T_{\text{large}}(\mathbf{X}(x, \mathbf{i}))_{\mathbf{i}}$.

Our analysis of T_{small} relies on the following simple observation: If we sample $\mathbf{x} \sim \mu$, then even conditioning on any choice of $\mathbf{i} = i$, the distribution of $\mathbf{X}(\mathbf{x}, \mathbf{i})$ is μ^k . This also means that $\mathbf{X}(\mathbf{x}, \mathbf{i})$ and \mathbf{i} are independent.

We claim that T_{small} has the two desired properties; low expected number of queries, and high accuracy on H . To bound the expected number of queries T_{small} makes on an input $\mathbf{x} \sim \mu$, we use that $\mathbf{X}(\mathbf{x}, \mathbf{i})$ is distributed according to μ^k . Therefore, $T_{\text{large}}(\mathbf{X}(\mathbf{x}, \mathbf{i}))$ makes, on average, dk queries. Expanded, we have that,

$$\sum_{i \in [k], j \in [n]} \Pr[T_{\text{large}}(\mathbf{X}(\mathbf{x}, \mathbf{i})) \text{ queries } \mathbf{X}(\mathbf{x})_j^{(i)}] = dk.$$

Whereas, the number of queries $T_{\text{small}}(\mathbf{x})$ makes only counts queries to the \mathbf{i}^{th} block, and is therefore,

$$\begin{aligned} \sum_{i \in [k], j \in [n]} \Pr [T_{\text{large}}(\mathbf{X}(\mathbf{x}, \mathbf{i})) \text{ queries } \mathbf{X}(\mathbf{x}, \mathbf{i})_j^{(i)} \cdot \mathbf{1}[\mathbf{i} = i]] \\ &= \sum_{i \in [k], j \in [n]} \Pr [T_{\text{large}}(\mathbf{X}(\mathbf{x}, \mathbf{i})) \text{ queries } \mathbf{X}(\mathbf{x}, \mathbf{i})_j^{(i)}] \cdot \frac{1}{k} \\ &= d. \end{aligned}$$

In the above, the first equality uses that \mathbf{i} is independent of $\mathbf{X}(\mathbf{x}, \mathbf{i})$, and so is still uniform on $[k]$ even conditioned on which queries T_{large} makes.

Lastly, we verify that T_{small} has high accuracy on the hardcore measure.

$$\begin{aligned}
\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x}) T_{\text{small}}(\mathbf{x}, \mathbf{i}) H(\mathbf{x})] &= \mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x}) T_{\text{large}}(\mathbf{X}(\mathbf{x}, \mathbf{i}))_{\mathbf{i}} H(\mathbf{x})] && \text{(Definition of } T_{\text{small}}) \\
&= \mathbb{E}_{\mathbf{i} \sim [k]} \left[\mathbb{E}_{\mathbf{X} \sim \mu^k} \left[f(\mathbf{X}^{(\mathbf{i})}) T_{\text{large}}(\mathbf{X})_{\mathbf{i}} H(\mathbf{X}^{(\mathbf{i})}) \right] \right] && (\mathbf{i}, \mathbf{X}(\mathbf{x}) \text{ are independent}) \\
&= \mathbb{E}_{\mathbf{i} \sim [k]} \left[\mathbb{E}_{\ell \sim \mu^k(T_{\text{large}})} [\text{Adv}_H(\ell, \mathbf{i})] \right] && \text{(Definition 27)} \\
&= \frac{1}{k} \mathbb{E}_{\ell \sim \mu^k(T_{\text{large}})} [\text{Adv}_H(\ell)] > \gamma \delta. && \blacktriangleleft
\end{aligned}$$

8.3 Understanding the error in terms of hardcore density and advantage

To state the main result of this subsection, we'll define the following distribution for the sum of independent Bernoulli random variables.

► **Definition 29.** For any $p \in [0, 1]^k$, we use $\text{BerSum}(p)$ to denote the distribution of $\mathbf{z} := z_1 + \dots + z_k$ where each z_i is independently drawn from $\text{Ber}(p_i)$.

The following generalizes Lemma 10.

► **Lemma 30** (Accuracy in terms of hardcore density and advantage of the leaves). Let H be a (γ, d) -hardcore measure w.r.t. μ for $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, and $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}$ be any tree. Then, for any $t \geq 0$,

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}) - f^{\otimes k}(\mathbf{X})\|_0 \leq t] \leq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\Pr_{\mathbf{z} \sim \text{BerSum}(p(\ell))} [\mathbf{z} \leq t] \right]$$

where $p(\ell) \in [0, 1]^k$ is the vector where

$$p(\ell)_i := \frac{\text{Dens}_H(\ell, i) - \text{Adv}_H(\ell, i)}{2} \quad \text{for each } i \in [k].$$

Lemma 30 implies a generalization of Lemma 10.

► **Corollary 31.** Let H be a (γ, d) -hardcore measure w.r.t. μ for $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, and $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}$ be any tree. Then, for any $t \geq 0$,

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}) - f^{\otimes k}(\mathbf{X})\|_0 \leq t] \leq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\min \left(1, \exp \left(t - \frac{\text{Dens}_H(\ell) - \text{Adv}_H(\ell)}{4} \right) \right) \right].$$

Proof. The Chernoff bound of Fact 2 says that, for any $p \in [0, 1]^k$ and $\mu := \sum_{i \in [k]} p_i$,

$$\Pr_{\mathbf{z} \sim \text{BerSum}(p)} [\mathbf{z} \leq t] \leq \begin{cases} \exp \left(-\frac{(\mu-t)^2}{2\mu} \right) & \text{if } \mu \geq t \\ 1 & \text{otherwise.} \end{cases}$$

We want to show that the above is bounded by $\min(1, e^{t-\mu/2})$. Clearly this holds for $\mu < t$, so we need only consider the case where $\mu \geq t$

$$\exp \left(-\frac{(\mu-t)^2}{2\mu} \right) = \exp \left(-\frac{\mu^2 - 2t\mu + t^2}{2\mu} \right) \leq \exp \left(-\frac{\mu^2 - 2t\mu}{2\mu} \right) = e^{t-\mu/2}.$$

Since $\exp \left(-\frac{(\mu-t)^2}{2\mu} \right) \leq 1$ as well, it is upper bounded by $\min(1, e^{t-\mu/2})$ as desired. The desired result follows from Lemma 30 as well as $\sum_{i \in [k]} p(\ell)_i = \frac{\text{Dens}_H(\ell) - \text{Adv}_H(\ell)}{2}$ for every leaf ℓ of T . ◀

16:20 A Strong Direct Sum Theorem for Distributional Query Complexity

The main observation underlying Lemma 30 is that, if we choose an input $\mathbf{X} \sim \mu^k$ conditioned on reaching a leaf $\ell \in T$, that \mathbf{X} is distributed according to a k -wise product distribution (i.e. from $\mu_1(\ell) \times \cdots \times \mu_k(\ell)$ for appropriately defined distributions). The below is essentially the same as Lemma 3.2 of [10], but we include a proof for completeness.

▷ **Claim 32.** For any (potentially randomized) tree $T : (\{\pm 1\}^n)^k \rightarrow \mathcal{Y}$ and leaf ℓ of T , if $\mathbf{X} \sim \mu^k$, then the distribution of \mathbf{X} conditioned on reaching the leaf ℓ is a product distribution over the k blocks of \mathbf{X} .

Proof. First, if T is a randomized tree, it is as a distribution over deterministic trees. If the desired result holds for each of those deterministic trees, it also holds for T . Therefore, it suffices to consider the case where T is deterministic.

We'll prove that the distribution of \mathbf{X} reaching any internal node *or* leaf of T is product by induction on the depth of that node. If that depth is 0, then all inputs reach it and so the desired result follows from μ^k being product.

For depth $d \geq 1$, let α be the parent of ℓ . Then α has depth $d - 1$, so by the inductive hypothesis, the distribution of inputs reaching α is product. Let $i \in [k], j \in [n], b \in \{\pm 1\}$ be chosen so that an input X reaches ℓ iff it reaches α and $X_j^{(i)} = b$. Then,

$$\begin{aligned} & \Pr[\mathbf{X} = X \mid \mathbf{X} \text{ reaches } \ell] \\ &= \Pr[\mathbf{X} = X \mid \mathbf{X} \text{ reaches } \alpha] \cdot \frac{\mathbb{1}[X_j^{(i)} = b]}{\Pr[X_j^{(i)} = b \mid \mathbf{X} \text{ reaches } \alpha]} \\ &= \frac{\mathbb{1}[X_j^{(i)} = b]}{\Pr[X_j^{(i)} = b \mid \mathbf{X} \text{ reaches } \alpha]} \cdot \prod_{\ell \in [k]} \Pr[\mathbf{X}^{(\ell)} = X^{(\ell)} \mid \mathbf{X} \text{ reaches } \alpha] \quad (\text{Inductive hypothesis}) \\ &= \left(\prod_{\ell \neq i} \Pr[\mathbf{X}^{(\ell)} = X^{(\ell)} \mid \mathbf{X} \text{ reaches } \alpha] \right) \cdot \frac{\Pr[\mathbf{X}^{(i)} = X^{(i)} \mid \mathbf{X} \text{ reaches } \alpha] \cdot \mathbb{1}[X_j^{(i)} = b]}{\Pr[X_j^{(i)} = b \mid \mathbf{X} \text{ reaches } \alpha]} \\ &= \left(\prod_{\ell \neq i} \Pr[\mathbf{X}^{(\ell)} = X^{(\ell)} \mid \mathbf{X} \text{ reaches } \alpha] \right) \cdot \Pr[\mathbf{X}^{(i)} = X^{(i)} \mid \mathbf{X} \text{ reaches } \alpha, \mathbf{X}_j^{(i)} = b]. \end{aligned}$$

The above is decomposed as a product over the k components of X , so is a product distribution. \triangleleft

We conclude this subsection with a proof of Lemma 30.

Proof of Lemma 30. Consider any leaf ℓ of T . We wish to compute the probability that $T(\mathbf{X})$ makes less than t mistakes on $f^{\otimes k}(\mathbf{X})$ given that \mathbf{X} reaches the leaf ℓ . On this leaf, T outputs a single vector $y \in \{\pm 1\}^k$. Meanwhile, by Claim 32, the distribution of \mathbf{X} is product over the blocks, and so $f(\mathbf{X}^{(1)}), \dots, f(\mathbf{X}^{(k)})$ are independent. Define $q(\ell) \in [0, 1]^k$ as,

$$q(\ell)_i := \Pr_{\mathbf{X} \sim \mu^k} [y_i \neq f(\mathbf{X}^{(i)})].$$

Then,

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}), f^{\otimes k}(\mathbf{X})\|_0 \leq t \mid \mathbf{X} \text{ reaches } \ell] = \Pr_{\mathbf{z} \sim \text{BerSum}(q(\ell))} [\mathbf{z} \leq t].$$

For $\mathbf{z} \sim \text{BerSum}(q)$, the probability $\mathbf{z} \leq t$ is monotonically decreasing in each q_i . Therefore, it suffices to show that $q(\ell)_i \geq p(\ell)_i$ for each $i \in [k]$. We compute,

$$\begin{aligned} q(\ell)_i &= \Pr_{\mathbf{X} \sim \mu^k} [T(\mathbf{X})_i \neq f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \\ &= \frac{1 - \mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X})_i f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell]}{2} \quad (T(\mathbf{X})_i, f(\mathbf{X}^{(i)}) \in \{\pm 1\}) \end{aligned}$$

Separating the above expectation into two pieces, for \mathbf{X} drawn from μ^k conditioned in \mathbf{X} reaching ℓ ,

$$\begin{aligned} \mathbb{E}[T(\mathbf{X})_i f(\mathbf{X}^{(i)})] &= \mathbb{E}[T(\mathbf{X})_i f(\mathbf{X}^{(i)})H(\mathbf{X}^{(i)})] + \mathbb{E}[T(\mathbf{X})_i f(\mathbf{X}^{(i)})(1 - H(\mathbf{X}^{(i)}))] \\ &\leq \text{Adv}_H(\ell, i) + \mathbb{E}[T(\mathbf{X})_i f(\mathbf{X}^{(i)})(1 - H(\mathbf{X}^{(i)}))] \quad (\text{Definition 9}) \\ &\leq \text{Adv}_H(\ell, i) + \mathbb{E}[(1 - H(\mathbf{X}^{(i)}))] = 1 + \text{Adv}_H(\ell, i) - \text{Dens}_H(\ell, i). \end{aligned}$$

Therefore,

$$q(\ell)_i \geq \frac{\text{Dens}_H(\ell, i) - \text{Adv}_H(\ell, i)}{2} = p(\ell)_i. \quad \blacktriangleleft$$

8.4 Completing the proof of the threshold direct sum theorem

In this subsection, we complete the proof of Theorem 23. Throughout this section, we'll use the following function:

$$g_t(z) := \min(1, e^{t-z/4}).$$

By using Corollary 31, it suffices to show that

$$\mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell))] \leq e^{-\delta k/10} + 1 - \gamma. \quad (10)$$

Recall that we have much information about how $\text{Dens}_H(\ell)$ distributes over the leaves via Lemma 26, but a coarser understanding of how $\text{Adv}_H(\ell)$ distributes via Lemma 28. Because of this, we will first bound the above equation where the $\text{Adv}_H(\ell)$ is set to 0 and analyze how much including that term affects the result.

► **Lemma 33.** *For any tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ and hardcore measure of density δ w.r.t. distribution μ ,*

$$\mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell))] \leq e^{-.121\delta k}.$$

Proof. We bound,

$$\mathbb{E}_{\ell \sim \mu^k(T)} \left[\min\left(1, \exp\left(\delta k/10 - \frac{\text{Dens}_H(\ell)}{4}\right)\right) \right] \leq e^{\delta k/10} \cdot \mathbb{E}_{\ell \sim \mu^k(T)} \left[e^{-\text{Dens}_H(\ell)/4} \right].$$

Since $z \mapsto e^{-z/4}$ is convex, we can use Lemma 26 to bound the above using the moment generating function of the binomial distribution,

$$\begin{aligned} \mathbb{E}_{\ell \sim \mu^k(T)} \left[e^{-\text{Dens}_H(\ell)/4} \right] &\leq \mathbb{E}_{z \sim \text{Bin}(k, \delta)} [e^{-z/4}] \\ &= (1 - \delta(1 - e^{-1/4}))^k \\ &\leq e^{-(1 - e^{-1/4})\delta k}. \end{aligned}$$

Combining the above,

$$\mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell))] \leq e^{-\delta k(1 - e^{-1/4} - 1/10)} \leq e^{-0.121\delta k}. \quad \blacktriangleleft$$

Next, we prove a Lipschitz-style bound for g . This will be useful in incorporating $\text{Adv}_H(\ell)$ to Equation (10).

16:22 A Strong Direct Sum Theorem for Distributional Query Complexity

► **Proposition 34.** For any $z, \Delta, t \geq 0$,

$$g_t(z - \Delta) \leq g_t(z) + \Delta/4. \quad (11)$$

Furthermore, if $z \geq 5t$, then

$$g_t(z - \Delta) \leq g_t(z) + \Delta/t. \quad (12)$$

Proof. Equation (11) follows from the $(1/4)$ -Lipschitzness of $g_t(z)$.

For Equation (12), fix any choice of $z \geq 5t$. We want to show that for any choice of Δ ,

$$\frac{g_t(z - \Delta) - g_t(z)}{\Delta} \leq 1/t.$$

We claim that the left hand side of the above inequality is maximized when $\Delta = z - 4t$. When Δ is increased beyond $z - 4t$, the numerator remains constant (because $g_t(z)$ is constant for any $z \leq 4t$, but the denominator increases, so the maximum cannot occur at any $\Delta > z - 4t$. On the other hand, $g_t(z)$ is convex when restricted to the domain $[4t, \infty)$, so the maximum cannot occur at any $\Delta < z - 4t$. Therefore, it suffices to consider $\Delta = z - 4t$, in which case,

$$\frac{g_t(z - \Delta) - g_t(z)}{\Delta} = \frac{1 - g_t(z)}{\Delta} \leq \frac{1}{\Delta} \leq 1/t. \quad \blacktriangleleft$$

We are now ready to prove the main result of this section.

Proof of Theorem 23. By applying Corollary 31,

$$\Pr_{\mathbf{X} \sim \mu^k} [\|T(\mathbf{X}) - f^{\otimes k}(\mathbf{X})\|_0 \leq \delta k/10] \leq \mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell))].$$

First, we consider the case where $\delta k \leq 40$. Here, by applying Equation (11),

$$\begin{aligned} & \mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell))] \\ & \leq \mathbb{E}_{\ell \sim \mu^k(T)} [g_{\delta k/10}(\text{Dens}_H(\ell))] + \frac{1}{4} \cdot \mathbb{E}_{\ell \sim \mu^k(T)} [\text{Adv}_H(\ell)] \\ & \leq e^{-0.121\delta k} + \gamma\delta k/4 \quad (\text{Lemmas 28 and 33}) \\ & \leq e^{-\delta k/10} + 10\gamma \quad (\delta k \leq 40) \end{aligned}$$

When $\delta k > 40$, we break down the desired result into two pieces, depending on whether $\text{Dens}_H(\ell)$ is small or large. For the piece where $\text{Dens}_H(\ell)$ is small, we just use that $g(\cdot)$ is bounded between 0 and 1 which means $g(z) - g(z - \Delta) \leq 1$,

$$\begin{aligned} & \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) \leq \delta k/2]] \\ & \leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) \leq \delta k/2]] + \Pr[\text{Dens}_H(\ell) \leq \delta k/2] \\ & \leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) \leq \delta k/2]] + e^{-\delta k/8}. \quad (\text{Lemma 13}) \end{aligned}$$

For the piece where $\text{Dens}_H(\ell)$ is large, we use Equation (12)

$$\begin{aligned} & \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) > \delta k/2]] \\ & \leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) > \delta k/2]] + \frac{10}{\delta k} \cdot \mathbb{E} [\text{Adv}_H(\ell) \cdot \mathbf{1}[\text{Dens}_H(\ell) > \delta k/2]] \\ & \quad (\text{Equation (12)}) \\ & \leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) > \delta k/2]] + \frac{10}{\delta k} \cdot \mathbb{E} [\text{Adv}_H(\ell)] \quad (\text{Dens}_H(\ell) \geq 0) \\ & \leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell)) \cdot \mathbf{1}[\text{Dens}_H(\ell) > \delta k/2]] + \frac{10}{\delta k} \cdot \gamma\delta k \quad (\text{Lemma 28}) \end{aligned}$$

Combining the above two pieces,

$$\begin{aligned} \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell) - \text{Adv}_H(\ell))] &\leq \mathbb{E} [g_{\delta k/10}(\text{Dens}_H(\ell))] + e^{-\delta k/8} + 10\gamma \\ &\leq e^{-\delta k/8} + e^{-0.121\delta k} + 10\gamma \end{aligned} \quad (\text{Lemma 33})$$

When $\delta k > 40$, $e^{-\delta k/8} + e^{-0.121\delta k} < e^{-\delta k/10}$, so we also recover the desired result in this case. \blacktriangleleft

9 Equivalence between direct sum theorems and XOR lemmas and the proof of Theorem 3

In this section, we prove the following claim which shows that a strong direct sum theorem implies a strong XOR lemma. We then derive Theorem 3 as a consequence of this equivalence and our strong direct sum theorem for query complexity (Theorem 1).

▷ **Claim 35 (Equivalence between direct sum theorems and XOR lemmas).** For every $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, integer $k \in \mathbb{N}$, multiplicative factor $M \in \mathbb{R}$, and $\varepsilon \in (0, 1)$, if the following direct sum theorem holds:

$$\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq M \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

then, the following XOR lemma holds:

$$\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, \frac{\varepsilon}{2}) \geq M \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

In order to prove Claim 35 and Theorem 3, we establish a lemma which allows us to convert any decision accurately computing $f^{\oplus k}$ into a decision tree accurately computing $f^{\otimes k}$. The following definition captures this conversion.

► **Definition 36 (The product tree).** Given a decision tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}$, the k -wise product tree $\tilde{T} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ is defined as follows. For the internal nodes, \tilde{T} has exactly the same structure as T . For a leaf ℓ in T , the leaf vector $(\ell_1, \dots, \ell_k) \in \{\pm 1\}^k$ in \tilde{T} is defined by

$$\ell_i := \text{sign} \left(\mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right)$$

for all $i \in [k]$.

Intuitively, \tilde{T} computes T 's best guess for $f(\mathbf{X}^{(i)})$ for each $i \in [k]$ on a given input $(X^{(1)}, \dots, X^{(k)})$. If T is really good at computing $f^{\oplus k}$ then at every leaf it should have queried enough variables to pin down f 's value on each of the input blocks. The main lemma formalizes this intuition.

► **Lemma 37.** For any $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, distribution μ over $\{\pm 1\}^n$, and tree $T : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}$, the k -wise product tree $\tilde{T} : (\{\pm 1\}^n)^k \rightarrow \{\pm 1\}^k$ satisfies

$$\Pr_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X}) = f^{\otimes k}(\mathbf{X})] \geq \mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) f^{\oplus k}(\mathbf{X})].$$

9.1 Proofs of Claim 35 and Theorem 3 assuming Lemma 37

The following corollary of Lemma 37 implies Claim 35 and Theorem 3.

► **Corollary 38** (Main corollary of Lemma 37). *For all $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$, $k \geq 1$, distributions μ over $\{\pm 1\}^n$, and $\varepsilon > 0$, $\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, \frac{\varepsilon}{2}) \geq \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon)$.*

Proof. Let \mathbf{A} be a randomized query algorithm for $f^{\oplus k}$ with error $\varepsilon/2$ and expected cost $q = \overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, \varepsilon/2)$. Let \mathcal{T} denote the distribution over decision trees determined by \mathbf{A} . Consider the algorithm $\tilde{\mathbf{A}}$ which computes $f^{\otimes k}(X)$ by sampling $T \sim \mathcal{T}$ and returning $\tilde{T}(X)$ where \tilde{T} is the decision tree from Lemma 37. Then, the success of $\tilde{\mathbf{A}}$ is

$$\begin{aligned} \mathbb{E}_{T \sim \mathcal{T}} \left[\Pr_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X}) = f^{\otimes k}(\mathbf{X})] \right] &\geq \mathbb{E}_{T \sim \mathcal{T}} \left[\mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) f^{\oplus k}(\mathbf{X})] \right] && \text{(Lemma 37)} \\ &\geq 1 - \varepsilon \end{aligned}$$

where the last step uses the fact that advantage is $1 - 2 \cdot \text{error}$. Since the structure of each \tilde{T} is the same as T , the expected cost of $\tilde{\mathbf{A}}$ is q which completes the proof. ◀

Proofs of Claim 35 and Theorem 3. By Corollary 38,

$$\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, \frac{\varepsilon}{2}) \geq \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq M \cdot \overline{\text{Depth}}^{\mu}(f, \delta).$$

Theorem 3 follows immediately by applying Claim 35 to Theorem 2. ◀

► **Remark 39** (On the necessity of the $1/2$ loss in ε in Corollary 38). One may wonder whether the $1/2$ loss in ε parameter in Corollary 38 is necessary. For example, can one show $\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, 0.51\varepsilon) \geq \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon)$? The issue is that $\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, 0.5) = 0$ for all functions $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ because the bias of $f^{\oplus k}$ is at least 0.5 . So such a statement cannot hold for all f in all parameter regimes. Concretely, one can show that if f is the parity of n bits and μ is uniform over $\{\pm 1\}^n$, then $\overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon) \geq \Omega(kn)$ for all constant $\varepsilon < 1$. Any path in a decision tree for $f^{\otimes k}$ which queries at most λkn bits for some constant $\lambda < 1$ has success probability $2^{-\Omega(k)}$. So to achieve any constant accuracy requires $\Omega(kn)$ expected depth. On the other hand, $\overline{\text{Depth}}^{\mu^k}(f^{\oplus k}, 0.5) = 1 \ll \overline{\text{Depth}}^{\mu^k}(f^{\otimes k}, \varepsilon)$ which is achieved by the decision tree that outputs a single constant value. Therefore, the $\varepsilon/2$ in Corollary 38 is necessary for such a statement to hold in full generality.

9.2 Proof of Lemma 37

Each ℓ_i for $i \in [k]$ satisfies

$$\begin{aligned} \mathbb{E}_{\mathbf{X} \sim \mu^k} [\ell_i \cdot f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] &= \left| \mathbb{E}_{\mathbf{X} \sim \mu^k} [f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right| \\ &\geq \mathbb{E}_{\mathbf{X} \sim \mu^k} [\ell \cdot f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell]. \end{aligned}$$

In particular, for all leaves ℓ of T ,

$$\mathbb{E}_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X})_i \cdot f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \geq \mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) \cdot f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell]. \quad (13)$$

Therefore:

$$\begin{aligned}
& \Pr_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X}) = f^{\otimes k}(\mathbf{X})] \\
&= \mathbb{E}_{\ell \sim \mu^k(T)} \left[\Pr_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X}) = f^{\otimes k}(\mathbf{X}) \mid \mathbf{X} \text{ reaches } \ell] \right] \\
&= \mathbb{E}_{\ell \sim \mu^k(T)} \left[\prod_{i \in [k]} \Pr_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X})_i = f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right] \quad (\text{Claim 32}) \\
&\geq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\prod_{i \in [k]} \mathbb{E}_{\mathbf{X} \sim \mu^k} [\tilde{T}(\mathbf{X})_i f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right] \\
&\geq \mathbb{E}_{\ell \sim \mu^k(T)} \left[\prod_{i \in [k]} \mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell] \right] \quad (\text{Equation (13)}) \\
&= \mathbb{E}_{\ell \sim \mu^k(T)} \left[\mathbb{E}_{\mathbf{X} \sim \mu^k} \left[T(\mathbf{X}) \prod_{i \in [k]} f(\mathbf{X}^{(i)}) \mid \mathbf{X} \text{ reaches } \ell \right] \right] \quad (\text{Claim 32}) \\
&= \mathbb{E}_{\ell \sim \mu^k(T)} \left[\mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) f^{\otimes k}(\mathbf{X}) \mid \mathbf{X} \text{ reaches } \ell] \right] \quad (\prod_{i \in [k]} f(\mathbf{X}^{(i)}) = f^{\oplus k}(\mathbf{X})) \\
&= \mathbb{E}_{\mathbf{X} \sim \mu^k} [T(\mathbf{X}) f^{\oplus k}(\mathbf{X})]
\end{aligned}$$

which completes the proof. \blacktriangleleft

10 Proof of Claim 4

\triangleright Claim 40 (The γ factor in Theorem 2 is necessary; Claim 4 restated). Let $\text{Par} : \{\pm 1\}^n \rightarrow \{\pm 1\}$ be the parity function and μ be the uniform distribution over $\{\pm 1\}^n$. Then for all γ ,

$$\overline{\text{Depth}}^{\mu^k}(\text{Par}^{\otimes k}, 1 - \gamma) \leq 2\gamma k \cdot \overline{\text{Depth}}^{\mu}(\text{Par}, \frac{1}{4}).$$

We will need the following simple proposition, which states that in any tree that seeks to compute the n -variable parity function, leaves of depth strictly less than n contribute $\frac{1}{2}$ to the error:

\blacktriangleright **Proposition 41.** *For any (potentially randomized) tree $T : \{\pm 1\}^n \rightarrow \{\pm 1\}$ and leaf ℓ of T with depth strictly less than n ,*

$$\Pr_{\mathbf{x} \sim \text{Unif}(\{\pm 1\}^n)} [T(\mathbf{x}) \neq \text{Par}(\mathbf{x}) \mid \mathbf{x} \text{ reaches } \ell] = \frac{1}{2}.$$

Proof. Since ℓ is at depth strictly less than n , there must be some index $i \in [n]$ not queried on the path to ℓ . Taking any input x that reaches ℓ , the input x' with the i^{th} bit flipped must also reach ℓ and have the opposite parity. Both of these inputs are equally likely under the uniform distribution and so the value of $\text{Par}(\mathbf{x})$ conditioned on \mathbf{x} reaching ℓ is equally likely to be $+1$ and -1 . Therefore, T errs half the time it reaches this leaf regardless of how it labels it. \blacktriangleleft

Proof of Claim 40. The proof proceeds in two parts. First, we show that $\overline{\text{Depth}}^{\mu}(\text{Par}, \frac{1}{4}) = \frac{n}{2}$. Second, we prove that $\overline{\text{Depth}}^{\mu^{\otimes k}}(\text{Par}^{\otimes k}, 1 - \gamma) \leq \gamma kn$.

(1) $\overline{\text{Depth}}^\mu(\text{Par}, \frac{1}{4}) = \frac{n}{2}$. Let T be an arbitrary randomized decision tree let $p_n(T)$ be the probability that T queries all n variables. Then, the expected depth of T is at least $n \cdot p_n(T)$. Meanwhile, by Proposition 41, the error of T in computing parity is at least $\frac{1}{2} \cdot p_n(T)$ w.r.t. the uniform distribution. Therefore, $\overline{\text{Depth}}^\mu(f, \frac{1}{4}) \geq \frac{n}{2}$.

While this direction is not needed for Claim 40 we show for completeness that $\overline{\text{Depth}}^\mu(f, \frac{1}{4}) \leq \frac{n}{2}$ by constructing a randomized¹ decision tree T for f . With probability $\frac{1}{2}$, T queries all n variables to compute f exactly. Otherwise, it simply outputs 0. T has expected depth $\frac{n}{2}$, and it errs only when it queries no variables and guesses incorrectly, which happens with probability $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. Thus, $\overline{\text{Depth}}^\mu(f, \frac{1}{4}) \leq \frac{n}{2}$.

(2) $\overline{\text{Depth}}^{\mu^{\otimes k}}(f^{\otimes k}, 1 - \gamma) \leq \gamma kn$. We construct a randomized² decision tree T for $f^{\otimes k}$. With probability γ , T queries all kn variables to compute $f^{\otimes k}$ exactly, and with probability $1 - \gamma$, it outputs 0. When it queries all variables, it has no error so its average error is at most $1 - \gamma$. Furthermore, its average depth is γkn . \triangleleft

References

- 1 Andris Ambainis, Loïck Magnin, Martin Roetteler, and Jérémie Roland. Symmetry-assisted adversaries for quantum state generation. In *2011 IEEE 26th Annual Conference on Computational Complexity (CCC)*, pages 167–177. IEEE, 2011.
- 2 Andris Ambainis, Robert Špalek, and Ronald de Wolf. A new quantum lower bound method, with applications to direct product theorems and time-space tradeoffs. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 618–633, 2006.
- 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, *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 612–625, 2021.
- 4 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 67–76, 2010.
- 5 Shalev Ben-David and Robin Kothari. Randomized query complexity of sabotaged and composed functions. *Theory of Computing*, 14(5):1–27, 2018. doi:10.4086/toc.2018.v014a005.
- 6 Eric Blais and Joshua Brody. Optimal Separation and Strong Direct Sum for Randomized Query Complexity. In *34th Computational Complexity Conference (CCC)*, volume 137, pages 29:1–29:17, 2019. doi:10.4230/LIPIcs.CCC.2019.29.
- 7 Guy Blanc, Caleb Koch, Carmen Strassle, and Li-Yang Tan. A strong composition theorem for junta complexity and the boosting of property testers. In *Proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1757–1777, 2023.
- 8 Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct products in communication complexity. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 746–755, 2013.
- 9 Joshua Brody, Jae Tak Kim, Peem Lerdputtipongporn, and Hariharan Srinivasulu. A strong xor lemma for randomized query complexity. *Theory of Computing*, 19(11):1–14, 2023. doi:10.4086/toc.2023.v019a011.
- 10 Andrew Drucker. Improved direct product theorems for randomized query complexity. *computational complexity*, 21(2):197–244, 2012.

¹ At the cost of increasing expected depth by 1, the tree can be derandomized. To derandomize it, read a single bit of the input and only query the rest if that bit is 1, which occurs with probability $\frac{1}{2}$.

² Again, this can be derandomized at the cost of adding ≤ 2 to the expected depth, since any biased coin can be simulated with a fair coin using 2 flips in expectation.

- 11 Andrew Drucker. Nondeterministic direct product reductions and the success probability of sat solvers. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 736–745, 2013.
- 12 Oded Goldreich, Noam Nisan, and Avi Wigderson. On yao’s xor-lemma. *Studies in Complexity and Cryptography*, 6650:273–301, 2011.
- 13 William Hoza. A technique for hardness amplification against AC0. *ECCC preprint TR23-176*, 2023.
- 14 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of IEEE 36th Annual Foundations of Computer Science (FOCS)*, pages 538–545, 1995.
- 15 Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform direct product theorems: simplified, optimized, and derandomized. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 579–588, 2008.
- 16 Russell Impagliazzo, Ran Raz, and Avi Wigderson. A direct product theorem. In *Proceedings of IEEE 9th Annual Conference on Structure in Complexity Theory*, pages 88–96, 1994.
- 17 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing (STOC)*, pages 220–229, 1997.
- 18 Rahul Jain. New strong direct product results in communication complexity. *Journal of the ACM (JACM)*, 62(3):1–27, 2015.
- 19 Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Information Processing Letters*, 110(20):893–897, 2010.
- 20 Rahul Jain, Attila Pereszlényi, and Penghui Yao. A direct product theorem for the two-party bounded-round public-coin communication complexity. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 167–176, 2012.
- 21 Hartmut Klauck. A strong direct product theorem for disjointness. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 77–86, 2010.
- 22 Hartmut Klauck, Robert Špalek, and Ronald de Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. *SIAM Journal on Computing*, 36(5):1472–1493, 2007. Preliminary version in FOCS 2004.
- 23 Adam R Klivans and Rocco A Servedio. Boosting and hard-core set construction. *Machine Learning*, 51:217–238, 2003.
- 24 Troy Lee and Jérémie Roland. A strong direct product theorem for quantum query complexity. *computational complexity*, 22:429–462, 2013.
- 25 Troy Lee, Adi Shraibman, and Robert Špalek. A direct product theorem for discrepancy. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity (CCC)*, pages 71–80, 2008.
- 26 Leonid A Levin. One-way functions and pseudorandom generators. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing (STOC)*, pages 363–365, 1985.
- 27 Noam Nisan, Steven Rudich, and Michael Saks. Products and help bits in decision trees. In *Proceedings 35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 318–329, 1994.
- 28 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994.
- 29 Ryan O’Donnell. Hardness amplification within np. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 751–760, 2002.
- 30 Ronen Shaltiel. Towards proving strong direct product theorems. *Computational Complexity*, 12(1/2):1–22, 2004.
- 31 Alexander A Sherstov. Strong direct product theorems for quantum communication and query complexity. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 41–50, 2011.

- 32 Robert Špalek. The multiplicative quantum adversary. In *23rd Annual IEEE Conference on Computational Complexity (CCC)*, pages 237–248, 2008.
- 33 Volker Strassen. Vermeidung von divisionen. *Journal für die reine und angewandte Mathematik*, 264:184–202, 1973.
- 34 Luca Trevisan. The Impagliazzo Hard-Core-Set Theorem. <https://lucatrevisan.wordpress.com/2007/11/06/the-impagliazzo-hard-core-set-theorem/>, 2007.
- 35 Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(1):137–168, 2008.
- 36 Andrew Yao. Theory and application of trapdoor functions. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.
- 37 Huacheng Yu. Strong xor lemma for communication with bounded rounds. In *Proceedings of the 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1186–1192, 2022.

A Figures of stacked and fair trees

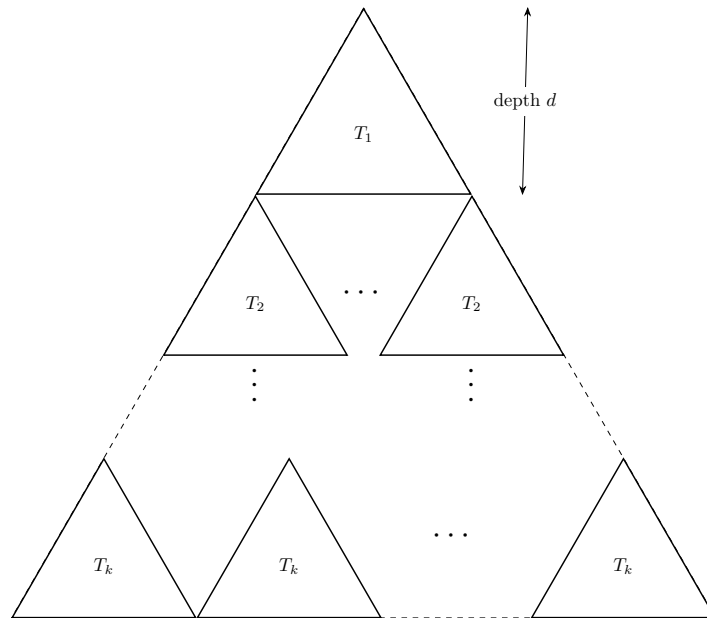
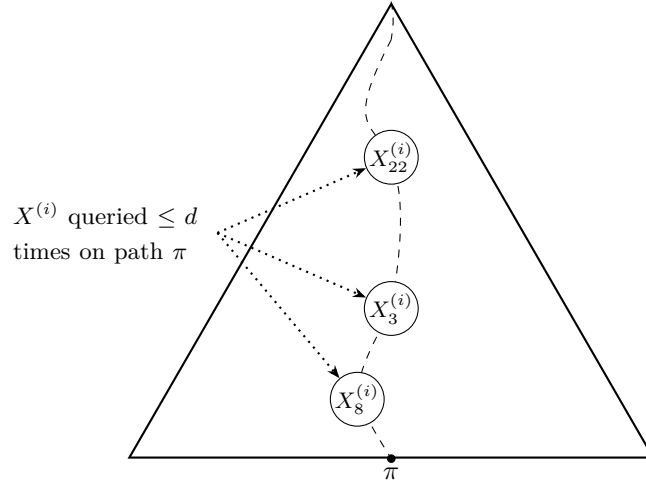


Figure 3 Illustration of a stacked decision tree for a function $f^{\otimes k}$. The decision tree consists of k depth- d decision trees, T_1, \dots, T_k , stacked on top of each other. For an input $X \in (\{\pm 1\}^n)^k$, the output $T(X)$ is computed sequentially, first by computing $T_1(X)$, then $T_2(X)$, and so on. The final output is $T(X) := (T_1(X), \dots, T_k(X))$.

B Proof of Theorem 6

Let \mathcal{H} denote the set of measures of density $\delta/2$ with respect to μ and let \mathcal{T} denote the set of decision trees T whose expected depth with respect to μ is at most d . Suppose for contradiction that there *does not* exist an $H \in \mathcal{H}$ which is (γ, d) -hardcore. That is, for all $H \in \mathcal{H}$ there is a tree T of expected depth at most d satisfying

$$\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})T(\mathbf{x})H(\mathbf{x})] > \gamma \mathbb{E}_{\mathbf{x} \sim \mu} [H(\mathbf{x})] = \gamma\delta/2. \tag{14}$$



■ **Figure 4** Illustration of a fair decision tree. For every block $i \in [k]$ and path π , the input block $X^{(i)}$ is queried at most d times.

We use the minimax theorem to switch the quantifiers in the above statement. Consider the payoff matrix M whose rows are indexed by distributions from \mathcal{H} and whose columns are indexed by algorithms from \mathcal{T} and whose entries are given by $M_{H,T} := \mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})T(\mathbf{x})H(\mathbf{x})]$. This is the payoff matrix for the zero-sum game where the row player first chooses a row H and the column player then chooses a column T and the payoffs are determined by $M_{H,T}$. Note that once the first player's strategy is fixed, we can assume without loss of generality that the second player's strategy is deterministic. Therefore, the minimax theorem for zero-sum games yields

$$\begin{aligned} \gamma\delta/2 &< \min_{\rho \in \mu(\mathcal{H})} \max_{T \in \mathcal{T}} (\rho^\top M)_T && \text{(Equation (14))} \\ &= \max_{\tau \in \mu(\mathcal{T})} \min_{H \in \mathcal{H}} (M\tau)_H && \text{(minimax theorem)} \end{aligned}$$

where $\mu(\cdot)$ denotes the set of distributions over a given set. Therefore, there is a fixed distribution τ over the set \mathcal{T} such that for all $H \in \mathcal{H}$

$$\mathbb{E}_{T \sim \tau} \left[\mathbb{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})T(\mathbf{x})H(\mathbf{x})] \right] > \gamma\delta/2. \quad (15)$$

This shows that

$$\Pr_{\mathbf{x} \sim \mu} \left[\mathbb{E}_{T \sim \tau} [T(\mathbf{x})]f(\mathbf{x}) \geq \gamma \right] \geq 1 - \delta/2.$$

In particular, if instead $\Pr_{\mathbf{x} \sim \mu} [\mathbb{E}_{T \sim \tau} [T(\mathbf{x})]f(\mathbf{x}) < \gamma] \geq \delta/2$ then we can contradict Equation (15) by constructing a $\delta/2$ -density H such that $H(x) := \Pr_{\mathbf{x} \sim \mu} [\mathbf{x} = x]$ for a $\delta/2$ -fraction of x satisfying $\mathbb{E}_{T \sim \tau} [T(\mathbf{x})]f(\mathbf{x}) < \gamma$. Equation (15) shows that $\mathbb{E}_{T \sim \tau} [T(\mathbf{x})]$ has good correlation with f for a large fraction of inputs. We obtain a single strategy from the distribution τ by sampling $\mathbf{T}_1, \dots, \mathbf{T}_r \sim \tau$ for r sufficiently large (chosen later) and defining T^* as $T^*(x) := \text{MAJ}(\mathbf{T}_1(x), \dots, \mathbf{T}_r(x))$. For every x for which $\mathbb{E}_{T \sim \tau} [T(x)]f(x) \geq \gamma$, we have

$$\Pr_{\mathbf{T}_1, \dots, \mathbf{T}_r \sim \tau} \left[\text{MAJ}(\mathbf{T}_1(x), \dots, \mathbf{T}_r(x)) \neq f(x) \right] \leq 2^{-\Omega(\gamma^2 r)}$$

16:30 A Strong Direct Sum Theorem for Distributional Query Complexity

by a Chernoff bound. Choosing $r = \Theta(\log(1/\delta)/\gamma^2)$ ensures that the failure probability is at most $\delta/2$. The decision tree T^* satisfies $\Pr_{\mathbf{x} \sim \mu}[T^*(\mathbf{x}) \neq f(\mathbf{x})] \leq \delta$. The expected depth of T^* is less than

$$r \cdot d = \Theta(d \log(1/\delta)/\gamma^2) < \overline{\text{Depth}}^\mu(f, \delta)$$

which is a contradiction.

C The lack of error reduction for distributional error

In Section 2, we showed how error reduction gave a simple proof of a strong direct sum theorem for *randomized* query complexity. The specific statement needed in that proof is the following standard error reduction by repetition theorem.

► **Fact 3** (Error reduction for $\overline{\mathbb{R}}$). *For any function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ and $\delta > 0$,*

$$\overline{\mathbb{R}}(f, \delta) \leq O(\log(\frac{1}{\delta})) \cdot \overline{\mathbb{R}}(f, 1/4).$$

Here, we give a short proof that no error reduction holds in the distributional setting, even with substantially weaker parameters.

▷ **Claim 42.** For any $n \in \mathbb{N}$, let μ be the uniform distribution over $\{\pm 1\}^n$. There is a function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$ satisfying,

$$\overline{\text{Depth}}^\mu(f, 1/4) = 0 \quad \text{and} \quad \overline{\text{Depth}}^\mu(f, 1/8) \geq \Omega(n).$$

Since any function on n bits can be computed exactly using n , the f in the above claim requires essentially the maximum number of queries to be computed to error $1/8$ despite requiring no queries to be computed to error $1/4$.

Proof. We first define f : If $x_1 = 0$, then $f(\mathbf{x}) = 0$. Otherwise, $f(\mathbf{x})$ is the parity of the remaining $n - 1$ bits of \mathbf{x} .

Note that,

$$\Pr_{\mathbf{x} \sim \mu}[f(\mathbf{x}) = 0] = \frac{1}{2} \cdot (\Pr[f(\mathbf{x}) = 0 \mid \mathbf{x}_1 = 0] + \Pr[f(\mathbf{x}) = 0 \mid \mathbf{x}_1 = 1]) = \frac{3}{4}.$$

Therefore, the 0 query algorithm that simply outputs 0 has an error of only $1/4$ on f .

It only remains to prove that $\overline{\text{Depth}}^\mu(f, 1/10) \geq \Omega(n)$. Consider any (potentially randomized) $T : \{\pm 1\}^n \rightarrow \{\pm 1\}$ and leaf ℓ of T at depth strictly less than $n - 1$. By Proposition 41

$$\Pr_{\mathbf{x} \sim \mu}[T(\mathbf{x}) \neq f(\mathbf{x}) \mid \mathbf{x} \text{ reaches } \ell, \mathbf{x}_1 = 1] = 1/2.$$

Let p be the probability that $T(\mathbf{x})$ queries a leaf of depth at least $n - 1$ given that $\mathbf{x}_1 = 1$. The above allows us to conclude that

$$\Pr_{\mathbf{x} \sim \mu}[T(\mathbf{x}) \neq f(\mathbf{x})] \geq \frac{1}{2} \cdot \Pr[T(\mathbf{x}) \neq f(\mathbf{x}) \mid \mathbf{x}_1 = 1] \geq \frac{1}{4} \cdot p.$$

Therefore, if T has error at most $1/8$, then p must be at least $1/2$, which shows that $\overline{\text{Depth}}^\mu(f, 1/8) \geq \frac{n-1}{4}$. ◁

Local Enumeration and Majority Lower Bounds

Mohit Gurumukhani ✉ 🏠 

Cornell University, Ithaca, NY, USA

Ramamohan Paturi ✉

Department of Computer Science and Engineering, University of California,
San Diego, La Jolla, CA, USA

Pavel Pudlák ✉

Institute of Mathematics of the Czech Academy of Sciences, Prague, Czech Republic

Michael Saks ✉

Department of Mathematics, Rutgers University, Piscataway, NJ, USA

Navid Talebanfard ✉ 

University of Sheffield, UK

Institute of Mathematics of the Czech Academy of Sciences, Prague, Czech Republic

Abstract

Depth-3 circuit lower bounds and k -SAT algorithms are intimately related; the state-of-the-art Σ_3^k -circuit lower bound (Or-And-Or circuits with bottom fan-in at most k) and the k -SAT algorithm of Paturi, Pudlák, Saks, and Zane (J. ACM'05) are based on the same combinatorial theorem regarding k -CNFs. In this paper we define a problem which reveals new interactions between the two, and suggests a concrete approach to significantly stronger circuit lower bounds and improved k -SAT algorithms. For a natural number k and a parameter t , we consider the $\text{ENUM}(k, t)$ problem defined as follows: given an n -variable k -CNF and an initial assignment α , output all satisfying assignments at Hamming distance $t(n)$ of α , assuming that there are no satisfying assignments of Hamming distance less than $t(n)$ of α . We observe that an upper bound $b(n, k, t)$ on the complexity of $\text{ENUM}(k, t)$ simultaneously implies depth-3 circuit lower bounds and k -SAT algorithms:

- **Depth-3 circuits:** Any Σ_3^k circuit computing the Majority function has size at least $\binom{n}{\frac{n}{2}}/b(n, k, \frac{n}{2})$.
- **k -SAT:** There exists an algorithm solving k -SAT in time $O\left(\sum_{t=1}^{n/2} b(n, k, t)\right)$.

A simple construction shows that $b(n, k, \frac{n}{2}) \geq 2^{(1-O(\log(k)/k))n}$. Thus, matching upper bounds for $b(n, k, \frac{n}{2})$ would imply a Σ_3^k -circuit lower bound of $2^{\Omega(\log(k)n/k)}$ and a k -SAT upper bound of $2^{(1-\Omega(\log(k)/k))n}$. The former yields an unrestricted depth-3 lower bound of $2^{\omega(\sqrt{n})}$ solving a long standing open problem, and the latter breaks the Super Strong Exponential Time Hypothesis.

In this paper, we propose a randomized algorithm for $\text{ENUM}(k, t)$ and introduce new ideas to analyze it. We demonstrate the power of our ideas by considering the first non-trivial instance of the problem, i.e., $\text{ENUM}(3, \frac{n}{2})$. We show that the expected running time of our algorithm is 1.598^n , substantially improving on the trivial bound of $3^{n/2} \simeq 1.732^n$. This already improves Σ_3^3 lower bounds for Majority function to 1.251^n . The previous bound was 1.154^n which follows from the work of Håstad, Jukna, and Pudlák (Comput. Complex.'95).

By restricting ourselves to monotone CNFs, $\text{ENUM}(k, t)$ immediately becomes a hypergraph Turán problem. Therefore our techniques might be of independent interest in extremal combinatorics.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases Depth 3 circuits, k -CNF satisfiability, Circuit lower bounds, Majority function

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.17

Related Version Full Version: <https://arxiv.org/abs/2403.09134>

Funding Mohit Gurumukhani: Supported by NSF CAREER Award 2045576 and a Sloan Research Fellowship.



© Mohit Gurumukhani, Ramamohan Paturi, Pavel Pudlák, Michael Saks,
and Navid Talebanfard;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 17; pp. 17:1–17:25



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Ramamohan Paturi: Partially supported by NSF grant 2212136.

Pavel Pudlák: Partially supported by grant EXPRO 19-27871X of the Czech Grant Agency and the institute grant RVO: 67985840.



Navid Talebanfard: This project has received funding from the European Union's Horizon Europe research and innovation programme under the Marie Skłodowska-Curie grant agreement No 101106684 – EXCICO. Views and opinions expressed are however those of author(s) only and do not necessarily reflect those of the European Union or REA. Neither the European Union nor the granting authority can be held responsible for them.

1 Introduction

Local search is a fundamental paradigm in solving the satisfiability problem: find an assignment close in Hamming distance to the initial assignment that satisfies the formula, if one exists. Papadimitriou [18] was the first to employ this idea in a randomized poly-time 2-SAT algorithm. Schöning [24] showed that a slight modification of this algorithm yields a running time of $(2 - 2/k)^n$ for k -SAT. Dantsin et al. [4] considered a deterministic version of local search and gave a deterministic $(2 - 2/(k + 1))^n$ time algorithm. Brueggemann and Kern [2] and Kutzkov and Scheder [13] improved this deterministic local search procedure and obtained faster deterministic 3-SAT algorithms. Moser and Scheder [17] eventually considered a variant of the local search problem and used it to give a deterministic k -SAT algorithm matching the running time of Schöning's.

Despite the success of local search, the fastest known k -SAT algorithm PPSZ and its improvements follow a different approach: pick a random variable x , if its value is not easily seen to be forced¹ then assign it randomly and continue (see [20, 19, 10, 8, 22]). The analysis of this simple yet powerful algorithm consists of a combinatorial theorem relating the size and the structure of the set of satisfying assignments of a k -CNF. The strength of this combinatorial theorem is further manifested by the fact that the state-of-the-art depth-3 circuit lower bounds are built on it [20, 19].

This curious interaction between lower bounds and algorithms has become less of a surprise over the years. Williams [27] initiated a whole new line of inquiry by showing that improved satisfiability algorithms for a circuit class automatically imply lower bounds for the same class. Conversely, almost all known circuit lower bound techniques have been adopted in satisfiability algorithms (see e.g. [11, 3]). Within this context the role of local search is unclear.

► **Question 1.** *Can we derive lower bounds from local search algorithms?*

This question is also motivated by a lack of progress in improving depth-3 circuit lower bounds and related upper bounds on k -SAT algorithms.

Depth-3 circuit lower bounds. A Σ_3^k circuit is an Or-And-Or circuit where the bottom fan-in is bounded by k , i.e., a disjunction of k -CNFs. We use $\Sigma_3^k(f)$ to denote the minimum number of k -CNFs in a Σ_3^k circuit computing a function f . The study of these circuits was advocated by Valiant [25] who showed that a strong enough Σ_3^k lower bound for every fixed k implies a super-linear lower bound for series-parallel circuits. Moreover, [7] showed that strong lower bounds even for small constant k such as $k = 16$ imply various circuit lower bounds including better general circuit lower bounds. The technique of [20] gives a lower

¹ For example if x appears in a unit clause, or if such a clause can be derived in small width resolution.

bound of $\Omega(2^{n/k})$ for the parity function and it is known to be tight. In fact this results in a $\Omega(n^{\frac{1}{4}}2^{\sqrt{n}})$ lower bound for computing parity by unrestricted depth-3 circuits which is tight up to a constant factor. A further improvement comes from [19] which gives a lower bound of $2^{cn/k}$ where $c > 1$ for the BCH code. At this point, this is the best known lower bound for computing any explicit function by Σ_3^k circuits.

Majority is a natural candidate for going beyond the $2^{\Omega(\sqrt{n})}$ depth-3 circuit lower bound. The natural Σ_3^k circuit for computing Majority has size $2^{O(n \log(k)/k)}$ (which implies an unrestricted depth-3 size upper bound of $2^{O(\sqrt{n \log n})}$). Håstad, Jukna and Pudlák [9] introduced the intriguing notion of *k-limits* to capture the depth-3 complexity of various functions and proved a lower bound of $2^{\Omega(\sqrt{n})}$ where the constant factor in the exponent is improved over the constant one could obtain from Switching Lemma. Regarding Σ_3^k circuits computing Majority, their result implies size lower bounds of 1.414^n for $k = 2$, and 1.154^n for $k = 3$, and for $k \geq 4$ it yields nothing. The Σ_3^2 bound is known to be essentially tight [21]. More recently, [14] proved a tight lower bound of $2^{\Omega(n \log(k)/k)}$ for computing Majority by Σ_3^k circuits where each And gate depends on at most k variables. Further, [1] studied the effect of negations for Σ_3^k circuits computing majority. However, the question of proving tight lower bounds for computing Majority by depth-3 circuits (even for fan-in 3 circuits) remains open.

k-SAT upper bounds. Lack of progress in improving the savings beyond $\Omega(\frac{1}{k})$ for k -SAT algorithms led researchers to consider SSETH (Super Strong Exponential Time Hypothesis). SSETH is the hypothesis that k -SAT cannot be solved with *savings* asymptotically more than $1/k$, i.e., there is no $2^{(1-\epsilon_k)n}$ time k -SAT algorithm with $\epsilon_k = \omega(1/k)$. However, SSETH is known to be false on average [26, 15], that is, the satisfiability of almost all k -CNFs can be decided with much larger savings. It is thus not unreasonable to attempt to get such savings even in the worst-case. Yet we cannot hope to achieve such a savings for a large subclass of PPSZ-style algorithms [23].

It appears that making progress towards larger k -SAT savings as well as depth-3 circuit lower bounds requires new ideas. We argue that local search has the potential to achieve this goal, and as an evidence for this claim, we apply local search ideas to give a new Σ_3^3 lower bound for Majority function.

1.1 Local enumeration, k -SAT and Σ_3^k lower bounds for Majority function

The local search problem is formally defined as follows.

(k, t)-SAT. Given an n -variable k -CNF F , a parameter t , and an assignment α , decide if there is a satisfying assignment β of F such that $d(\alpha, \beta) \leq t(n)$, where $d(\cdot)$ is the Hamming distance.

Dantsin et al. [4] gave a simple branching algorithm which solves (k, t) -SAT in time $\text{poly}(n) \cdot k^t$. This already gives a non-trivial algorithm for 3-SAT: solve $(k, \frac{n}{2})$ -SAT starting with the all-0 and all-1 assignments. To get a non-trivial algorithm for larger k , they used *covering codes*, i.e., a small and asymptotically optimal number $C(n, t)$ of Hamming balls of a given radius t that cover the entire n -dimensional Boolean cube. Then an upper bound of $C(n, r) \cdot k^t$ follows immediately for k -SAT by solving (k, t) -SAT starting with the centers of each of the balls in the covering code. Setting $t = \frac{n}{k+1}$ minimizes this quantity. Thus

improved upper bounds for (k, t) -SAT immediately imply improved k -SAT upper bounds, and indeed this is what [2, 13] did by proving an upper bound of c^t for some $c < 3$ for $(3, t)$ -SAT. However, this improvement in local search is not sufficient to yield better upper bounds for k -SAT when we want to use the technique for large t . It is conceivable that improved bounds for large t combined with covering codes would yield improved k -SAT algorithms. This leads to the following question:

► **Question 2.** *What is the complexity of $(k, \epsilon n)$ -SAT where $0 < \epsilon \leq \frac{1}{2}$?*

It is also natural to consider the enumeration problem for (k, t) -SAT: enumerate all satisfying assignments within Hamming distance t of an initial assignment. We note that even a weaker form of this problem already captures the circuit complexity of Majority function. For this purpose, we introduce the following class of parameterized problems.

Enum(k, t). Given an n -variable k -CNF F and an initial assignment α , output all satisfying assignments of F at a Hamming distance t from α assuming that there are no satisfying assignments of F at a Hamming distance of less than t from α .

We observe that upper bounds on $\text{ENUM}(k, t)$ imply depth-3 circuit lower bounds and k -SAT algorithms.

► **Proposition 3.** *Assume that $\text{ENUM}(k, t)$ can be solved in randomized expected time $b(n, k, t)$. Then*

1. *any Σ_3^k circuit requires at least $\binom{n}{n/2}/b(n, k, \frac{n}{2})$ size for computing Majority function.*
2. *k -SAT can be solved in time $O(\sum_{t=1}^{n/2} b(n, k, t))$.*

Proof.

- 1) Consider a Σ_3^k circuit C that computes Majority function. We will write $C = \bigvee_{i=1}^m F_i$, where each F_i is a k -CNF. None of the F_i s has a satisfying assignment with Hamming weight less than $n/2$. By assumption we can enumerate all satisfying assignments of Hamming weight exactly $n/2$ in expected time $b(n, k, \frac{n}{2})$. This in particular implies that the total number of such satisfying assignments for each F_i is at most $b(n, k, \frac{n}{2})$. Since F_i s should together cover all assignments of Hamming weight exactly $n/2$ and since there are $\binom{n}{n/2}$ such assignments, the claim follows.
- 2) We can trivially check if there is a satisfying assignment of Hamming weight at most $n/2$ in $1 + \sum_{t=1}^{n/2} b(n, k, t)$ steps. In the same number of steps we can check if there is a satisfying assignment of Hamming weight at least $n/2$. ◀

Observe that $b(n, k, \frac{n}{2})$ cannot be too small: define the k -CNF $\text{Maj}_{n,k}$ by partitioning the n variables into sets of size $2(k-1)$ and by including all positive clauses of size k from each of the parts. It is easy to see that every satisfying assignment of this formula must set at least $k-1$ variables in each part to 1 and the total number satisfying assignments with Hamming weight $n/2$ is $2^{(1-O(\log(k)/k))n}$, thus $b(n, k, \frac{n}{2}) \geq 2^{(1-O(\log(k)/k))n}$. It follows that a matching upper bound for $\text{ENUM}(k, \frac{n}{2})$ refutes SETH and gives Σ_3^k lower bound of $2^{\Omega(\frac{\log k}{k}n)}$ for Majority which in turn implies a $2^{\omega(\sqrt{n})}$ unrestricted depth-3 circuit lower bound, breaking a decades long barrier.

1.2 Our contributions

In this paper, we study $\text{ENUM}(3, \epsilon n)$ and obtain new algorithms and lower bounds. Note that this is the first non-trivial instance of $\text{ENUM}(k, t)$, since $\text{ENUM}(2, t)$ can be solved in 2^t steps using a simple extension of the local search algorithm, and it is easy to see that this is tight for $t \leq n/2$ by considering the 2-CNF consisting of t disjoint monotone clauses.

► **Theorem 4** (Main result). $ENUM(3, t)$ can be solved in expected time

1. 3^t , for $t \leq \frac{n}{3}$,
2. $1.164^n \times 1.9023^t$, for $\frac{n}{3} < t \leq \frac{3n}{7}$,
3. $1.1962^n \times 1.7851^t$, for $\frac{3n}{7} < t \leq \frac{n}{2}$.

In particular, $ENUM(3, \frac{n}{2})$ can be solved in expected time 1.598^n .

Consequently, we get

► **Corollary 5.** $\Sigma_3^3(\text{Maj}) \geq 1.251^{n-o(n)}$.

Our lower bound is the best known bound to compute Majority function by Σ_3^3 circuits. Note that $\text{Maj}_{n,3}$ has $6^{n/4} \simeq 1.565^n$ satisfying assignments of Hamming weight $n/2$. Our bound is not too far from the optimal bound and it is a substantial improvement over $3^{n/2} \simeq 1.732^n$.

In the following, we explain how our approach to enumeration differs from the well-known approaches. The k^t algorithm used by [4] which solves (k, t) -SAT is a simple branching procedure. Without loss of generality assume that the initial assignment is all-0. If there is no monotone clause in the formula, then the all-0 assignment satisfies the formula. Otherwise select a monotone clause $C = x_1 \vee \dots \vee x_k$. Then for each x_i , recursively solve $(k, (t-1))$ -SAT for the formula restricted by $x_i = 1$. An obvious weakness of this algorithm is that if the depth of the recursion tree is more than n/k , then some assignments will be considered more than once in the tree thus leading to redundant computation.

Our starting point is to search the recursion tree (which we call a transversal tree following the hypergraph nomenclature) so each satisfying assignment (which we call a transversal in the following) within the ball is visited exactly once. We observe that such a non-redundant search can be conducted with any clause ordering and any fixed ordering of variables within the clause. During the search of the transversal tree, it is easy to decide whether a subtree contains any new satisfying assignments by considering the labels of the child edges of the nodes along the path. If there are no new satisfying assignments in a subtree, we will prune it. It turns out that this approach is isomorphic to the seminal method of Monien-Speckenmeyer [16] where for each i we recursively solve the problem under the restriction $x_1 = \dots = x_{i-1} = 0, x_i = 1$. However, it is not clear how to improve upon the bound obtained by [16]. We show that by choosing the clause ordering carefully and randomly ordering clause variables, a better bound can be obtained. In other words, we will consider *randomized* Monien-Speckenmeyer trees with careful clause ordering. The crux of our contribution is a new analysis of randomized transversal trees.

Connection to hypergraph Turán problems. Recall that a transversal in a hypergraph is a set of vertices that intersects every hyperedge. The recent work [6] gives a connection between depth-3 circuits and transversals². Here we find another connection. Given n, t , and k , let $R^+(n, t, k)$ be the maximum number of transversals of size t in an n -vertex k -graph with no transversal of size $t - 1$. Since a k -graph can be viewed as a monotone k -CNF, our algorithm enumerates minimum size transversals and thus gives new upper bounds for $R^+(n, t, k)$. Mantel's theorem, which is a special case of Turán's seminal theorem, gives the exact value $R^+(n, n - 2, 2) = n^2/4$, and the Turán problem for 3-graphs can be phrased as showing $R^+(n, n - 3, 3) = (5/9 + o(1))n^3$ (see [12] for a thorough survey of this and related

² [6] results are stated in terms of cliques which are dual to transversals.

problems). Our technique allows us to derive new bounds for $R^+(n, t, 3)$, where $t = \Theta(n)$. Although we currently do not get any useful results for $t = n - o(n)$, we hope that our techniques can be extended to make progress for this regime of parameters.

Enumeration algorithms for CNFs with bounded negations. As an additional application of our enumeration techniques, we get the following enumeration algorithm:

► **Theorem 6.** *Let F be a CNF of arbitrary width where either each clause contains at most 3 negative literals or each clause contains at most 3 positive literals. Then, we can enumerate all minimal satisfiable solutions of F in time $O(1.8204^n)$.*

2 Preliminaries

In this section, we introduce concepts and notation that we use for the rest of the paper. Let $F = (X, \mathcal{C})$ be a k -CNF with variable set X and clause set \mathcal{C} . We view the satisfying assignments of F as subsets of variables which are set to 1.

► **Definition 7 (Transversals).** *A set $S \subseteq X$ is a transversal for F if the assignment that sets the variables in S to 1 and the variables in $X \setminus S$ to 0 is a satisfying assignment of F . The size of a transversal S is defined as $|S|$.*

We say S is a minimal transversal for F if no subset of S is a transversal. We call the corresponding satisfying assignment a minimal satisfying assignment.

Our use of transversals for discussing satisfying assignments is motivated by the notion of transversals in hypergraphs. We view a monotone k -CNF (where all literals in every clause are positive) as a k -graph where every hyperedge has size at most k . This leads to a 1-1 correspondence between the transversals of a monotone k -CNF and those of the corresponding hypergraph.

In this paper, we are primarily interested in minimum-size transversals.

► **Definition 8 (Transversal number).** *For a satisfiable k -CNF F , we define transversal number $\tau(F)$ to be the cardinality of the minimum-size transversal of F . We use $\Gamma(F)$ to denote the set of all minimum-size transversals of F and $\#\Gamma(F)$ to denote the cardinality of $\Gamma(F)$.*

Let $t \in [n]$ and $\alpha \in \{0, 1\}^n$ be such that every satisfying assignment of F is at a distance of at least t from α . We reduce the $\text{ENUM}(k, t)$ problem for F from the initial assignment α to the $\text{ENUMZ}(k)$ problem: enumerate all minimum-size transversals of a k -CNF³. Indeed, let G be the k -CNF formula (over the variable set X) where its clause set is obtained from that of F by the following replacement of literals: For each variable $v \in X$, if α sets v to 1, then we swap occurrences of the positive and the negative literals corresponding to v in \mathcal{C} . Otherwise, if α sets v to 0, we leave the corresponding literals as they are. G is also a k -CNF and for all $y \in \{0, 1\}^n$, $G(y) = F(y \oplus \alpha)$. Clearly, there exists a transversal within distance t from 0^n in G if and only if there exists a transversal within distance t from α in F .

We now prove the following useful proposition that allows us to assume all clauses of F have width exactly k .

³ This problem has been previously considered by, e.g. [5] with a different name MIN-ONES k -SAT and a non-trivial algorithm is given independent of $\tau(F)$. Here we focus on running times with fine dependence on $\tau(F)$.

► **Proposition 9.** *For every k -CNF $F = (X, \mathcal{C})$ with $\tau(F) \leq n - k$, there exists a k -CNF G where every clause has width exactly k and $\tau(G) = \tau(F)$ and the set of transversals of G includes the set of transversals of F . Furthermore, if F is monotone, G is monotone.*

Proof. Assume that F has a clause C of width $1 \leq k' < k$. Let F' be the formula obtained from F by removing the clause C and adding a clause $C' = C \cup C''$ for each $S \subseteq X \setminus C$, $|S| = k - k'$, where C'' is a monotone clause over variables in S . The proposition follows since every transversal of F is a transversal of F' and any transversal of F' which is not a transversal of F must have size at least $n - k + 1$. ◀

3 Transversal Trees and Tree Search

In this section, we present an algorithm called TREESEARCH for solving ENUMZ(k), i.e., to enumerate all minimum-size transversals of a satisfiable k -CNF F . Let $t = \tau(F)$ be the transversal number of F . Our algorithm considers a tree, called *transversal tree*, of depth t where each minimum-size transversal corresponds to at least one leaf node at depth t . Our algorithm TREESEARCH traverses the tree to enumerate the leaves at depth t corresponding to minimum-size transversals. However, a leaf at depth t need not correspond to a minimum-size transversal and furthermore two distinct leaves at depth t may correspond to the same minimum-size transversal. For these reasons, enumerating all leaves can take significantly more time than the total number $\#\Gamma(F)$ of minimum-size transversals. We deal with this issue by pruning subtrees so that TREESEARCH would only encounter minimum-size transversals that are not encountered elsewhere. While our pruning approach is isomorphic to that of Monien-Speckenmeyer [16], our analysis and bound crucially depend on our choice of clause ordering and random ordering of the child nodes of the transversal tree. In the following, we define the required concepts and present our constructions.

► **Definition 10** ((k, X) -trees). *For $k \geq 1$ and a set X of variables, a (k, X) -tree is a directed k -ary tree T with a node and edge labeling Q and a tree-edge ordering π which satisfies the following properties.*

1. *Each edge is directed from parent to child. Each non-leaf node has at most k children. Children of a node are ordered from left to right according to π .*
2. *Each (tree) edge e is labeled with a variable $q_e \in X$. Each node v is labeled with a set $Q_v \subseteq X \cup \{\perp\}$.*
3. *For each node, labels of child edges are distinct. If $e = (u, v)$ is an edge, then $Q_v = Q_u \cup \{q_e\}$ or $Q_v = Q_u \cup \{q_e, \perp\}$.*
4. *Labels of edges along any path are distinct.*
5. *All leaves v where $\perp \notin Q_v$ are at the same level.*

Let T be a (k, X) -tree with root r and labeling Q . For node v of T , let T_v denote the subtree T_v of T rooted at v . If u and v are nodes of T such that u is an ancestor of v , P_{uv} denotes the unique path from u to v in T .

► **Definition 11** (Shoot of a tree path). *If u and v are nodes of T such that u is an ancestor of v , the subgraph consisting of all the child edges of the nodes along the unique path from u to v is called the shoot $S_{uv} = S_{uv}^T$ from u to v . In particular $P_{uv} \subseteq S_{uv}$.*

For a path P_{uv} , the labels of the edges along the path are called *path variables* of P_{uv} . For a shoot S_{uv} , labels of the shoot edges are called shoot variables.

- **Definition 12** (Transversal tree). *Let $F = (X, \mathcal{C})$ be a k -CNF on the variable set X . A (k, X) -tree T rooted at r with labeling Q and tree-edge ordering π is a transversal tree for F if*
1. $Q_r = \emptyset$ if F has no empty clause and otherwise $Q_r = \{\perp\}$, and
 2. for every node v , each minimum-size transversal of F which is an extension of Q_v will appear as the label of a leaf in the subtree rooted at v .

It is easy to see that a transversal tree T for a satisfiable k -CNF F has depth $\tau(F)$ and every leaf v of T such that $\perp \notin Q_v$ is at depth $\tau(F)$. We also note that any subtree of a transversal tree is also a transversal tree.

- **Definition 13** (Valid and invalid leaves). *Let T be a transversal tree for a satisfiable k -CNF F . We say that a leaf v of T is valid if Q_v is a minimum-size transversal of F . Otherwise it is invalid.*

For a transversal tree T of a satisfiable k -CNF F , let $\Gamma(T)$ denote the collection of minimum-size transversals associated with the valid leaves of T . The definition of transversal tree implies the following basic fact.

- **Fact 14.** $\Gamma(F) = \Gamma(T)$.

3.1 Construction of Transversal Trees

In this section, we show how to construct transversal trees for a satisfiable k -CNF $F = (X, \mathcal{C})$. The construction produces a labeling Q on nodes and edges. We will not specify a tree-edge ordering in the construction. However, we will later select a left-right ordering where child nodes are ordered randomly and independently for each node. The construction depends on the ordering of clauses. Let Π denote an ordering of the clauses in F .

We start with the tree T with just one node r (the root node) with the label $Q_r = \emptyset$. Assume that we are about to expand a non-leaf node v . By construction, v is at depth less than $\tau(F)$ and $\perp \notin Q_v$. Since v is at depth less than $\tau(F)$, Q_v is not a transversal. Select the first monotone clause $C_v = \{a_1, \dots, a_{k'}\}$ according to the clause order Π , where $k' \leq k$. Such a monotone clause must exist since every clause is non-empty and since otherwise an all-0 assignment will satisfy the formula contradicting the fact that Q_v is not a transversal. Also, it must be the case that $C_v \cap Q_v = \emptyset$ as none of the variables from Q_v can appear in C_v . For each $a \in C_v$,

1. Create a child node v_a for v and label the edge (v, v_a) by a .
2. Simplify the clauses by setting the variables along the path P_{rv_a} to 1. If there is an empty clause, set $Q_{v_a} = Q_v \cup \{a, \perp\}$ and v_a will not be expanded and thus will become a leaf node. Otherwise, label v_a by $Q_{v_a} = Q_v \cup \{a\}$.
3. If the level of the node is $\tau(F)$, make it a leaf node.
4. Order the child nodes of v left-right according to a tree-edge ordering.

- **Proposition 15.** *The tree T with the labeling as described above is a transversal tree for F .*

Proof. Indeed each non-leaf node has at most k children. All leaves v with $\perp \notin Q_v$ must be at the same level $\tau(F)$ since any node v at a level smaller than $\tau(F)$ and does not contain \perp in Q_v can be expanded and since the construction stops at level $\tau(F)$. It is easy to verify that the labeling Q has the requisite properties. For every v with $\perp \notin Q_v$ and every extension Y of Q_v to a minimum-size transversal, there exists a leaf with label Y in the subtree T_v since there is a child edge (v, v') of v with label a for some $a \in Y \cap C_v \neq \emptyset$ where C_v is the clause used to expand v . Inductively we can construct a path from v' to a leaf which ultimately has Y as the label. ◀

3.2 TreeSearch: An Algorithm for Enumerating the Valid Leaves of T

Our goal is to search the transversal tree to enumerate all minimum-size transversals. However, visiting all leaf nodes may take at least $k^{\tau(F)}$ time. To improve the efficiency of the search, we prune the tree during our search while guaranteeing the enumeration of each minimum-size transversal exactly once.

Let F be satisfiable k -CNF. Let Π be a clause ordering for F . Let T be transversal tree for F constructed using Π and some tree-edge ordering π . We note that for any tree-edge ordering π , the edges of any shoot S_{uv} (where u is any ancestor of v) are situated in one of three ways with respect to the tree path from u to v : 1) to the left of the tree path, 2) to the right of the tree path, or 3) along the tree path. Our key insight is that for any node v we can determine whether the subtree T_v potentially contains any *new* minimum-size transversals by considering the labels of the edges in the shoot S_{rv} .

Our TREESEARCH starts with the root node r of T . Assume that we are currently visiting the node v . Let T_v be a subtree of T rooted at v . If v is not a leaf, let C_v be the monotone clause used for expanding the node v (based on the clause order Π) and $a_1, \dots, a_{k'}$ be the ordering of its variables according to π for some $k' \leq k$. For $1 \leq i \leq k'$, let v_{a_i} be the i -th child node of v . The edge $e_i = (v, v_{a_i})$ is labeled with a_i . The search procedure TREESEARCH starting at node v works as follows:

- If v is a leaf, output Q_v if it is a transversal. In any case, return to the parent.
- Otherwise, process the children of v in order. Let $T_i = T_{v_{a_i}}$ be the transversal tree rooted at the child v_{a_i} for $a_i \in C_v$. Prune the subtree T_i if and only if the shoot S_{rv} contains an edge $e' = (u', v')$ where u' is ancestor of v such that $Q_{e_i} = Q_{e'}$, and the edge e' appears to the left of the path P_{rv} . Search the tree T_i if it is not pruned.

► **Fact 16.** *For any clause ordering Π and tree-edge ordering π , TREESEARCH outputs all minimum-size transversals of F exactly once.*

3.3 Canonical Clause Ordering and Random Tree Edge Ordering

The time complexity of TREESEARCH is bounded by the number of leaf nodes it visits. While we know that TREESEARCH outputs all minimum-size transversals without redundancy, it is much less clear how to analyze its complexity. We need two ideas to analyze TREESEARCH to get a good bound. The first idea is a *canonical* clause ordering Π in which a sequence of maximally disjoint monotone clauses will precede all other clauses. The second idea is a random π , that is, a tree-edge ordering that orders the children of every node in the transversal tree uniformly and independently at random.

4 Analysis of TreeSearch for Monotone k -CNFs

Let $F = (X, \mathcal{C})$ be a monotone k -CNF where every clause has exactly k literals. Let $t = \tau(F)$ be the transversal number of F . We assume that $t \leq \frac{3n}{5}$. Let T be a transversal tree for F where T is constructed using a canonical clause ordering Π . The child edges of each of its nodes are randomly ordered from left-right independent of other nodes. Let π denote this random tree-edge ordering. Let r be its root and Q its labeling.

In this section, we analyze TREESEARCH and prove Theorem 4 for the monotone case. We start with a few ideas required to keep track of the effect of the random ordering on pruning. We then build upon them Section 5 to prove Theorem 4 for general k -CNFs.

4.1 Random Tree Edge Ordering and Pruning

► **Definition 17** (Cut event). *We say that an edge $e = (u, v)$ is cut if u has an ancestor u' with a child edge $e' = (u', v')$ where $Q_{e'} = Q_e$ and e' appears to the left of the path P_{ru} according to π .*

We use $\phi(e)$ to denote the event that the edge e is *not* cut. We also use $\phi(P_{uv})$ to denote the event no edge along P_{uv} is cut. We use $\phi(u) = \phi(P_{ru})$ to denote the event that none of the edges along the path from the root to the node u are cut.

► **Definition 18** (Survival probability of paths and nodes). *The survival probability $\sigma(P_{uv})$ of a path P_{uv} is $\mathbf{P}(\phi(P_{uv}))$. The survival probability $\sigma(u)$ of a node u is $\mathbf{P}(\phi(u))$.*

► **Definition 19** (Survival value of a transversal tree). *For a subtree T_u rooted at u , we define $\sigma(T_u) = \sum_{v \text{ is leaf of } T_u} \sigma(v)$ as the survival value of T_u . We write $\sigma(T) = \sigma(T_r)$ where r is the root node.*

Our main tool for upper bounding the expected running time of TREESEARCH is the following basic fact.

► **Fact 20.** *The expected number of leaves visited by TREESEARCH is exactly $\sigma(T)$. In particular $\#\Gamma(F) \leq \sigma(T)$.*

Proof. This follows by definition, since TREESEARCH only visits surviving leaves of T under a random tree-edge ordering. Furthermore, let Y be a minimum-size transversal of F . We argue that $\sum_{v: Q_v=Y} \mathbf{P}(\pi(P_{rv})) = 1$ which implies $\#\Gamma(F) \leq \sigma(T)$. ◀

Our goal is to upper bound $\sigma(T)$ by bounding the survival probabilities of the leaves at depth t . A path survives if and only none of its edges are cut. For an edge to be cut, it is necessary that some ancestor of the edge has a child edge with the same label as that of the edge. We will keep track of repeated edge labels via markings to bound the cut probabilities from below and thereby bounding the survival probabilities from above.

► **Definition 21** (Marking set of an edge). *The marking set $M(e)$ of an edge $e = (u, v)$ in T is the set of nodes $w \neq u$ in the path P_{ru} which have a child edge e' such that $Q_e = Q_{e'}$. We say that the nodes in $M(e)$ mark the edge e . We also say that the nodes in $M(e)$ mark the label of e .*

► **Definition 22** (Marked edges). *An edge $e = (v, u)$ in T is marked if $M(e) \neq \emptyset$.*

Marked edges are precisely those that have a non-zero probability of being cut. In fact, we can calculate the survival probability exactly.

► **Fact 23.** *For an edge e in T , $\sigma(e) = 2^{-|M(e)|}$.*

► **Definition 24.** *Let u be a node in T . For each node v along the path P_{ru} , let $N_u(v) = \{e \in P_{ru} \mid v \in M(e)\}$. $N_u(v)$ is the set of edges along the path P_{ru} marked by v .*

► **Fact 25.** *For any node u in T , the survival probability $\sigma(P_{ru})$ of the path P_{ru} is given by*

$$\sigma(P_{ru}) = \prod_{v: \text{ a node along } P_{ru}} \frac{1}{|N_u(v)| + 1}$$

Proof. P_{ru} survives if and only if for every node $v \in P_{ru}$ and every edge $e \in P_{ru}$ that v marks, the child edge of v with the same label as that of e appears to the right of the path. This happens with probability $\frac{1}{|N_u(v)| + 1}$ and since these events are independent, the claim follows. ◀

4.2 An Analysis of TreeSearch for Monotone 3-CNF

Although Fact 25 gives us a fairly complete picture of the survival probabilities of individual paths in T in terms of the edge markings of the path, we need a few ideas to find nontrivial upper bounds on $\sigma(T)$. The first idea is the concept of a weight which captures the number of marked edges in a path or a shoot.

► **Definition 26** (Weight). *Let P_{uv} be a path in T . The weight of P_{uv} is defined as $W(P_{uv}) \stackrel{\text{def}}{=} |\{e \in P : M(e) \neq \emptyset\}|$, i.e., the number of marked edges along P_{uv} . The weight of a shoot S_{uv} denoted by $W(S_{uv})$ is the number of marked edges in the shoot S_{uv} .*

The following fact provides a lower bound on the weight of each root to leaf shoot in T .

► **Fact 27.** *Every root to leaf shoot S_{ru} in T has a weight of at least $3t - n$.*

Proof. Since the depth of T is t , a root to leaf shoot has $3t$ edges and there are only n distinct edge labels, at least $3t - n$ labels appear at least twice. ◀

► **Definition 28** (Weight of a tree). *We say that a tree has weight w if every root to leaf shoot of the tree has weight at least w .*

► **Definition 29** ($M(w, d)$). *For non-negative integers w and d , let $M(w, d)$ be the maximum survival value over all ternary depth- d transversal trees with weight w .*

We can now upper bound $\sigma(T)$ in terms of $M(w, d)$ exploiting the canonical clause ordering. Our canonical clause ordering Π starts with a maximal collection of disjoint clauses C_1, C_2, \dots, C_m so that all clauses in the formula intersect with at least one of the clauses from these disjoint clauses. We observe that $m \geq \frac{t}{3}$ for monotone F since otherwise by setting all the variables in the monotone clauses C_i , we satisfy F contradicting that the transversal number of F is t .

► **Lemma 30.**

$$\sigma(T) \leq \begin{cases} 3^t & t \leq \frac{n}{3} \\ 3^{\frac{t}{3}} \times M(3t - n, \frac{2}{3}t) & \text{otherwise} \end{cases}$$

Proof. For $t \leq \frac{n}{3}$, we use the trivial upper bound of 1 on the survival probability of paths to conclude that $\sigma(T) \leq 3^t$ as desired.

For $t \geq \frac{n}{3}$, we use the fact that the canonical clause ordering starts with a maximal collection of disjoint clauses C_1, C_2, \dots, C_m where $m \geq \frac{t}{3}$. We observe that for $1 \leq i \leq \frac{t}{3} \leq m$ each node at level i of T is expanded by the same clause C_i . Moreover, none of the child edges of nodes at level $1 \leq i \leq \frac{t}{3} \leq m$ are marked as the corresponding clauses are disjoint. The result follows since for every node u at level $\frac{t}{3} + 1$, the subtree T_u has depth $\frac{2t}{3}$ and the weight of every root to leaf shoot in T_u is at least $3t - n$ minus the number of marked edges from root to $u = 3t - n - 0 = 3t - n$ (Fact 27). ◀

4.2.1 Upper Bounds on $M(w, d)$

Upper bounding $M(w, d)$ for T based on random tree-edge ordering π is challenging. Instead we introduce a different random process π' for T : Each edge e survives with probability p_e independently where $p_e = \lambda$ if e is marked and 1 otherwise, where we define $\lambda \stackrel{\text{def}}{=} \frac{1}{\sqrt{3}}$. The concept of survival probability under π' can be extended to paths and nodes of T . For example, the $\sigma'(P) = \prod_{e \in P} p_e$ is the survival probability of the path P under π' . Similarly, we

17:12 Local Enumeration and Majority Lower Bounds

define the survival value $\sigma'(T)$ of the transversal tree T according to π' as $\sum_{v \text{ is a leaf}} \sigma'(P_{rv})$. We define $M'(w, d)$ as the maximum survival value $\sigma'(T')$ of transversal trees T' of depth d where every root to leaf shoot has weight at least w . The following lemma shows that $\sigma(T) \leq \sigma'(T)$.

► **Lemma 31.** *For a root to leaf path P_{ru} , $\sigma(P) \leq \lambda^{W(P_{ru})} = \sigma'(P)$ which in turn implies $\sigma(T) \leq \sigma'(T)$ and $M(w, d) \leq M'(w, d)$.*

Proof. Given an edge $e \in P_{ru}$, define the contribution y_e of e to the survival probability (according to π) of P_{ru} as

$$y_e \stackrel{\text{def}}{=} \prod_{v \in M(e)} \left(\frac{1}{|N_u(v)| + 1} \right)^{1/|N_u(v)|}$$

where the empty product is considered as 1. By Fact 25, $\sigma(P_{ru}) = \prod_{e: M(e) \neq \emptyset} y_e$. It is then sufficient to show that $y_e \leq p_e$. Observe that $N_u(v)$ can be at most 2 since F is a 3-CNF. For each $v \in M(e)$ with $N_u(v) = 1$, the probability that v does not cut e is exactly $\frac{1}{2}$, independent of other nodes. If $N_u(v) = 2$ for $v \in M(e)$, v marks another edge e' along the path in addition to e . The probability that v cuts neither e nor e' is exactly $\frac{1}{2} \times \frac{2}{3} = \frac{1}{3}$, independent of other nodes. If $N_u(v) = 2$, we regard the probability of each edge surviving as the geometric average λ of the survival probabilities of individual edges. As a consequence, y_e can be written as $(\frac{1}{2})^a (\frac{1}{\sqrt{3}})^b$ for some non-negative integers a and b where a is the number of ancestors v of e such that v marks exactly one edge along the path and b is the number of ancestors v of e such that v marks exactly two edges along the path. We now argue that y_e is at most p_e . If e is not marked, then $a + b = 0$ and hence $y_e = p_e$. If e is marked, we have $a + b > 0$ which implies $y_e \leq \lambda = p_e$. ◀

The following lemma determines $M'(w, d)$.

► **Lemma 32.** *For all $0 \leq d \leq n, 0 \leq w \leq 3d$, we have*

$$M'(w, d) = \begin{cases} (2 + \lambda)^w 3^{d-w} & 0 \leq w \leq d \\ (1 + 2\lambda)^{w-d} (2 + \lambda)^{2d-w} & d \leq w \leq 2d \\ (3\lambda)^{w-2d} (1 + 2\lambda)^{3d-w} & 2d \leq w \leq 3d \end{cases}$$

Lemma 32 already gives an upper bound $\sigma(T) \leq M'(3t - n, t)$. However, taking advantage of Lemma 30, we can improve this bound to establish Theorem 4 for monotone formulas.

4.2.2 Proof of Theorem 4 for monotone formulas

Proof. By Fact 20 and Lemma 31 the expected time of TREESEARCH is bounded by $\sigma'(T)$ (up to polynomial factors). We divide the proof into cases based on the value of t .

Case 1. $t \leq \frac{n}{3}$. Applying Lemma 30 for the case $t \leq \frac{n}{3}$, we get $\sigma(T) \leq 3^t$.

Case 2. $\frac{n}{3} < t \leq \frac{3n}{7}$. $t \leq \frac{3n}{7}$ implies $3t - n \leq \frac{2t}{3}$. We apply Lemma 30 together with Lemma 32 for the case $0 \leq w \leq d$ to get

$$\sigma(T) \leq 3^{\frac{t}{3}} M' \left(3t - n, \frac{2t}{3} \right) = \left(\frac{3}{2 + \lambda} \right)^n \left(\frac{(2 + \lambda)^3}{9} \right)^t \leq 1.164^n \times 1.9023^t$$

Case 3. $\frac{3n}{7} \leq t \leq \frac{n}{2}$. We note that $t \leq \frac{n}{2}$ implies $3t - n \leq t \leq \frac{4t}{3}$ and $t \geq \frac{3n}{7}$ implies $3t - n \geq \frac{2t}{3}$. We apply Lemma 30 together with Lemma 32 for the case $d \leq w \leq 2d$ to get

$$\sigma(T) \leq 3^{\frac{t}{3}} M' \left(3t - n, \frac{2t}{3} \right) = \left(\frac{2 + \lambda}{1 + 2\lambda} \right)^n \left(\left(\frac{3(1 + 2\lambda)^7}{(2 + \lambda)^5} \right)^{1/3} \right)^t \leq 1.1962^n \times 1.7851^t \quad \blacktriangleleft$$

4.2.3 Proof of Lemma 32

Let T' be a tree of depth d and weight $0 \leq w \leq 3d$. It is clear that the survival value $\sigma'(T')$ of T' is determined once the edges are marked consistent with the fact that every root to leaf shoot has weight at least w . We say that a transversal tree T' of depth d and weight w is *normal* if it has the following marking: Let $w = id + j$ where $0 \leq i \leq 3$ and $0 \leq j < d$. Mark $i + 1$ children of every non-leaf node in the first j levels and mark i children for each of the remaining non-leaf nodes. The survival value of normal tree is exactly $((i + 1)\lambda + 2 - i)^j (i\lambda + 3 - i)^{d-j}$. One can easily see that this is given exactly as $M'(w, d)$ as in the statement of the lemma. We will show that normal trees have the largest survival values by induction on d which completes the proof of Lemma 32.

Let T' be a tree of depth d and weight w with the maximum survival value $\sigma'(T')$. Let r be the root of T' . Assume that l_1 children of r are marked. Let T'_1, T'_2 , and T'_3 be the subtrees of T' of depth $d - 1$ and weight $w - l_1$ with r_1, r_2 , and r_3 as their root nodes respectively. T'_1, T'_2 , and T'_3 are normal by induction hypothesis. Assume that l_2 child edges of each of r_i are marked. The survival value of T' is $g_1 g_2 M'(w - (l_1 + l_2), d - 2)$ where $g_1 = (3 - l_1 + l_1 \lambda)$ and $g_2 = (3 - l_2 + l_2 \lambda)$. It is easy to see that if $l_1 + l_2$ is held constant, $g_1 g_2$ is maximized when l_1 and l_2 are as equal as possible. If $|l_1 - l_2| = 1$, the survival value of T' does not change if r has marked l_2 children and each r_i has l_1 marked children, that is, if the number of markings of the first two levels are exchanged.

If $l_1 = l_2$, then T' is normal. If $|l_1 - l_2| \geq 2$, then T' does not have the largest survival value which is a contradiction. We are left with the case that l_1 and l_2 differ by one. If $l_1 < l_2$, we swap l_1 and l_2 without changing the survival value and normality follows from induction. If $l_1 > l_2$, the tree is already normal. Otherwise, its survival value cannot be the maximum.

5 Analysis of TreeSearch for arbitrary 3-CNFs

In this section, we analyze transversal trees for arbitrary 3-CNFs and prove Theorem 4. We will introduce few more ideas in addition to those introduced in Section 4.

Throughout this section, we fix a 3-CNF $F = (X, \mathcal{C})$ and let T be its canonical transversal tree with root node r . Let the number of maximally disjoint width 3 clauses used to develop F be m . Let $X_D \subset X$ denote set of variables that appeared in this set of m disjoint clauses. We also note that unlike in Section 4, the clauses used to develop T may not have width exactly 3.

5.1 Slight modification to canonical ordering and TreeSearch

We extend the conditions on the canonical ordering Π of clauses and how TREESEARCH uses Π . We order clauses in Π so that all maximally disjoint width 3 clauses appear first, followed by all width 3 monotone clauses, followed by all other clauses. For a node u at level below m , we impose that instead of choosing the first unsatisfied monotone clause from Π , u instead chooses an unsatisfied width 3 monotone clause C from Π such that C does not contain any variable $x \in X_D$ that has appeared twice in the shoot S_{ru} . If such a clause does not exist, then C can pick the first unsatisfied monotone clause from Π .

5.2 Fullness and Double marking

We reuse the notion of weight here and observe that all basic facts and basic lemmas from monotone analysis apply here as well. We introduce one more definition related to this:

17:14 Local Enumeration and Majority Lower Bounds

► **Definition 33** (Uniform Weight). *Let $S = S_{uv}$ be a shoot in T of length ℓ . Let a be the number of edges in the shoot S_{uv} . The uniform weight of S denoted by $W^+(S) = W(S) + 3\ell - a$.*

We can similarly find a lower bound to this quantity for every root to leaf path:

► **Fact 34.** *Every root to leaf shoot in T has a uniform weight of at least $3t - n$.*

Proof. Let S be arbitrary root to leaf path with a edges. As there are only n distinct edge labels, at least $a - n$ labels appear at least twice. So, $W(S) \geq a - n$. Since the depth of T is t , we infer that $W^+(S) = W(S) + 3t - a \geq 3t - n$ as desired. ◀

We extend the idea of markings and introduce double markings:

► **Definition 35** (Double marking). *We say an edge $e \in T$ is doubly marked if $|M(e)| \geq 2$. Let $P = P_{uv}$ be a path from u to v . We write $W_{\geq 2}(P)$ to denote number of edges e in P such that $|M(e)| \geq 2$.*

Recall that when proving Lemma 30, we took advantage of the fact that any monotone 3-CNF G with $\tau(G) = t$ will contain $\frac{t}{3}$ disjoint monotone clauses. However, there is no such guarantee for arbitrary 3-CNFs. Observe that if the number of maximally disjoint monotone clauses is small, then many clauses of width 3 will intersect with it and this will cause many edges to be doubly marked. We formalize this intuition and introduce a new parameter called *fullness* that will help keep track of this:

► **Definition 36** (Fullness). *Let u be arbitrary node at level $\geq m$ in T . Let u_m be the node at level m along the path P_{ru} . Then, fullness of the shoot S_{ru} is defined as*

$$Y(S_{ru}) := |\{x \in X_D \setminus Q_{u_m} : \exists e = (a, b) \in S_{ru}, \text{depth}(a) \geq m, Q_e = x\}|,$$

i.e., the number of variables in X_D that are not along the path in the first m levels and appear as labels of some edge in the shoot after level m . For arbitrary nodes u, v where u is ancestor of v and u appears at level $\geq m$, we define $Y(S_{uv}) = Y(S_{rv}) - Y(S_{ru})$.

This parameter is useful because every node after level m will have at least one edge that will either “add to” fullness or at least one edge that will have marking set of size at least 2. The edges that are “doubly marked” will contribute very little to the recursion. Moreover, we will see that fullness of any root to leaf shoot is at most $2m$. As m is small, very few nodes will be such that they will not contain any doubly marked edges. This is our key insight and we formally prove this now.

► **Fact 37.** *Let M denote the set of all monotone clauses of width 3 in F . Then, every clause $C \in M$ must contain at least one variable x such that $x \in X_D$.*

► **Lemma 38.** *Let $u \in T$ be a node at level greater than m . Then, at least one of the following must be true:*

1. u has at most 2 edges going out.
2. u has one edge e going out such that $|M(e)| \geq 2$.
3. u has one edge e going out such that $Q_e \in X_D$ and $|M(e)| = 1$.

Proof. Set variables that are part of Q_u to 1 and let F' be the simplified 3-CNF. Say case 1 does not happen. Then, the monotone clause C used to develop edges out of u has width 3 and so, C is also present in F . By Fact 37, u contains an edge e such that $Q_e = x$ and $x \in X_D$. If $|M(e)| \geq 2$, then case 2 is satisfied and if $|M(e)| = 1$, then case 3 is satisfied. ◀

5.3 An Analysis of TreeSearch for arbitrary 3-CNF

We extend Lemma 31 for arbitrary 3-CNFs taking into double markings into account.

► **Lemma 39.** *Let T be a transversal tree and let $P = P_{ru}$ be a path starting from root r . Then $\sigma(P) \leq (\frac{1}{\sqrt{3}})^{W^+(P)+W_{\geq 2}(P)}$.*

Proof. For a marked edge $e \in P$, we define the contribution of e as

$$q_e := \prod_{v \in M(e)} \left(\frac{1}{|N_u(v)| + 1} \right)^{1/|N_u(v)|}.$$

By Fact 25, $\sigma(P) = \prod_{e: M(e) \neq \emptyset} q_e$. It is then sufficient to show that $q_e \leq \lambda$ for every marked edge e . Note that q_e can be written as $(\frac{1}{2})^a (\frac{1}{\sqrt{3}})^b$, for some non-negative integers a and b such that $a + b \geq |M(e)|$. This quantity is at most $\frac{1}{\sqrt{3}}$ if $|M(e)| = 1$ and is at most $\frac{1}{3}$ if $|M(e)| \geq 2$. Finally, we observe that e contributes 1 to $W(P)$ if $|M(e)| \geq 1$ and contributes 1 to $W_{\geq 2}(P)$ if $|M(e)| \geq 2$. ◀

► **Definition 40** ($NM(w, d, y)$). *For non-negative integers w, d, y , define $NM(w, d, y)$ to be the maximum of sum of survival probabilities of leaves over depth- d transversal trees T for 3-CNFs. Moreover, for every root to leaf shoot S , $W^+(S) \geq w$ and $Y(S) \leq y$.*

5.3.1 Proving Theorem 4

We will show the following as our main lemma:

► **Lemma 41.** *Let F be a 3-CNF over n variables with $\tau(F) = t \leq \frac{n}{2}$. Then for any canonical transversal tree T for F , it holds that*

$$\sigma(T) \leq \begin{cases} 3^t & t \leq \frac{n}{3} \\ 3^{\frac{t}{3}} \times M'(3t - n, \frac{2t}{3}) & \text{otherwise} \end{cases}$$

where $M'(w, d)$ is the same bound we obtained in Section 4.

Using this, Theorem 4 follows from Lemma 41 by using the exact same argument as in the Proof of monotone case of Theorem 1.

Our main lemma will make use of the following bounds on $NM(w, d, y)$:

► **Lemma 42.** *For all $0 \leq d \leq n, 0 \leq w \leq 3d, 0 \leq y \leq d$, it holds that:*

$$NM(w, d, y) \leq \begin{cases} (2 + \lambda)^y (2 + \lambda^2)^{d-y} & 0 \leq w \leq d \\ (2 + \lambda)^{y-(w-d)} (1 + 2\lambda)^{w-d} (2 + \lambda^2)^{d-y} & d \leq w \leq d + y \\ (1 + 2\lambda)^y (2 + \lambda^2)^{2d-w} (1 + \lambda + \lambda^2)^{w-d-y} & d + y \leq w \leq 2d \\ (1 + 2\lambda)^{y-(w-2d)} (3\lambda)^{w-2d} (1 + \lambda + \lambda^2)^{d-y} & 2d \leq w \leq 2d + y \\ (3\lambda)^y (1 + \lambda + \lambda^2)^{3d-w} (2\lambda + \lambda^2)^{w-2d-y} & 2d + y \leq w \leq 3d \end{cases}$$

Moreover, for $y \geq d$:

$$NM(w, d, y) \leq M'(w, d)$$

where $M'(w, d)$ is from Section 4.

Using this, we now prove our main lemma, which yields Theorem 4 as desired.

17:16 Local Enumeration and Majority Lower Bounds

Proof of Lemma 41 assuming Lemma 42. If $t \leq \frac{n}{3}$, then observe that T has at most 3^t leaves and we trivially bound $\sigma(T) \leq 3^t$.

For $t \geq \frac{n}{3}$, we proceed by considering the maximal set of disjoint monotone width 3 clauses in F used to develop the first m levels of T . For every node $u \in T$ at level m , let the subtree rooted at u be T_u . We can bound $\sigma(T_u) \leq NM(w, d, y)$ where $d = t - m$, $w = 3t - n$, $y = 2m$. Hence, $\sigma(T) \leq 3^m NM(w, d, y)$.

If $m \geq \frac{t}{3}$, then $y = 2m \geq t - m = d$. Applying Lemma 42, we infer that

$$\begin{aligned} \sigma(T) &\leq 3^m M'(3t - n, t - m) \\ &= 3^{t/3} \left(3^{m-t/3} M' \left(3t - n, \frac{2t}{3} - \left(m - \frac{t}{3} \right) \right) \right) \\ &\leq 3^{t/3} M' \left(3t - n, \frac{2t}{3} \right) \end{aligned}$$

and we infer the claim.

So, we assume that $m \leq \frac{t}{3}$ and try to find the value of m which will maximize $\sigma(T)$. Notice that in this case, $y \leq d$ and so, we can't directly reduce to the case of $M'(w, d)$. As $t \leq \frac{n}{2}$, it must be that $w = 3t - n \leq t \leq t + m \leq d + y$. This implies $w \leq d + y$. We now take two cases based on value of w and apply Lemma 42 for the case of $y \leq d$.

Case 1. $0 \leq w \leq d$. In this case, we see that:

$$\begin{aligned} \sigma(T) &\leq 3^m NM(3t - n, t - m, 2m) \\ &\leq 3^m (2 + \lambda)^{2m} (2 + \lambda^2)^{t-3m} \\ &= (2 + \lambda^2)^t \left(\frac{(3)(2 + \lambda)^2}{(2 + \lambda^2)^3} \right)^m \end{aligned}$$

Here, the fraction has value > 1 and so is maximized when m is maximized, i.e., when $m = \frac{t}{3}$ in which case:

$$\begin{aligned} \sigma(T) &\leq 3^{t/3} (2 + \lambda)^{2t/3} \\ &= 3^{t/3} M' \left(3t - n, \frac{2t}{3} \right) \end{aligned}$$

Here the last equality follows by considering the case of $0 \leq w \leq d$ for $M'(w, d)$.

Case 2. $d \leq w \leq d + y$. In this case, we see that:

$$\begin{aligned} \sigma(T) &\leq 3^m NM(3t - n, t - m, 2m) \\ &\leq 3^m (2 + \lambda)^{n-2t+m} (1 + 2\lambda)^{2t-n+m} (2 + \lambda^2)^{t-3m} \\ &= \left(\frac{2 + \lambda}{1 + 2\lambda} \right)^{n-2t} (2 + \lambda^2)^t \left(\frac{(3)(2 + \lambda)(1 + 2\lambda)}{(2 + \lambda^2)^3} \right)^m \end{aligned}$$

Here, the rightmost fraction is > 1 and so is maximized when m is maximized, i.e., when $m = \frac{t}{3}$ in which case:

$$\begin{aligned} \sigma(T) &\leq 3^{t/3} (2 + \lambda)^{n-5t/3} (1 + 2\lambda)^{7t/3-n} \\ &= 3^{t/3} M' \left(3t - n, \frac{2t}{3} \right) \end{aligned}$$

Here the last equality follows by considering the case of $d \leq w \leq 2d$ for $M'(w, d)$ (we can do this as $y \leq d$ and hence, $w \leq 2d$).

Thus, in either case, we exactly recover the monotone bound as desired. \blacktriangleleft

5.3.2 Upper bounds on $NM(w, d, y)$

As done in monotone analysis, we let $\lambda \stackrel{\text{def}}{=} \frac{1}{\sqrt{3}}$. Our goal is to prove Lemma 42. We introduce a recurrence relation $L(w, d, y)$ that we argue will upper bound $NM(w, d, y)$.

► **Definition 43** ($L(w, d, y)$). We define $L(w, d, y) : \mathbb{N}^3 \rightarrow \mathbb{R}$ recursively as follows:

$$L(w, 0, y) = \begin{cases} 1 & w \leq 0 \\ 0 & w > 0 \end{cases}$$

For $w \leq 3d, 0 \leq d \leq n$, and $y \geq 1$, define $L(w, d, y)$ as:

$$L(w, d, y) = \max\{(2 + \lambda)L(w - 1, d - 1, y - 1), \\ (1 + 2\lambda)L(w - 2, d - 1, y - 1), \\ 3\lambda L(w - 3, d - 1, y - 1)\}$$

For $w \leq 3d, 0 \leq d \leq n$, and $y = 0$, define $L(w, d, 0)$ as:

$$L(w, d, 0) = \max\{(2 + \lambda^2)L(w - 1, d - 1, 0), \\ (1 + \lambda + \lambda^2)L(w - 2, d - 1, 0), \\ (2\lambda + \lambda^2)L(w - 3, d - 1, 0)\}$$

We claim that $L(w, d, y)$ gives a good bound on $M(w, d, y)$.

► **Proposition 44.** For all $0 \leq d \leq n, 0 \leq w \leq 3d, 0 \leq y$, it holds that: $NM(w, d, y) \leq L(w, d, y)$

We will show the following bound on $L(w, d, y)$.

► **Lemma 45.** For all $0 \leq d \leq n, 0 \leq w \leq 3d, 0 \leq y \leq d$, it holds that:

$$L(w, d, y) \leq \begin{cases} (2 + \lambda)^y (2 + \lambda^2)^{d-y} & 0 \leq w \leq d \\ (2 + \lambda)^{y-(w-d)} (1 + 2\lambda)^{w-d} (2 + \lambda^2)^{d-y} & d \leq w \leq d + y \\ (1 + 2\lambda)^y (2 + \lambda^2)^{2d-w} (1 + \lambda + \lambda^2)^{w-d-y} & d + y \leq w \leq 2d \\ (1 + 2\lambda)^{y-(w-2d)} (3\lambda)^{w-2d} (1 + \lambda + \lambda^2)^{d-y} & 2d \leq w \leq 2d + y \\ (3\lambda)^y (1 + \lambda + \lambda^2)^{3d-w} (2\lambda + \lambda^2)^{w-2d-y} & 2d + y \leq w \leq 3d \end{cases}$$

Moreover, for $y \geq d$:

$$L(w, d, y) \leq M'(w, d)$$

where $M'(w, d)$ is from Section 4.

Combining Proposition 44 and Lemma 45, Lemma 42 trivially follows.

5.3.3 Proving $NM(w, d, y) \leq L(w, d, y)$

Using Lemma 38 and Lemma 39, we come up with a recurrence for $NM(w, d, y)$ and show it's bounded by $L(w, d, y)$, proving Proposition 44.

Proof of Proposition 44. Recall that in the canonical ordering, all width 3 monotone clauses appear first and remaining clauses appear later. After exhausting the width 3 monotone clauses, the remaining clauses that we develop in the transversal tree have width at most 2.

17:18 Local Enumeration and Majority Lower Bounds

Towards this, for non-negative integers w, d : let $M_2(w, d)$ be the maximum sum of survival probabilities over all transversal trees for 2-CNFs where every root to leaf path has uniform weight at least w . Recall that uniform weight is defined with respect to 3-CNFs and we continue using that definition. We get various recurrences by considering cases on number of marked edges out of the root node (0 or 1 or 2) and by observing that some cases are dominated by others (such as various cases of width 1 clauses). The remaining recurrences that are not dominated by any other recurrence are the following:

$$M_2(w, d) \leq \max\{(2)M_2(w-1, d-1), \\ (1+\lambda)M_2(w-2, d-1), \\ 2\lambda M_2(w-3, d-1)\}$$

By induction, we infer that $M_2(w, d) \leq L(w, d, 0)$.

We now develop a recurrence for $NM(w, d, 0)$. Recall that in canonical ordering, either we exhaust all width 3 monotone clause and reach $M_2(w, d)$, or we develop width 3 monotone clause. Observe that Lemma 38 guarantees that every node in such a tree must have an edge e coming out of it such that $|M(e)| \geq 2$. Taking cases on the number of marked edges coming out of the root node and whether root node has 3 or at most 2 edges coming out, we get many recurrences. However certain recurrences are dominated by others and the remaining recurrences that are not dominated by any other recurrence are as follows:

$$NM(w, d, 0) \leq \max\{(2+\lambda^2)NM(w-1, d-1, 0), \\ (1+\lambda+\lambda^2)NM(w-2, d-1, 0), \\ (2\lambda+\lambda^2)NM(w-3, d-1, 0), \\ M_2(w, d)\}$$

By induction, we again infer that $NM(w, d, 0) \leq L(w, d, 0)$.

We now develop a recurrence for $NM(w, d, y)$. We again take advantage of the fact that in canonical ordering, either we exhaust all width 3 monotone clause and reach $M_2(w, d)$, or we develop width 3 monotone clause. Moreover, amongst width 3 clauses, canonical ordering causes either y to decrease by at least 1 or we exhaust such clauses and all remaining clauses have the property that a node developed using such a clause will have an outgoing edge e such that $|M(e)| \geq 2$.

We get many recurrences for $NM(w, d, y)$ by considering cases on number of marked edges (1 or 2 or 3) out of the root node, number of marked edges that cause Y to decrease (1 or 2 or 3), various combinations of number of double marked edges (1 or 2 or 3), whether the root node has at most 2 edges coming out, and whether the root node has no edges that cause Y to decrease by at least 1. Notice that if the root node has no edges that cause Y to decrease by at least 1, then by clause ordering, no other width 3 clause can cause Y to decrease and hence, we are in case $NM(w, d, 0)$. Lastly, we observe that certain recurrences are dominated by others. The remaining recurrences that are not dominated by any other recurrences are the following:

$$NM(w, d, y) \leq \max\{(2+\lambda)NM(w-1, d-1, y-1), \\ (1+2\lambda)NM(w-2, d-1, y-1), \\ 3\lambda NM(w-3, d-1, y-1), \\ NM(w, d, 0), \\ M_2(w, d)\}$$

By induction, utilizing the fact that $L(w, d, y) \geq L(w, d, 0)$, we again infer that $NM(w, d, y) \leq L(w, d, y)$ as desired. \blacktriangleleft

5.3.4 Upper bound on $L(w, d, 0)$

We first show Lemma 45 for the special case of $y = 0$:

► **Lemma 46.** *For all $0 \leq d \leq n, 0 \leq w \leq 3d$, it holds that:*

$$L(w, d, 0) \leq \begin{cases} (2 + \lambda^2)^d & 0 \leq w \leq d \\ (2 + \lambda^2)^{2d-w} (1 + \lambda + \lambda^2)^{w-d} & d \leq w \leq 2d \\ (1 + \lambda + \lambda^2)^{3d-w} (2\lambda + \lambda^2)^{w-2d} & 2d \leq w \leq 3d \end{cases}$$

Proof. Let $G_1, G_2, G_3, G : \mathbb{N}^2 \rightarrow \mathbb{R}$ be defined as:

$$\begin{aligned} G_1(w, d) &= (2 + \lambda^2)^d \\ G_2(w, d) &= (2 + \lambda^2)^{2d-w} (1 + \lambda + \lambda^2)^{w-d} \\ G_3(w, d) &= (1 + \lambda + \lambda^2)^{3d-w} (2\lambda + \lambda^2)^{w-2d} \\ G(w, d) &= \min\{G_1(w, d), G_2(w, d), G_3(w, d)\} \end{aligned}$$

For $1 \leq i \leq 3$, define P_i to be the set of pairs (w, d) such that $d \geq 0$ and $w \in [(i-1)d, id]$. We will show the following two propositions:

► **Proposition 47.** *For all $1 \leq i \leq 3$, and all $(w, d) \in P_i : G(w, d) = G_i(w, d)$.*

► **Proposition 48.** *For all $1 \leq i \leq 3$ and all $(w, d) \in P_i : L(w, d, 0) \leq G(w, d)$.*

We observe that Proposition 47 and Proposition 48 together imply our claim.

Proof of Proposition 47. The result follows immediately from the following claims:

▷ **Claim 49.** $G_1(w, d) \leq G_2(w, d)$ if and only if $w \leq d$, with equality when $w = d$.

▷ **Claim 50.** $G_2(w, d) \leq G_3(w, d)$ if and only if $w \leq 2d$, with equality when $w = 2d$.

Claim 49 holds because:

$$\frac{G_1(w, d)}{G_2(w, d)} = \left(\frac{2 + \lambda^2}{1 + \lambda + \lambda^2} \right)^{w-d}$$

which is greater than 1 if and only if $w > d$. Claim 50 holds because:

$$\frac{G_2(w, d)}{G_3(w, d)} = \left(\frac{(1 + \lambda + \lambda^2)^2}{(2\lambda + \lambda^2)(2 + \lambda^2)} \right)^{w-2d}$$

which is greater than 1 if and only if $w > 2d$. ◀

Proof of Proposition 48. We consider cases on value of w and in every case, induct on d and apply Definition 43 to infer the claim.

Case 1. Assume $(w, d) \in P_1$.

$$\begin{aligned} L(w, d, 0) &\leq \max\{(2 + \lambda^2)G(w-1, d-1, 0), \\ &\quad (1 + \lambda + \lambda^2)G(w-2, d-1, 0), \\ &\quad (2\lambda + \lambda^2)G(w-3, d-1, 0)\} \\ &\leq \max\{(2 + \lambda^2)G_1(w-1, d-1), (1 + \lambda + \lambda^2)G_1(w-2, d-1), \\ &\quad (2\lambda + \lambda^2)G_1(w-3, d-1)\} \\ &= G_1(w, d) \max\{1, (1 + \lambda + \lambda^2)/(2 + \lambda^2), (2\lambda + \lambda^2)/(2 + \lambda^2)\} \\ &= G_1(w, d) \\ &= G(w, d) \end{aligned}$$

The last equality follows by applying Proposition 47 for the case $(w, d) \in P_1$.

17:20 Local Enumeration and Majority Lower Bounds

Case 2. Assume $(w, d) \in P_2$.

$$\begin{aligned}
L(w, d, 0) &\leq \max\{(2 + \lambda^2)G(w - 1, d - 1, 0), \\
&\quad (1 + \lambda + \lambda^2)G(w - 2, d - 1, 0), \\
&\quad (2\lambda + \lambda^2)G(w - 3, d - 1, 0)\} \\
&\leq \max\{(2 + \lambda^2)G_2(w - 1, d - 1), (1 + \lambda + \lambda^2)G_2(w - 2, d - 1), \\
&\quad (2\lambda + \lambda^2)G_2(w - 3, d - 1)\} \\
&= G_2(w, d) \max\{1, 1, (2\lambda + \lambda^2)(2 + \lambda^2)/(1 + \lambda + \lambda^2)^2\} \\
&= G_2(w, d) \\
&= G(w, d)
\end{aligned}$$

The last equality follows by applying Proposition 47 for the case $(w, d) \in P_2$.

Case 3. Assume $(w, d) \in P_3$.

$$\begin{aligned}
L(w, d, 0) &\leq \max\{(2 + \lambda^2)G(w - 1, d - 1, 0), \\
&\quad (1 + \lambda + \lambda^2)G(w - 2, d - 1, 0), \\
&\quad (2\lambda + \lambda^2)G(w - 3, d - 1, 0)\} \\
&\leq \max\{(2 + \lambda^2)G_3(w - 1, d - 1), (1 + \lambda + \lambda^2)G_3(w - 2, d - 1), \\
&\quad (2\lambda + \lambda^2)G_3(w - 3, d - 1)\} \\
&= G_3(w, d) \max\{(2 + \lambda^2)(2\lambda + \lambda^2)/(1 + \lambda + \lambda^2)^2, 1, 1\} \\
&= G_3(w, d) \\
&= G(w, d)
\end{aligned}$$

The last equality follows by applying Proposition 47 for the case $(w, d) \in P_3$. ◀

◀

5.3.5 Upper bound on $L(w, d, y)$

We are finally ready to give general bounds on $L(w, d, y)$:

Proof of Lemma 45. Notice that if $y \geq d$, then if we try and unravel the recurrence, no path can lead to the case $y = 0, d > 0$. Hence, y plays no role in restricting the recurrence and $L(w, d, y)$ follows the same recurrence as $M'(w, d)$, yielding the claim.

For $y \leq d$, we proceed by first defining $H_1, H_2, H_3, H_4, H_5, H : \mathbb{N}^3 \rightarrow \mathbb{R}$ as follows:

$$\begin{aligned}
H_1(w, d, y) &= (2 + \lambda)^y (2 + \lambda^2)^{d-y} \\
H_2(w, d, y) &= (2 + \lambda)^{y-(w-d)} (1 + 2\lambda)^{w-d} (2 + \lambda^2)^{d-y} \\
H_3(w, d, y) &= (1 + 2\lambda)^y (2 + \lambda^2)^{2d-w} (1 + \lambda + \lambda^2)^{w-d-y} \\
H_4(w, d, y) &= (1 + 2\lambda)^{y-(w-2d)} (3\lambda)^{w-2d} (1 + \lambda + \lambda^2)^{d-y} \\
H_5(w, d, y) &= (3\lambda)^y (1 + \lambda + \lambda^2)^{3d-w} (2\lambda + \lambda^2)^{w-2d-y} \\
H(w, d, y) &= \min\{H_2(w, d, y), H_3(w, d, y), H_4(w, d, y), H_5(w, d, y)\}
\end{aligned}$$

For $1 \leq i \leq 5$, define $Q_i \subset \mathbb{N}^3$ as follows:

$$\begin{aligned}
Q_1 &= \{(w, d, y) \in \mathbb{N}^3 : 0 \leq w \leq d + y\} \\
Q_2 &= \{(w, d, y) \in \mathbb{N}^3 : d \leq w \leq d + y\} \\
Q_3 &= \{(w, d, y) \in \mathbb{N}^3 : d + y \leq w \leq 2d\} \\
Q_4 &= \{(w, d, y) \in \mathbb{N}^3 : 2d \leq w \leq 2d + y\} \\
Q_5 &= \{(w, d, y) \in \mathbb{N}^3 : 2d + y \leq w \leq 3d\}
\end{aligned}$$

We will show the following propositions that together imply our claim:

► **Proposition 51.** For all $1 \leq i \leq 5$, and all $(w, d, y) \in Q_i : H(w, d, y) = H_i(w, d, y)$.

► **Proposition 52.** For all $1 \leq i \leq 5$ and all $(w, d, y) \in Q_i : L(w, d, y) \leq H(w, d, y)$.

We will in fact use Proposition 51 in the proof of Proposition 52. Hence, we prove the former first:

Proof of Proposition 51. The result follows immediately from the following claims:

▷ **Claim 53.** $H_1(w, d, y) \leq H_2(w, d, y)$ if and only if $w \leq d$, with equality when $w = d$.

▷ **Claim 54.** $H_2(w, d, y) \leq H_3(w, d, y)$ if and only if $w \leq d + y$, with equality when $w = d + y$.

▷ **Claim 55.** $H_3(w, d, y) \leq H_4(w, d, y)$ if and only if $w \leq 2d$, with equality when $w = 2d$.

▷ **Claim 56.** $H_4(w, d, y) \leq H_5(w, d, y)$ if and only if $w \leq 2d + y$, with equality when $w = 2d + y$.

Claim 53 holds because:

$$\frac{H_1(w, d, y)}{H_2(w, d, y)} = \left(\frac{2 + \lambda}{1 + 2\lambda} \right)^{w-d}$$

which is greater than 1 if and only if $w > d$. Claim 54 holds because:

$$\frac{H_2(w, d, y)}{H_3(w, d, y)} = \left(\frac{(1 + 2\lambda)(2 + \lambda^2)}{(2 + \lambda)(1 + \lambda + \lambda^2)} \right)^{w-d-y}$$

which is greater than 1 if and only if $w > d + y$. Claim 55 holds because:

$$\frac{H_3(w, d, y)}{H_4(w, d, y)} = \left(\frac{(1 + 2\lambda)(1 + \lambda + \lambda^2)}{(2 + \lambda^2)(3\lambda)} \right)^{w-2d}$$

which is greater than 1 if and only if $w > 2d$. Claim 56 holds because:

$$\frac{H_4(w, d, y)}{H_5(w, d, y)} = \left(\frac{(3\lambda)(1 + \lambda + \lambda^2)}{(1 + 2\lambda)(2\lambda + \lambda^2)} \right)^{w-2d-y}$$

which is greater than 1 if and only if $w > 2d + y$. ◀

We prove our final proposition:

Proof of Proposition 52. We observe that for $y = 0$, our claim follows from Lemma 46. We use this fact in the inductive argument below and only consider cases where $y \geq 1$. We consider cases on value of w and in every case, induct on $d + y$ and apply Definition 43 to infer the claim.

Case 1. Assume $(w, d, y) \in Q_1$ and $y \geq 1$.

$$\begin{aligned} L(w, d, y) &= \max\{(2 + \lambda)H(w - 1, d - 1, y - 1), (1 + 2\lambda)H(w - 2, d - 1, y - 1), \\ &\quad (3\lambda)H(w - 3, d - 1, y - 1)\} \\ &\leq \max\{(2 + \lambda)H_1(w - 1, d - 1, y - 1), (1 + 2\lambda)H_1(w - 2, d - 1, y - 1), \\ &\quad (3\lambda)H_1(w - 3, d - 1, y - 1)\} \\ &= H_1(w, d, y) \max\left\{1, \frac{1 + 2\lambda}{2 + \lambda}, \frac{3\lambda}{2 + \lambda}\right\} \\ &= H_1(w, d, y) \\ &= H(w, d, y) \end{aligned}$$

The last equality follows by applying Proposition 51 for the case $(w, d, y) \in Q_1$.

17:22 Local Enumeration and Majority Lower Bounds

Case 2. Assume $(w, d, y) \in Q_2$ and $y \geq 1$.

$$\begin{aligned}
 L(w, d, y) &\leq \max\{(2 + \lambda)L(w - 1, d - 1, y - 1), (1 + 2\lambda)L(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)L(w - 3, d - 1, y - 1)\} \\
 &\leq \max\{(2 + \lambda)H_2(w - 1, d - 1, y - 1), (1 + 2\lambda)H_2(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)H_2(w - 3, d - 1, y - 1)\} \\
 &= H_2(w, d, y) \max\left\{1, 1, \frac{(3\lambda)(2 + \lambda)}{(1 + 2\lambda)^2}\right\} \\
 &= H_2(w, d, y) \\
 &= H(w, d, y)
 \end{aligned}$$

The last equality follows by applying Proposition 51 for the case $(w, d, y) \in Q_2$.

Case 3. Assume $(w, d, y) \in Q_3$ and $y \geq 1$.

$$\begin{aligned}
 L(w, d, y) &\leq \max\{(2 + \lambda)L(w - 1, d - 1, y - 1), (1 + 2\lambda)L(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)L(w - 3, d - 1, y - 1)\} \\
 &\leq \max\{(2 + \lambda)H_3(w - 1, d - 1, y - 1), (1 + 2\lambda)H_3(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)H_3(w - 3, d - 1, y - 1)\} \\
 &= H_3(w, d, y) \max\left\{\frac{(2 + \lambda)(1 + \lambda + \lambda^2)}{(1 + 2\lambda)(2 + \lambda^2)}, 1, \frac{(2 + \lambda^2)(3\lambda)}{(1 + 2\lambda)(1 + \lambda + \lambda^2)}\right\} \\
 &= H_3(w, d, y) \\
 &= H(w, d, y)
 \end{aligned}$$

The last equality follows by applying Proposition 51 for the case $(w, d, y) \in Q_3$.

Case 4. Assume $(w, d, y) \in Q_4$ and $y \geq 1$.

$$\begin{aligned}
 L(w, d, y) &\leq \max\{(2 + \lambda)L(w - 1, d - 1, y - 1), (1 + 2\lambda)L(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)L(w - 3, d - 1, y - 1)\} \\
 &\leq \max\{(2 + \lambda)H_4(w - 1, d - 1, y - 1), (1 + 2\lambda)H_4(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)H_4(w - 3, d - 1, y - 1)\} \\
 &= H_4(w, d, y) \max\left\{\frac{(3\lambda)(2 + \lambda)}{(1 + 2\lambda)^2}, 1, 1\right\} \\
 &= H_4(w, d, y) \\
 &= H(w, d, y)
 \end{aligned}$$

The last equality follows by applying Proposition 51 for the case $(w, d, y) \in Q_4$.

Case 5. Assume $(w, d, y) \in Q_5$ and $y \geq 1$.

$$\begin{aligned}
 L(w, d, y) &\leq \max\{(2 + \lambda)L(w - 1, d - 1, y - 1), (1 + 2\lambda)L(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)L(w - 3, d - 1, y - 1)\} \\
 &\leq \max\{(2 + \lambda)H_5(w - 1, d - 1, y - 1), (1 + 2\lambda)H_5(w - 2, d - 1, y - 1), \\
 &\quad (3\lambda)H_5(w - 3, d - 1, y - 1)\} \\
 &= H_5(w, d, y) \max\left\{\frac{(2 + \lambda)(2\lambda + \lambda^2)^2}{(3\lambda)(1 + \lambda + \lambda^2)^2}, \frac{(1 + 2\lambda)(2\lambda + \lambda^2)}{(3\lambda)(1 + \lambda + \lambda^2)}, 1\right\} \\
 &= H_5(w, d, y) \\
 &= H(w, d, y)
 \end{aligned}$$

The last equality follows by applying Proposition 51 for the case $(w, d, y) \in Q_5$. ◀

◀

6 Satisfiability for CNFs with bounded negations

We now use TREESEARCH to give an enumeration algorithm for class of CNFs with arbitrary width and bounded negations in each clause.

We will use the following well known estimate of binomial coefficients:

► **Proposition 57.** *Let $H_2 : (0, 1) \rightarrow (0, 1)$ be the binary entropy function defined as $H_2(x) = -x \log_2(x) + (1 - x) \log_2(1 - x)$. Then, for $k \leq n/2$, it holds that: $\sum_{i=0}^k \binom{n}{i} \leq \text{poly}(n) 2^{nH_2(k/n)}$.*

Proof of Theorem 6. Without loss of generality we assume each clause F contains at most 3 positive literals. Indeed, if every clause in F contains at most 3 negative literals, then we can negate every literal in every clause and consider the resultant CNF. This CNF is satisfiable if and only if the original CNF was satisfiable. Moreover, the new CNF has the property that each clause contains at most 3 positive literals.

Let $c = 0.71347$. Then, we use TREESEARCH to to enumerate all minimal satisfiable assignments of weight at most cn . We then exhaustively go over all assignments α with weight at least cn and check whether α satisfies F and output such minimal α .

The runtime of the exhaustive procedure is

$$\text{poly}(n) \sum_{i=cn}^n \binom{n}{i} \leq \text{poly}(n) 2^{nH_2(c)} \leq O(1.8204^n)$$

Notice that when we develop the transversal tree, we only develop positive monotone clauses. Any positive monotone clauses that we encounter during the TREESEARCH procedure for F must have width at most 3 as each clause contains at most 3 positive literals. Hence, the resultant transversal tree T is still a ternary tree. So, every root to leaf shoot S must have weight at least $3t - n$ where $t = cn$. We do not put any lower bound on Y for any such shoot and so, we set $y = \infty$. Then, the runtime of TREESEARCH upto polynomial factors is bounded by $NM(3(cn) - n, cn, \infty) \leq M'((3c - 1), cn)$. We observe that $c \leq 3c - 1 \leq 2c$ and so, we are in the regime where $w \leq d \leq 2d$. Thus,

$$\begin{aligned} M'((3c - 1)n, cn) &\leq \left(\left(1 + \frac{2}{\sqrt{3}} \right)^{2c-1} \left(2 + \frac{1}{\sqrt{3}} \right)^{1-c} \right)^n \\ &\leq 1.8204^n \end{aligned}$$

Hence, the runtime of our algorithm is indeed $O(1.8204^n)$ as desired. ◀

7 Conclusion

We gave a new non-trivial algorithm for $\text{ENUM}(3, \frac{n}{2})$: given an n -variable 3-CNF with no satisfying assignment of Hamming weight less than $\frac{n}{2}$, we can enumerate all satisfying assignments of Hamming weight exactly $\frac{n}{2}$ in expected time 1.598^n . Several fascinating questions with major consequences remain open. Here we list the most pressing.

1. We already mentioned that $\text{ENUM}(3, \frac{n}{2})$ cannot be solved in less than 1.565^n steps. Close this gap.
2. Can our approach produce significant improvements for k -CNFs with $k > 3$?

It seems that to make progress towards resolving these problems, deeper analysis of the structure of k -CNFs will be required.

References

- 1 Kazuyuki Amano. Depth-three circuits for inner product and majority functions. In Satoru Iwata and Naonori Kakimura, editors, *34th International Symposium on Algorithms and Computation, ISAAC 2023, December 3-6, 2023, Kyoto, Japan*, volume 283 of *LIPICs*, pages 7:1–7:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ISAAC.2023.7.
- 2 Tobias Brüggemann and Walter Kern. An improved deterministic local search algorithm for 3-SAT. *Theor. Comput. Sci.*, 329(1-3):303–313, 2004. doi:10.1016/j.tcs.2004.08.002.
- 3 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Comput. Complex.*, 24(2):333–392, 2015. doi:10.1007/s00037-015-0100-0.
- 4 Evgeny Dantsin, Andreas Goerdt, Edward A. Hirsch, Ravi Kannan, Jon M. Kleinberg, Christos H. Papadimitriou, Prabhakar Raghavan, and Uwe Schöning. A deterministic $(2 - 2/(k + 1))^n$ algorithm for k -SAT based on local search. *Theor. Comput. Sci.*, 289(1):69–83, 2002. doi:10.1016/S0304-3975(01)00174-8.
- 5 Fedor V. Fomin, Serge Gaspers, Daniel Lokshtanov, and Saket Saurabh. Exact algorithms via monotone local search. *J. ACM*, 66(2):8:1–8:23, 2019. doi:10.1145/3284176.
- 6 Peter Frankl, Svyatoslav Gryaznov, and Navid Talebanfard. A variant of the VC-dimension with applications to depth-3 circuits. 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 72:1–72:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.ITCS.2022.72.
- 7 Alexander Golovnev, Alexander S. Kulikov, and R. Ryan Williams. Circuit depth reductions. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPICs*, pages 24:1–24:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.ITCS.2021.24.
- 8 Thomas Dueholm Hansen, Haim Kaplan, Or Zamir, and Uri Zwick. Faster k -SAT algorithms using biased-PPSZ. 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 578–589. ACM, 2019. doi:10.1145/3313276.3316359.
- 9 Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth-three circuits. *Comput. Complex.*, 5(2):99–112, 1995. doi:10.1007/BF01268140.
- 10 Timon Hertli. Breaking the PPSZ barrier for unique 3-SAT. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 600–611. Springer, 2014. doi:10.1007/978-3-662-43948-7_50.
- 11 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0 . 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 961–972. SIAM, 2012. doi:10.1137/1.9781611973099.77.
- 12 Peter Keevash. Hypergraph Turán problems. In *Surveys in combinatorics 2011*, volume 392 of *London Math. Soc. Lecture Note Ser.*, pages 83–139. Cambridge Univ. Press, Cambridge, 2011.
- 13 Konstantin Kutzkov and Dominik Scheder. Using CSP to improve deterministic 3-sat. *CoRR*, abs/1007.1166, 2010. arXiv:1007.1166.
- 14 Victor Lecomte, Prasanna Ramakrishnan, and Li-Yang Tan. The composition complexity of majority. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 19:1–19:26. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.CCC.2022.19.
- 15 Andrea Lincoln and Adam Yedidia. Faster random k -CNF satisfiability. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPICs*, pages 78:1–78:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.ICALP.2020.78.

- 16 Burkhard Monien and Ewald Speckenmeyer. Solving satisfiability in less than 2^n steps. *Discret. Appl. Math.*, 10(3):287–295, 1985. doi:10.1016/0166-218X(85)90050-2.
- 17 Robin A. Moser and Dominik Scheder. A full derandomization of schöning’s k -SAT algorithm. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 245–252. ACM, 2011. doi:10.1145/1993636.1993670.
- 18 Christos H. Papadimitriou. On selecting a satisfying truth assignment (extended abstract). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 163–169. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185365.
- 19 Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k -SAT. *J. ACM*, 52(3):337–364, 2005. doi:10.1145/1066100.1066101.
- 20 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. *Chic. J. Theor. Comput. Sci.*, 1999, 1999. URL: <http://cjtc.cs.uchicago.edu/articles/1999/11/contents.html>.
- 21 Ramamohan Paturi, Michael E. Saks, and Francis Zane. Exponential lower bounds for depth three boolean circuits. *Comput. Complex.*, 9(1):1–15, 2000. doi:10.1007/PL00001598.
- 22 Dominik Scheder. PPSZ is better than you think. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 205–216. IEEE, 2021. doi:10.1109/FOCS52979.2021.00028.
- 23 Dominik Scheder and Navid Talebanfard. Super strong ETH is true for PPSZ with small resolution width. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPICs*, pages 3:1–3:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.CCC.2020.3.
- 24 Uwe Schöning. A probabilistic algorithm for k -SAT based on limited local search and restart. *Algorithmica*, 32(4):615–623, 2002. doi:10.1007/s00453-001-0094-7.
- 25 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *Mathematical foundations of computer science (Proc. Sixth Sympos., Tatranská Lomnica, 1977)*, pages 162–176. Lecture Notes in Comput. Sci., Vol. 53, 1977.
- 26 Nikhil Vyas and R. Ryan Williams. On super strong ETH. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing – SAT 2019 – 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings*, volume 11628 of *Lecture Notes in Computer Science*, pages 406–423. Springer, 2019. doi:10.1007/978-3-030-24258-9_28.
- 27 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.


Pseudorandomness, Symmetry, Smoothing: I

Harm Derksen ✉

Northeastern University, Boston, MA, USA

Peter Ivanov ✉

Northeastern University, Boston, MA, USA

Chin Ho Lee ✉ 

North Carolina State University, Raleigh, NC, USA

Emanuele Viola ✉

Northeastern University, Boston, MA, USA

Abstract

We prove several new results about bounded uniform and small-bias distributions. A main message is that, small-bias, even perturbed with noise, does not fool several classes of tests better than bounded uniformity. We prove this for threshold tests, small-space algorithms, and small-depth circuits. In particular, we obtain small-bias distributions that

- achieve an optimal lower bound on their statistical distance to any bounded-uniform distribution. This closes a line of research initiated by Alon, Goldreich, and Mansour in 2003, and improves on a result by O'Donnell and Zhao.
- have heavier tail mass than the uniform distribution. This answers a question posed by several researchers including Bun and Steinke.
- rule out a popular paradigm for constructing pseudorandom generators, originating in a 1989 work by Ajtai and Wigderson. This again answers a question raised by several researchers. For branching programs, our result matches a bound by Forbes and Kelley.

Our small-bias distributions above are symmetric. We show that the xor of any two symmetric small-bias distributions fools any bounded function. Hence our examples cannot be extended to the xor of two small-bias distributions, another popular paradigm whose power remains unknown. We also generalize and simplify the proof of a result of Bazzi.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases pseudorandomness, k -wise uniform distributions, small-bias distributions, noise, symmetric tests, thresholds, Krawtchouk polynomials

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.18

Funding *Harm Derksen*: Partially supported by NSF grant DMS 2147769.

Peter Ivanov: Supported by NSF grant CCF-2114116.

Emanuele Viola: Supported by NSF grant CCF-2114116.

Acknowledgements CHL thanks Salil Vadhan and Terence Tao for helpful discussions.

1 Introduction

A distribution D over $\{-1, 1\}^n$ is (ε, k) -biased if for every $S \subseteq [n]$ of size $0 < |S| \leq k$ we have $|\mathbb{E}[D^S]| \leq \varepsilon$, where $D^S := \prod_{i \in S} D_i$. If $\varepsilon = 0$ then any k bits are uniform and D is called k -wise uniform; if $k = n$ then D is called ε -biased. The study of these distributions permeates and precedes theoretical computer science. They were studied already in the 40's [52], are closely related to universal hash functions [17], error-correcting codes (see e.g. [35]), and in their modern guise were introduced in the works [5, 23, 47].

(ε, k) -biased distributions behave like the uniform distribution in that several prominent tests cannot distinguish the two distributions.



© Harm Derksen, Peter Ivanov, Chin Ho Lee, and Emanuele Viola;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 18; pp. 18:1–18:27

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



► **Definition 1.** A test $f: \{-1, 1\}^n \rightarrow [-1, 1]$ is δ -fooled by a distribution D we have $|\mathbb{E}[f(U)] - \mathbb{E}[f(D)]| \leq \delta$, where U is the uniform distribution.

At the same time, (ε, k) -biased distributions can be sampled efficiently from a short seed. The combination of these facts enables many applications in algorithm design, coding theory, pseudorandomness, and more. For background we refer the reader to [60, 35, 64], where bounds on seed lengths are also discussed.

To generate k -wise uniformity, seed length $ck \log n$ is sufficient, and necessary for $k < n^c$; while for ε -biased seed length $c \log(n/\varepsilon)$ is sufficient and necessary. In this paper, as in [64], every occurrence of “ c ” denotes a possibly different positive real number. The notation “ c_x ” for parameter(s) x indicates that this number may depend on x and only on x . Replacing “ c ” with $O(1)$ everywhere is consistent with one common interpretation of the big-Oh notation.

It is known that any ε -biased distribution is close to a k -wise uniform distribution in total variation (a.k.a. statistical, L_1 , etc.) distance [6, 4, 50].

► **Lemma 2** (Theorem 1.1 [50]). Any (ε, k) -biased distribution is $((\frac{e^3 n}{k})^{k/2} \varepsilon)$ -close to a k -wise uniform distribution in total variation distance.

Hence, any property enjoyed by k -wise uniform distributions is inherited by distributions with bias n^{-ck} . Unsurprisingly, that is precisely the bias for which the seed length of the latter matches that of the former, as discussed above. The question arises as to which tests can be fooled with a larger bias, which would result in shorter seed length.

► **Question 3.** Which tests can distinguish some ε -biased distribution from every k -wise uniform distribution, for an ε suitably larger than the bound in Lemma 2?

For a concrete setting, one can think e.g. $k = 10 \log n$ and $\varepsilon = n^{-100}$, or any $\varepsilon = n^{-o(\log n)}$.

Question 3 is a computational version of the classic question of the statistical distance between ε -biased and k -wise uniform distributions, studied in [6, 4, 50]. Lemma 2 shows that for small ε , *no test*, efficient or not, can distinguish the distributions. Those works also give lower bounds in various ranges of parameters, which means that in those ranges, there exists some ε -biased distribution such that every k -wise uniform distribution can be distinguished from it by some test. However, the arguments in these papers either do not apply to the tests we consider below or for bias ε larger than n^{-k} , which is the regime of interest here. These works are discussed more below.

A trivial test which cannot distinguish between small-bias and k -wise uniform distributions in the sense of Question 3 is parity. By definition, the bias of parity is at most ε , which is ε -close to the bias of the uniform distribution, which is 0. And the uniform distribution is in particular k -wise uniform.

However, the answer to Question 3 was not known for various other classes of tests of interest. To our knowledge, it was not known for symmetric or even threshold tests (a.k.a. tail, deviation, concentration bounds, etc.). In particular, Bun and Steinke posed the following question in [16].

In this work, we focused on understanding the limits of k -wise independent distributions. Gopalan et al. [31] gave a much more sophisticated generator with nearly optimal seed length. But could simple, natural pseudorandom distributions, such as small-bias spaces, give strong tail bounds themselves?

More concretely, the following question has been asked by several researchers. We use $1^\top x$ to denote the sum $\sum_{i=1}^n x_i$ of $x \in \{-1, 1\}^n$, and B to denote the binomial distribution $1^\top U$.

► **Question 4.** *Is it true that for every a , there exists b such that $\Pr[1^\top D \geq t] \leq \Pr[B \geq t] + 1/n^a$ for every n^{-b} -biased distribution D and $t = \sqrt{n \log n}$?*

The answer to Question 4 was known to be negative for $t = 0$ (corresponding to the majority function): One can take D to be uniform on strings of weight 0 modulo 3, see [8]. The answer was also known to be positive when $t > n^{1/2+\varepsilon}$ for a constant ε because all the relevant quantities are small enough; formally combine Corollary 28 with Lemma 2. But for other values of t closer to \sqrt{n} the answer was less clear.

Smoothed tests and distributions

A main focus of this paper is on *smoothed tests* and *smoothed distributions*, which are tests and distributions perturbed by *noise*.

► **Definition 5.** N_ρ is the noise distribution on $\{-1, 1\}^n$, where each bit is independently set to uniform with probability $1 - \rho$ and 1 otherwise. We write $D \cdot N_\rho$ for the coordinate-wise product of D and N_ρ , which corresponds to bit-wise xor over $\{0, 1\}$. Note $x \cdot N_1 = x$ and $x \cdot N_0 = U$, for any x .

For a test f and a distribution D , a *smoothed distribution* is defined as $D \cdot N_\rho$ and a *smoothed test* is defined as $T_\rho f(x) := \mathbb{E}[f(x \cdot N_\rho)]$, for some retention rate $\rho \in [0, 1]$. Note that $\mathbb{E}[f(D \cdot N_\rho)] = \mathbb{E}[T_\rho f(D)]$ and we will use both viewpoints interchangeably throughout.

Note that smoothing does not increase the distance of any two distributions, with respect to any class of tests which is closed under shifts. So distinguishing smoothed distributions is at least as hard as distinguishing the corresponding (non-smooth) distributions. A main motivation for considering smoothed tests and distributions comes from several paradigms for constructing pseudorandom generators (PRGs) that combine (ε, k) -biased distributions in different ways. These paradigms have been proposed in the last 15 years or so and are discussed next; for additional background, we refer the readers to the recent monograph [35].

Small-bias plus (pseudorandom) noise

This paradigm goes back to Ajtai and Wigderson [3], but saw no further work until it was revived by Gopalan, Meka, Reingold, Trevisan, and Vadhan [32]. It has been used in a number of subsequent works including [30, 54, 56, 34, 42, 21, 29, 45, 40, 28, 22]. In particular, this paradigm gave rise to PRGs with near-optimal seed lengths for several well-studied classes of tests, including combinatorial rectangles [32, 40] and read-once AC^0 formulas [27, 28].

The Ajtai–Wigderson paradigm comprises several steps. A main step in this paradigm requires fooling (the average of) a random restriction of tests with a pseudorandom distribution. (This can also be viewed as constructing a fractional pseudorandom generator [19, 20, 18].) The works by Haramaty, Lee, and Viola [41, 34, 42, 40] have reinterpreted the notion of “random restrictions” as *perturbing or xor-ing a small-bias distribution with noise*. The perspective of noise has proved influential and is maintained in several following works, including the present one.

This perspective of noise has been used to prove a variety of new results in areas ranging from communication complexity [34], coding theory [34, 55], Turing machines [63], and one-way small-space computation [29, 45].

In particular, building on the proof in [34], Forbes and Kelley [29] significantly improved the parameters in [34] and obtained pseudorandom generators with seed length $c \log^3 n$ that fool one-way logspace computation. The main new feature of their result over the classic generator by Nisan [48] is that the order in which the input is read by the computation is

arbitrary. A main step in the result in [29] is showing that $c \log n$ -wise uniformity xor-ed with noise fools logspace. After their work, a natural question, asked independently by several researchers, is whether one can improve the seed length to $o(\log^3 n)$ by replacing $c \log n$ -wise uniformity with polynomial bias.

► **Question 6.** *Does $1/\text{poly}(n)$ -bias plus noise fool one-way logspace?*

A positive answer would give improved generators for small-space algorithms from $c \log^3 n$ to $c \log^2 n$, bringing the parameters of the result in [29], which works in any order, in alignment with the classic fixed-order result of Nisan [48].

In fact, the answer to Question 6 was not known even for the special case of one-way logspace algorithms which compute *symmetric tests*; or for the even more restricted class of *threshold tests* that have the form $\mathbb{1}(1^\top x \geq t)$, for any bias larger than $n^{-\log n}$.

Xor-ing small-bias distributions

Starting with [13], researchers have considered the bit-wise xor of several independent copies of small-bias distributions. The work [41] draws a connection with the previous paradigm, showing that for a *special class* of small-bias distributions, the paradigms are equivalent.

These distributions – the xor of several small-bias distributions – appear to be significantly more powerful than a single small-bias distribution, while retaining a modest seed length. We refer to [49, 62, 35, 64] for background.

Despite several attempts [12, 46, 41], no definitive counterexample to this paradigm has been found; its power remains unknown.

1.1 Our results

In this work we prove several new results on (ε, k) -biased distributions. A main message is that, for several natural classes of tests, *small-bias distributions are no better than bounded uniformity*, i.e., we provide new information about Question 3, and answer Question 4 and Question 6.

To set the stage, we start with showing that k -wise uniformity plus noise does fool symmetric functions with error 2^{-ck} . Note that noise is necessary, for parity is not fooled even by $(n-1)$ -wise uniformity. And even for threshold tests, the error would be polynomial [26, 10] rather than exponential in k .

► **Theorem 7.** *Let D be a distribution on $\{-1, 1\}^n$ that is either*

- (i) *$(2k)$ -wise uniform, or*
- (ii) *$(ck/n)^{4k}$ -biased.*

Let $f: \{-1, 1\}^n \rightarrow [-1, 1]$ be symmetric. Then $|\mathbb{E}[f(U)] - \mathbb{E}[f(D \cdot N_\rho)]| \leq c \cdot (e\rho)^{k/2}$.

Theorem 7.i follows from [29] when $k \geq c \log n$, but their proof does not apply to smaller k . Our result applies to any k , and this will be critical.

Theorem 7.ii follows from Theorem 7.i via the following simple extension of Lemma 2, which we establish by taking noise into account. (A direct application of Lemma 2 would give a larger error of 2^{-ck} .)

► **Lemma 8.** *Let D be an (ε, k) -biased distribution on $\{-1, 1\}^n$. Then $D \cdot N_\rho$ is $((\frac{e^3 \rho n}{k})^{k/2} \varepsilon)$ -close to a k -wise uniform distribution in total variation distance.*

A natural question is whether larger bias suffices in Theorem 7.ii. A main result in this work is that it does not, even for threshold tests. The best possible bound for small-bias distributions is in fact obtained by combining Theorem 7.i with the generic Lemma 8.

► **Theorem 9.** *There exists a $(ck/n)^k$ -biased distribution D such that $\Pr[1^\top(D \cdot N_\rho) \geq 2\sqrt{kn}] \geq \Pr[B \geq 2\sqrt{kn}] + (c\rho)^{2k}$ for every $\rho \in [0, 1]$.*

In fact, the distribution D in Theorem 9 (and in Theorem 10 below) is simultaneously $(2k - 1)$ -wise uniform.

Theorem 9 gives a negative answer to Question 4. Specifically, setting ρ to be a constant and $k = \log n$ we obtain bias $1/n^{\omega(1)}$ but the error is $\geq 1/n^c$.

Note that our negative answer holds even with noise, while an answer to Question 4 was not known even for plain small-bias distributions. This makes our results stronger. Moreover, we do not know of a simpler proof if one does not care about noise. Indeed, we obtained several different proofs of essentially Theorem 9, see [25]. In all these proofs (including the one presented here) the small-bias distribution D itself can be written as $D := D' \cdot N_{c\rho}$, that is, by adding noise to another distribution. Further adding noise to D then comes at little cost, as already pointed out in [41], see Claim 22. We also mention that some of these proofs cover wider range of parameters, and provide new information even for bounded uniformity. We refer to [25] for more on this.

Combining Theorem 9 with Theorem 7, one immediately obtains a *smoothed* threshold test which distinguishes some n^{-k} -bias distribution from any ck -wise uniform distribution, answering Question 3 for such tests.

For general symmetric tests and the same distribution D , we prove a stronger result improving on the classic line of works in [6, 4, 50] and finally matching Lemma 8.

► **Theorem 10.** *There exists a $(ck/n)^k$ -biased distribution D such that for every $\rho \in [0, 1]$ the following holds. There exists a symmetric function $f: \{-1, 1\}^n \rightarrow \{0, 1\}$ such that for every $(2k)$ -wise uniform distribution D_{2k} ,*

$$\mathbb{E}[f(1^\top(D \cdot N_\rho))] \geq \mathbb{E}[f(D_{2k})] + \left(\frac{c\rho}{\sqrt{\log(1/2\rho)}}\right)^{2k}.$$

Again, this result was not known even without noise. Note that Theorem 10 implies the same separation without noise simply setting $\rho := 1$. But the other way around is not clear.

An interesting question is whether one could prove a single result that implies both Theorem 10 and Theorem 9.

From Theorem 9, we derive several consequences on small-space computation and small-depth circuits.

One-way small space. We give a negative answer to Question 6.

► **Corollary 11.** *For any $\rho \in (0, 1]$, there is a distribution D on $\{-1, 1\}^n$ that is $n^{-c \log_{1/\rho} n}$ -biased and a threshold-of-thresholds $T: \{-1, 1\}^n \rightarrow \{0, 1\}$ such that $\mathbb{E}[T(U)] - \mathbb{E}[T(D \cdot N_\rho)] \geq 1/3$. In particular, there is a read-once branching program T of width n^c for which the inequality holds.*

Corollary 11, in combination with Lemma 8, matches a result in [29], which shows that the error is ρ^{ck} for k -wise uniform when $k \geq c \log_{1/\rho} n$.

Proof of Corollary 11 from Theorem 9. We divide the input into \sqrt{n} blocks, and in each block sample an independent copy of the $n^{-k/2}$ -biased distribution from Theorem 9 on \sqrt{n} bits. The resulting distribution has the required properties, since the bias of a test that spans multiple blocks equals the product of the biases in each block.

18:6 Pseudorandomness, Symmetry, Smoothing: I

In each block, a suitable threshold tells $D \cdot N_\rho$ from uniform with advantage $(c\rho)^k \geq n^{-0.1}$ for $k = c \log_{1/\rho} n$. A threshold of \sqrt{n} such blocks is sufficient to boost the advantage to constant.

Finally, this threshold-of-thresholds computation can be implemented with $c \log n$ bits of space, by simply maintaining two counters. ◀

What may have made this problem harder is that it was not clear what distinguishing bound one should expect in Theorem 9. One may be tempted to aim for larger advantage, perhaps independent from k . But as we showed in Theorem 7, this is false: k -wise uniformity plus noise fools thresholds with error 2^{-ck} . One can then ask if k -wise uniformity fools with error 2^{-ck} more general classes of tests, like threshold of thresholds. Corollary 11 shows this is also false.

Constant-depth circuits. Next we discuss a negative result for fooling the circuit class AC^0 . It is known that polylogarithmic independence or quasi-polynomial bias fools AC^0 [7, 53, 15, 57], and these bounds are nearly tight. But despite attempts [41] it was not known if logarithmic uniformity plus noise, or polynomial bias plus noise suffices. We show that bias $n^{-\omega(1)}$ is necessary.

► **Corollary 12.** *For any $\rho \in (0, 1]$ there is a distribution D on $\{-1, 1\}^n$ that is $n^{-c_\rho \log \log n}$ biased and an AC^0 circuit C of size n^c and depth c such that $\mathbb{E}[C(U)] - \mathbb{E}[C(D \cdot N_\rho)] \geq 1/3$.*

The proof is similar to before, except we take blocks of polylogarithmic length, and set $k = c_\rho \log \log n$. The threshold in each block can be computed in AC^0 since it's only on polylogarithmic number of bits. By our setting of k , the advantage in each block is polylogarithmic, and so computing approximate majority [2] (cf. [61]) suffices to have constant advantage.

Sum of small-bias distributions. A next natural question is whether our counterexamples can be extended to the xor of two small-bias distributions. We show that they cannot. Specifically, our small-bias distributions are symmetric, and we show that the sum of two such distributions fools any function (symmetric or not).

► **Theorem 13.** *Let D_1 and D_2 be two independent n^{-20k} -biased, symmetric distributions on $\{-1, 1\}^n$. Then $|\mathbb{E}[f(D_1 \cdot D_2)] - \mathbb{E}[f]| \leq c_k(n^{-0.3k})$ for any function $f: \{-1, 1\}^n \rightarrow [-1, 1]$.*

In fact, we prove stronger results. We show that to fool any symmetric function it suffices for one of the two distributions to be symmetric (Corollary 33). In fact, this holds even if one of the two distributions is any fixed string x with $1^\top x \leq n^{0.99}$ (Theorem 34); and we complement this with a result showing that the result is false if $1^\top x$ is large. This is in Section 4.

Typical shifts. We generalize and simplify the proof of a result by Bazzi [9]. We first discuss his result. Let $C \subseteq \{0, 1\}^n$ be a binary linear code with minimum distance $k + 1$ and maximum distance $n - k - 1$. Let U_{C^\perp} be the uniform distribution on the dual code of C , and $\mathbf{u} \sim \{0, 1\}^n$ be a uniform string. Bazzi [9] showed that for most shifts \mathbf{u} , the distribution $\mathbf{u} + U_{C^\perp}$ fools any symmetric function $f: \{0, 1\}^n \rightarrow \{0, 1\}$:

$$\mathbb{E}_{\mathbf{u}}[|\mathbb{E}[f(\mathbf{u} + U_{C^\perp})] - \mathbb{E}[f]|] \leq (k/n)^{ck}.$$

It follows from the distance properties of C that U_{C^\perp} fools all parity tests of size at most k and at least $n - k$ (with no error). We show that in fact the conclusion above holds for every distribution D that fools such parity tests, without requiring the distribution to be linear.

► **Theorem 14.** *Let D be a distribution on $\{-1, 1\}^n$ such that $\mathbb{E}[D^S] = 0$ for every subset S of size $\ell \in [1, k] \cup [n - k, n]$, and $\mathbf{u} \sim \{0, 1\}^n$ be a uniform string. For every symmetric function $f: \{-1, 1\}^n \rightarrow [-1, 1]$,*

$$\mathbb{E}_{\mathbf{u}}[|\mathbb{E}[f(\mathbf{u} \cdot D)] - \mathbb{E}[f]|] \leq 6(k/n)^{\frac{k-1}{4}}.$$

For context, we note that the condition on fooling large parity tests is necessary, as otherwise, one can consider the uniform distribution D on strings with the parity 0 (say), which is $(n - 1)$ -wise uniform, and for every shift u , the parity of $u + D$ (which is a symmetric function) is simply the parity of u .

Also, note that no fixed shift \mathbf{u} suffices, for else one can shift D by this \mathbf{u} and give a counterexample. This does not contradict the results discussed above about shifting symmetric small-bias distributions because the shift of a symmetric distribution is not in general symmetric.

1.2 Krawtchouk polynomials

All our results rely on bounds for the (shifted) Krawtchouk polynomials \overline{K} , which can be defined by

$$\overline{K}(k, t) := \sum_{|S|=k} z^S,$$

where $z \in \{-1, 1\}^n$ is any string such that $1^\top z = t$, and z^S is the product of the bits of z indexed by S . It can be shown that this is a degree- k polynomial in t .

This is a classic quantity (cf. [39]) and the bounds we need do not seem well known.

To illustrate the bounds we find it convenient to define the normalized version of \overline{K} ,

$$N\overline{K}(k, t) := \frac{\overline{K}(k, t)}{\binom{n}{k}}$$

and its “with replacement” counterpart

$$NR(k, t) := \mathbb{E}_{f: [k] \rightarrow [n]} \left[\prod_{i \in [k]} z_{f(i)} \right] = (t/n)^k,$$

where f is uniform.

Note that $N\overline{K}$ is the same as NR conditioned on f having no collisions – via the correspondence $S = \{f(i) : i \in [k]\}$ – which is the same as saying that the images of f are picked from $[n]$ without replacement.

The bounds on $N\overline{K}$ (and hence \overline{K}) can now be understood as approximations to $NR(k, t)$.

First we prove a lower bound, needed for Theorem 9. The proof is short and follows from known results on Krawtchouk polynomials. However, we are unable to find the result we need in the literature.

▷ **Claim 15.** $N\overline{K}(k, t) \geq (\frac{t}{2n})^k = NR(k, t)/2^k$ for $t \geq 2\sqrt{k(n-k)}$.

For Theorem 10 we need an upper bound. We could use a bound which to our knowledge appeared first in [11]. Since the proof in the latter is somewhat technical, we also give a new simple proof of a stronger bound, stated next, building on the recent work by Tao [59]. We could also use [11] for Theorem 7, but we would get a bound of the form $(a\rho)^k$ for an unspecified constant a . The stronger bound in Corollary 16 proved here gives a better dependence on ρ and gets us closer to the natural bound of ρ^k , which is currently not clear.

► **Corollary 16.** *For every $1 \leq k \leq n$, we have $|N\overline{K}(k, t)| \leq (\frac{k}{n} + \frac{t^2}{n^2})^{k/2}$.*

Note this is similar to $NR(k, t)$ except for the extra term k/n .

For other results we need additional bounds which hold in regimes where the above bounds are loose, such as when k is close to $n/2$ and t is close to 0. To illustrate, let n be even and $1^\top x = t := 0$, corresponding to $x \in \{-1, 1\}^n$ being a balanced string. Note that $\overline{K}(k, 0)$ is the k -th coefficient of the polynomial $(1 - x^2)^{n/2}$, which is $(-1)^{k/2} \binom{n/2}{k/2} \mathbb{1}(k \text{ is even})$. In this case, Corollary 16 gives an upper bound of $\binom{n}{k} (k/n)^{k/2}$. In particular, when $k = n/2$, the bound is roughly $2^{3n/4}$. By contrast, the bound given next by Proposition 17 is $2^{\frac{n}{2}H(k/n)}$, which when $k = n/2$ becomes $2^{n/2}$.

► **Proposition 17.** *Let $k = \beta n$ and $t = (1 - 2\alpha)n$. We have $\log_2 |\overline{K}(k, t)| \leq \frac{n}{2} (1 + H(\beta) - H(\alpha))$, where $H(\alpha) = -\alpha \log_2(\alpha) - (1 - \alpha) \log_2(1 - \alpha)$ is the binary entropy function.*

A similar bound also appears in [51, Lemma 2.1]. Using the estimate $H(1/2 + \gamma) \geq 1 - 4\gamma^2$ for $\gamma \in [0, 1/2]$, we have the following corollary.

► **Corollary 18.** $|\overline{K}(k, t)| \leq 2^{\frac{n}{2}(H(\frac{k}{n}) + \frac{t^2}{n^2})}$.

In Section 6 we prove bounds more general than the above.

2 Small-bias plus noise is far from bounded uniformity

In this section we prove Theorems 9 and 10. We build on the work by O’Donnell and Zhao [50]. In particular, we use the same distribution D . However, jumping ahead, our analyses differ from [50] in three ways, each of which is critical for us:

1. while we analyze the same symmetric test in Theorem 10, we use a new and explicit threshold test in Theorem 9;
2. the distinguishing advantages in Theorems 9 and 10 are explicit and stronger. This relies on our use of (and bounds for) Krawtchouk polynomials, instead of the Hermite approximation in [50];
3. we take noise into account.

We now define D and derive some properties of it. Then in the next subsections the theorems are proved in turn.

► **Definition 19.** *For a parameter $\alpha \in [0, \frac{1}{5e}]$, define $D_\alpha : \{-1, 1\}^n \rightarrow \mathbb{R}$ to be*

$$D_\alpha(x) := 2^{-n} \left(1 + \alpha^k \binom{n}{2k} \right)^{-\frac{1}{2}} \sum_{|S|=2k} x^S \quad \text{for every } x \in \{-1, 1\}^n.$$

Note that the right hand side is the Fourier transform of D_α , and thus $2^n \widehat{D}_\alpha(\emptyset) = \sum_{x \in \{-1, 1\}^n} D_\alpha(x) = 1$. We now show that for $\alpha \leq 1/(5e)$, we have $D_\alpha(x) \geq 0$ for every $x \in \{-1, 1\}^n$ and thus it is a distribution.

▷ Claim 20. For $\alpha \leq 1/(5e)$, we have $D_\alpha(x) \geq 0$ for every x .

Proof. The key observation is that as a degree- $(2k)$ polynomial in t , the zeros of $\overline{K}(2k, t)$ all lies within $|t| \leq 2\sqrt{(2k-1)(n-2k+2)} \leq 2\sqrt{2kn}$ [43] (see also [39, Section 5]). As $2k$ is even, we know that when x is the all-1 or all-(-1) string (i.e., $1^\top x \in \{n, -n\}$), we have $\overline{K}(2k, 1^\top x) := \sum_{|S|=2k} x^S > 0$. So $\overline{K}(2k, 1^\top x)$ can only be negative when $|1^\top x| \leq 2\sqrt{2kn}$. In this interval, using Corollary 16 and $\alpha \leq 1/(5e)$, we have

$$\alpha^k \binom{n}{2k}^{-\frac{1}{2}} \left| \sum_{|S|=2k} x^S \right| \leq \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \binom{n}{2k} \left(\frac{10k}{n} \right)^k \leq (5e\alpha)^k \leq 1. \quad \triangleleft$$

By the above, D_α is a well-defined distribution whenever $\alpha \leq 1/(5e)$. The following claim is immediate.

▷ Claim 21. For $\alpha \leq 1/(5e)$, D_α is a distribution that is $(2k-1)$ -wise uniform, $\alpha^k \binom{n}{2k}^{-1/2}$ -biased, and $(\alpha e^3/2)^k$ -close to $(2k)$ -wise uniform.

Proof. The first two properties follow directly from the definition of D_α , that is, for every nonempty S , we have $|\mathbb{E}[D_\alpha^S]| = 2^n |\widehat{D}_\alpha(S)| = \alpha^k \binom{n}{2k}^{-1/2} \mathbb{1}(|S| = 2k)$. The closeness to $(2k)$ -wise uniform follows directly from Lemma 2. \triangleleft

Observe that the family $\{D_\alpha : \alpha \geq 0\}$ is closed under adding noise, as shown in the following claim.

▷ Claim 22. $D_\alpha \cdot N_\rho = D_{\alpha \cdot \rho^2}$ for every $\rho \in [0, 1]$.

Proof. Observe that N_ρ dampens each size- $(2k)$ (Fourier) coefficient of D_α by a factor of ρ^{2k} . To see this, note that $N_\rho(x_i) = \frac{1}{2}(1 + \rho x_i)$, and thus

$$N_\rho(x) = 2^{-n} \left(1 + \sum_S \rho^{|S|} x^S \right).$$

By Plancherel's theorem, each Fourier coefficient of the convolution $D_\alpha \cdot N_\rho$ is the product of the coefficient of D_α and N_ρ . So we have

$$(D_\alpha \cdot N_\rho)(x) = 2^{-n} \left(1 + \rho^{2k} \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S \right) = D_{\alpha \cdot \rho^2}(x). \quad \triangleleft$$

2.1 Distinguishing D_α from uniform with a threshold

We now show that a specific threshold distinguishes D_α from the uniform distribution. First, we establish the following claim showing that D_α always puts more mass than U on unbalanced strings.

▷ Claim 23. $\Pr[1^\top D_\alpha = t] \geq \Pr[B = t] \cdot (1 + (\frac{\alpha t^2}{4kn})^k)$ for every $t \geq 2\sqrt{kn}$ and $\rho \in [0, 1]$.

Proof. By our lower bound on Krawtchouk polynomials (Claim 15), we have

$$\begin{aligned} \Pr[1^\top D_\alpha = t] &= \Pr[B = t] \left(1 + \alpha^k \binom{n}{2k}^{-1/2} \overline{K}(2k, t) \right) \\ &\geq \Pr[B = t] \left(1 + \alpha^k \binom{n}{2k}^{1/2} \left(\frac{t}{2n} \right)^{2k} \right) \\ &\geq \Pr[B = t] \left(1 + \left(\frac{\alpha t^2}{4kn} \right)^k \right). \end{aligned} \quad \triangleleft$$

18:10 Pseudorandomness, Symmetry, Smoothing: I

Theorem 9 then easily follows from Claim 23 by summing over all the points at the tail, and then setting α to be $\rho^2/(5e)$.

Proof of Theorem 9. From Claim 23, it follows that

$$\begin{aligned} \Pr\left[1^\top D_\alpha \geq 2\sqrt{kn}\right] &\geq \sum_{t \geq 2\sqrt{kn}} \Pr[B = t] \cdot \left(1 + \left(\frac{\alpha t^2}{4kn}\right)^k\right) \\ &\geq \Pr[B \geq 2\sqrt{kn}] \cdot \left(1 + \left(\frac{\alpha(2\sqrt{kn})^2}{4kn}\right)^k\right) \\ &\geq \Pr[B \geq 2\sqrt{kn}] + 2^{-ck} \cdot \alpha^k, \end{aligned}$$

where the last inequality is because by tail bounds for the binomial distribution (cf. [1]) we have $\Pr[B \geq 2\sqrt{kn}] \geq 2^{-ck}$. The theorem then follows from setting α to $\rho^2/(5e)$, and noting that $D_{1/(5e)} \cdot N_\rho = D_{\rho^2/(5e)}$ by Claim 22. \blacktriangleleft

2.2 Distinguishing D_α from bounded uniformity with a symmetric test

In this section, we prove Theorem 10. We start with a claim showing that it suffices to consider bounded symmetric functions instead of Boolean symmetric test.

\triangleright **Claim 24.** Let D_1, D_2 be any distributions on $\{-1, 1\}^n$. Suppose there is a symmetric function $f: \{-1, 1\}^n \rightarrow [-1, 1]$ such that $\mathbb{E}[f(D_1)] \geq \mathbb{E}[f(D_2)] + \varepsilon$. Then there exists a symmetric Boolean function $f': \{-1, 1\}^n \rightarrow \{-1, 1\}$ such that $\mathbb{E}[f'(D_1)] \geq \mathbb{E}[f'(D_2)] + \varepsilon$.

Proof. Define $g: \{-n, \dots, n\} \rightarrow [-1, 1]$ so that $g(1^\top x) := f(x)$. Considering the randomized function $\mathbf{g}: \{-n, \dots, n\} \rightarrow \{-1, 1\}$ defined by

$$\mathbf{g}(w) := \begin{cases} 1 & \text{with probability } \frac{1+g(w)}{2} \\ -1 & \text{with probability } \frac{1-g(w)}{2}. \end{cases}$$

As f is symmetric, we have

$$\mathbb{E}\left[\mathbb{E}[\mathbf{g}(1^\top D_1)]\right] = \mathbb{E}[f(D_1)] \geq \mathbb{E}[f(D_2)] + \varepsilon = \mathbb{E}\left[\mathbb{E}[\mathbf{g}(1^\top D_2)]\right] + \varepsilon,$$

and so by averaging, there must be a choice \mathbf{g}' of \mathbf{g} such that $\mathbb{E}[\mathbf{g}'(1^\top D_1)] \geq \mathbb{E}[\mathbf{g}'(1^\top D_2)] + \varepsilon$. Defining $f': \{-1, 1\}^n \rightarrow \{-1, 1\}$ by $f'(x) := \mathbf{g}'(1^\top x)$ proves the claim. \blacktriangleleft

We now define our symmetric test. For a sufficiently small constant α , let $\beta := \frac{100}{\log(1/\alpha)}$. Define the homogeneous degree- k polynomial $p_\beta: \{-1, 1\}^n \rightarrow \mathbb{R}$ by

$$p_\beta(x) := \beta^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S = 2^n D_\beta(x) - 1.$$

Let f_β be its truncation so that it is bounded by 1, that is, we define $f_\beta: \{-1, 1\}^n \rightarrow [-1, 1]$ by $f_\beta(x) := \min\{1, p_\beta(x)\}$. As α is sufficiently small, so is β . Thus, by Claim 20, we have $f_\beta(x) \geq -1$ and so $f_\beta(x) \in [-1, 1]$ for every $x \in \{-1, 1\}^n$.

\triangleright **Claim 25.** $\mathbb{E}[f_\beta(D_\alpha)] - \mathbb{E}[f_\beta(D)] \geq (\alpha\beta)^k/2$ for any k -wise uniform distribution D .

Proof. As p_β has degree- $(2k)$, for any $(2k)$ -wise uniform distribution D , we have $\mathbb{E}[f_\beta(D)] \leq \mathbb{E}[p_\beta(D)] = \mathbb{E}[p_\beta(U)] = 0$. Note that we can write $f_\beta(x)$ as $p_\beta(x) - (p_\beta(x) - 1)\mathbb{1}(p_\beta(x) > 1)$, and so

$$\mathbb{E}[f_\beta(D_\alpha)] = \mathbb{E}[p_\beta(D_\alpha)] - \mathbb{E}[(p_\beta(D_\alpha) - 1)\mathbb{1}(p_\beta(D_\alpha) > 1)]. \quad (1)$$

To bound $\mathbb{E}[f_\beta(D_\alpha)]$ from below, we will compute $\mathbb{E}[p_\beta(D_\alpha)]$ and then bound $\mathbb{E}[(p_\beta(D_\alpha) - 1)\mathbb{1}(p_\beta(D_\alpha) > 1)]$ from above.

Observe that

$$\begin{aligned} \mathbb{E}\left[\sum_{|S|=2k} U^S\right] &= \sum_{|S|=2k} \mathbb{E}[U^S] = 0 \\ \mathbb{E}\left[\left(\sum_{|S|=2k} U^S\right)^2\right] &= \sum_{\substack{|S|=2k \\ |T|=2k}} \mathbb{E}[U^{S\Delta T}] = \binom{n}{2k} \\ \mathbb{E}\left[\left(\sum_{|S|=2k} U^S\right)^3\right] &= \sum_{\substack{|S|=2k \\ |T|=2k \\ |R|=2k}} \mathbb{E}[U^{S\Delta T\Delta R}] = \binom{n}{2k} \binom{2k}{k} \binom{n-2k}{k}, \end{aligned}$$

where the last equality is because the number of subsets $S, T, R \subseteq [n]$ of size $2k$ that satisfy $S \Delta T = R$ is $\binom{n}{2k} \binom{2k}{k} \binom{n-2k}{k}$.

We have

$$\begin{aligned} \mathbb{E}[p_\beta(D_\alpha)] &= \sum_{x \in \{-1,1\}^n} D_\alpha(x) \mathbb{E}[p_\beta(x)] \\ &= \sum_{x \in \{-1,1\}^n} 2^{-n} \left(1 + \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S\right) \left(\beta^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S\right) \\ &= (\alpha\beta)^k \binom{n}{2k}^{-1} \mathbb{E}\left[\left(\sum_{|S|=2k} U^S\right)^2\right] \\ &= (\alpha\beta)^k \binom{n}{2k}^{-1} \binom{n}{2k} = (\alpha\beta)^k. \end{aligned} \quad (2)$$

We also have

$$\begin{aligned} \mathbb{E}[p_\beta(D_\alpha)^2] &= 2^{-n} \sum_{x \in \{-1,1\}^n} \left(1 + \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S\right) \left(\beta^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S\right)^2 \\ &= \beta^{2k} \binom{n}{2k}^{-1} \mathbb{E}\left[\left(\sum_{|S|=2k} U^S\right)^2\right] + (\alpha\beta^2)^k \binom{n}{2k}^{-\frac{3}{2}} \mathbb{E}\left[\left(\sum_{|S|=2k} U^S\right)^3\right] \\ &= \beta^{2k} + (\alpha\beta^2)^k \binom{n}{2k}^{-\frac{1}{2}} \binom{2k}{k} \binom{n-2k}{k} \\ &\leq \beta^{2k} + (\alpha\beta^2)^k \binom{2k}{k}^{\frac{3}{2}} \\ &\leq \beta^{2k} + (8\alpha\beta^2)^k, \\ &\leq 1, \end{aligned} \quad (3)$$

where the first inequality is because $\binom{n-2k}{k} \leq \binom{n}{k}^{\frac{1}{2}} \binom{n-k}{k}^{\frac{1}{2}} = \binom{n}{2k}^{\frac{1}{2}} \binom{2k}{k}^{\frac{1}{2}}$, using the identity $\binom{n}{m} \binom{n-m}{k-m} = \binom{n}{k} \binom{k}{m}$. The last inequality is for small enough α .

18:12 Pseudorandomness, Symmetry, Smoothing: I

Next we bound above $\mathbb{E}[(p_\beta(D_\alpha) - 1)\mathbb{1}(p_\beta(D_\alpha) > 1)]$. We in fact bound the greater quantity $\mathbb{E}[p_\beta(D_\alpha)\mathbb{1}(p_\beta(D_\alpha) > 1)]$. Using Cauchy–Schwarz and (3), the latter is at most

$$\mathbb{E}[p_\beta(D_\alpha)^2]^{\frac{1}{2}} \Pr[p_\beta(D_\alpha) > 1]^{\frac{1}{2}} \leq 1 \cdot \Pr[p_\beta(D_\alpha) > 1]^{\frac{1}{2}}. \quad (4)$$

We will show that

$$\Pr[p_\beta(D_\alpha) > 1]^{\frac{1}{2}} \leq e^{-(\frac{1}{e\beta} - 1)\frac{k}{4}}. \quad (5)$$

So, using $\beta = \frac{100}{\log(1/\alpha)}$, (4) is less than $(\alpha\beta)^k/2$. Plugging these bounds into (1), we conclude that

$$\mathbb{E}[f_\beta(D_\alpha)] - \mathbb{E}[f_\beta(D)] \geq (\alpha\beta)^k/2$$

for any $(2k)$ -wise uniform D , proving the claim assuming (5) holds.

It remains to prove (5). Suppose $|p_\beta(x)| > 1$. Then by its definition and Corollary 16, it must be the case that

$$1 \leq \beta^k \binom{n}{2k}^{-\frac{1}{2}} \left| \sum_{|S|=2k} x^S \right| \leq \beta^k \binom{n}{2k}^{\frac{1}{2}} \left(\frac{2k}{n} + \frac{(1^\top x)^2}{n^2} \right)^k \leq (e\beta)^k \left(1 + \frac{(1^\top x)^2}{2kn} \right)^k,$$

which implies $x \in E_\beta := \{x \in \{-1, 1\}^n : (1^\top x)^2 \geq (\frac{1}{e\beta} - 1)2kn\}$. Jumping ahead, we will use below that by Hoeffding’s inequality, we have $\Pr[U \in E_\beta] \leq e^{-k(\frac{1}{e\beta} - 1)}$. In the meanwhile, we use the implication just noted to write

$$\begin{aligned} \Pr[p_\beta(D_\alpha) > 1] &\leq \Pr[D_\alpha \in E_\beta] = 2^{-n} \sum_{x \in E_\beta} \left(1 + \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \sum_{|S|=2k} x^S \right) \\ &= \Pr[U \in E_\beta] + \alpha^k \binom{n}{2k}^{-\frac{1}{2}} 2^{-n} \sum_{x \in E_\beta} \sum_{|S|=2k} x^S. \end{aligned} \quad (6)$$

We rewrite and bound the second term using Cauchy–Schwarz as follows:

$$\begin{aligned} \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \mathbb{E} \left[\sum_{|S|=2k} U^S \cdot \mathbb{1}(U \in E_\beta) \right] &\leq \alpha^k \binom{n}{2k}^{-\frac{1}{2}} \mathbb{E} \left[\left(\sum_{|S|=2k} U^S \right)^2 \right]^{\frac{1}{2}} \cdot \Pr[U \in E_\beta]^{\frac{1}{2}} \\ &= \alpha^k \cdot \Pr[U \in E_\beta]^{\frac{1}{2}}. \end{aligned}$$

Therefore

$$(6) \leq \Pr[U \in E_\beta]^{\frac{1}{2}} \left(\Pr[U \in E_\beta]^{\frac{1}{2}} + \alpha^k \right) \leq e^{-(\frac{1}{e\beta} - 1)\frac{k}{2}} \cdot (1/2 + 1/2).$$

This proves (5). ◁

Proof of Theorem 10. For any $\rho \in (0, 1]$, by Claim 22 we have $D_c \cdot N_\rho = D_{c\rho^2}$. So we can take α to be $c\rho^2$, and thus $\beta = c/\log(1/2\rho)$. By Claim 25, the distinguishing advantage is at least $(c\rho^2/\log(1/2\rho))^k$. ◀

3 Bounded uniformity plus noise fools symmetric tests

Here we prove Theorem 7. The starting observation for the proof of this theorem (and also of Theorems 13 and 14) is that the Fourier expansion of any symmetric function is a linear combination of the Krawtchouk polynomials $\overline{K}(\ell, 1^\top x) := \sum_{|S|=\ell} x^S$ weighted by the coefficients $\widehat{f}([\ell])$. As k -wise uniformity fools all parities of size at most k , it suffices to consider the $\ell > k$ terms. While $\overline{K}(\ell, 1^\top x)$ can be as large as $\binom{n}{\ell}$ on the all-1 string, it follows from Cauchy–Schwarz that its average $\mathbb{E}_x[|\overline{K}(\ell, 1^\top x)|]$ is at most $\binom{n}{\ell}^{1/2}$. Moreover, a simple argument (Fact 26) shows that $|\widehat{f}([\ell])|$ is bounded by $\binom{n}{\ell}^{-1/2}$, the reciprocal of the upper bound on $\mathbb{E}_x[|\overline{K}(\ell, 1^\top x)|]$, and so their product is at most 1, which is then dampened to $\rho^\ell \leq \rho^k$ by noise.

To make this argument go through, we use Corollary 16 to show that $|\overline{K}(\ell, 1^\top x)|$ is close to $\binom{n}{\ell}^{1/2}$ when x is nearly-balanced, which holds with high probability under k -wise uniformity (Corollary 28).

We start by proving a few useful facts about symmetric functions and distributions.

► **Fact 26.** *Let $f: \{-1, 1\}^n \rightarrow [-1, 1]$ be any symmetric function. For every $S \subseteq [n]$ of size ℓ , we have (1) $\widehat{f}(S) = \widehat{f}([\ell])$ and (2) $|\widehat{f}([\ell])| \leq \binom{n}{\ell}^{-1/2}$.*

Proof. (1) is clear. To see (2), by Cauchy–Schwarz and Parseval’s identity, we have

$$\binom{n}{\ell} |\widehat{f}([\ell])| = \left| \sum_{|S|=\ell} \widehat{f}(S) \right| \leq \binom{n}{\ell}^{1/2} \left(\sum_{|S|=\ell} \widehat{f}(S)^2 \right)^{1/2} \leq \binom{n}{\ell}^{1/2} \mathbb{E}[f(U)^2] \leq \binom{n}{\ell}^{1/2}. \blacktriangleleft$$

We also need the following well-known moment bounds for k -wise uniform distributions. For a short proof see [14, Lemma 32].

► **Lemma 27.** *Let D be a $(2k)$ -wise uniform distribution on $\{-1, 1\}^n$. Then $\mathbb{E}[(\sum_{i=1}^n D_i)^{2k}] \leq \sqrt{2} (2kn/e)^k$.*

By Markov’s inequality, this implies the following tail bound.

► **Corollary 28.** *Let D be a $(2k)$ -wise uniform distribution on $\{-1, 1\}^n$. For every integer $t > 0$, we have*

$$\Pr[|1^\top D| \geq t] \leq \sqrt{2} \left(\frac{2kn}{et^2} \right)^k.$$

The following fact says that a distribution remains close to itself after conditioning on any high probability event.

► **Fact 29.** *Let D be any distribution on $\{-1, 1\}^n$ and E be any event. Then the conditional distribution $D \mid E$ is $(1 - \Pr[E])$ -close to D .*

Proof. Let \overline{E} be the complement of E . For every Boolean test $g: \{-1, 1\}^n \rightarrow \{0, 1\}$ we have

$$\begin{aligned} \mathbb{E}[g(D)] &= \mathbb{E}[g(D \mid E)](1 - \Pr[\overline{E}]) + \mathbb{E}[g(D \mid \overline{E})] \Pr[\overline{E}] \\ &= \mathbb{E}[g(D \mid E)] + (\mathbb{E}[g(D \mid \overline{E})] - \mathbb{E}[g(D \mid E)]) \Pr[\overline{E}]. \end{aligned}$$

So $|\mathbb{E}[g(D)] - \mathbb{E}[g(D \mid E)]| \leq \Pr[\overline{E}]$, as $|\mathbb{E}[g(D \mid \overline{E})] - \mathbb{E}[g(D \mid E)]|$ is bounded by 1. \blacktriangleleft

18:14 Pseudorandomness, Symmetry, Smoothing: I

Proof of Theorem 7. Define $G := \{x \in \{-1, 1\}^n : |\sum_{i=1}^n x_i| \leq \sqrt{nk/3\rho}\}$. We write $f := f_{\leq k} + f_{>k}$, where $f_{\leq k}(x) = \sum_{|S| \leq k} \widehat{f}(S)x^S$, and $f_{>k}(x) := f(x) - f_{\leq k}(x) = \sum_{|S| > k} \widehat{f}(S)x^S$. For convenience let $Z := D \cdot N_\rho$. As Z is $(2k)$ -wise uniform, we have

$$\mathbb{E}[f] = \mathbb{E}[f_{\leq k}(Z)] = \mathbb{E}[f_{\leq k}(Z)\mathbb{1}(D \in G)] + \mathbb{E}[f_{\leq k}(Z)\mathbb{1}(D \notin G)].$$

So we can bound the error by

$$\begin{aligned} |\mathbb{E}[f(Z)] - \mathbb{E}[f]| &= |\mathbb{E}[f(Z)\mathbb{1}(D \in G)] + \mathbb{E}[f(Z)\mathbb{1}(D \notin G)] - \mathbb{E}[f]| \\ &\leq |\mathbb{E}[f_{\leq k}(Z)\mathbb{1}(D \in G)] - \mathbb{E}[f]| + |\mathbb{E}[f_{>k}(Z)\mathbb{1}(D \in G)]| + \Pr[D \notin G] \\ &\leq |\mathbb{E}[f_{\leq k}(Z)\mathbb{1}(D \notin G)]| + |\mathbb{E}[f_{>k}(Z)\mathbb{1}(D \in G)]| + \Pr[D \notin G], \end{aligned} \quad (7)$$

We now bound each term individually. By Corollary 28, we have

$$\Pr[D \notin G] \leq \sqrt{2} \cdot \left(\frac{2 \cdot 3\rho}{e}\right)^k \leq \sqrt{2} \cdot (e\rho)^k. \quad (8)$$

We now bound the first term, As $f_{\leq k}$ has degree $2k$, by Parseval's identity and $(2k)$ -wise uniformity of Z , we have

$$\mathbb{E}[f_{\leq k}(Z)^2] = \mathbb{E}[f_{\leq k}(U)^2] = \mathbb{E}[f(U)^2] \leq 1.$$

By Cauchy–Schwarz, the first term in (7) is at most

$$|\mathbb{E}[f_{\leq k}(Z)\mathbb{1}(D \notin G)]| \leq \mathbb{E}[f_{\leq k}(Z)^2]^{1/2} \Pr[D \notin G]^{1/2} \leq 2^{1/4} \cdot (e\rho)^{k/2}. \quad (9)$$

It remains to bound the second term in (7). For every $x \in G$, we will show that

$$|\mathbb{E}[f_{>k}(x \cdot N_\rho)]| \leq 7 \cdot (e\rho)^{k/2}. \quad (10)$$

Plugging (8)–(10) into (7) gives an error bound of at most $10(e\rho)^{k/2}$, as desired.

We now show (10). As $\mathbb{E}[N_\rho^S] = \rho^{|S|}$, we have

$$\left| \mathbb{E}[f_{>k}(x \cdot N_\rho)] \right| = \left| \sum_{|S| > k} \rho^{|S|} \widehat{f}(S)x^S \right| = \left| \sum_{\ell=k+1}^n \rho^\ell \sum_{|S|=\ell} \widehat{f}(S)x^S \right|.$$

Applying Fact 26 and Corollary 16, and using the inequality $\binom{n}{\ell} \leq (en/\ell)^\ell$, we have

$$\begin{aligned} \left| \sum_{\ell=k+1}^n \rho^\ell \sum_{|S|=\ell} \widehat{f}(S)x^S \right| &\leq \left| \sum_{\ell=k+1}^n \rho^\ell \cdot \widehat{f}([\ell]) \sum_{|S|=\ell} x^S \right| \\ &\leq \sum_{\ell=k+1}^n \rho^\ell \cdot |\widehat{f}([\ell])| \cdot \left| \sum_{|S|=\ell} x^S \right| \\ &\leq \sum_{\ell=k+1}^n \rho^\ell \cdot \binom{n}{\ell}^{1/2} \left(\frac{\ell}{n} + \frac{k}{3\rho n} \right)^{\ell/2} \\ &\leq \sum_{\ell=k+1}^n \rho^\ell \cdot e^{\ell/2} \left(1 + \frac{k}{3\rho\ell} \right)^{\ell/2} \\ &\leq \sum_{\ell=k+1}^n \left(\rho \left(\rho e + \frac{e}{3} \right) \right)^{\ell/2} \\ &\leq 7 \cdot (e\rho)^{k/2} \end{aligned}$$

where the last inequality is because we can assume $\rho \leq 1/e$, as otherwise the conclusion is trivial, and so we have $\rho e + e/3 \leq 1 + e/3 \leq 2$. This shows (10). \blacktriangleleft

4 Shifted symmetric small-bias fools symmetric tests

In this section we prove Theorem 13. The proof follows a similar high-level idea to the proof of Theorem 7, but we trade symmetry for noise, because xor-ing the uniform permutation of a string has a similar effect to adding noise (see Claim 31).

As mentioned in the introduction, we actually prove stronger results about fooling symmetric functions. One can then obtain Theorem 13 by combining Corollary 33 below and the following claim.

▷ **Claim 30.** Let D be a symmetric distribution on $\{0, 1\}^n$. If $|D|$ is ε -close to the binomial distribution $\text{Bin}(n, 1/2)$, then D is ε -close to the uniform distribution.

Proof. We have

$$\sum_{w=0}^n \sum_{|x|=w} \left| 2^{-n} - \frac{D(w)}{\binom{n}{w}} \right| = \sum_{w=0}^n \binom{n}{w} \left| 2^{-n} - \frac{D(w)}{\binom{n}{w}} \right| = \sum_{w=0}^n \left| 2^{-n} \binom{n}{w} - D(w) \right| \leq \varepsilon. \quad \triangleleft$$

In turn, Corollary 33 follows from Theorem 32, showing that any symmetric small-bias distribution xor-ed a nearly-balanced string fools symmetric functions. Then we prove Theorem 34, which is a generalization of Theorem 32 that covers a more general settings of parameters. We complement Theorem 34 with a lower bound (Claim 35).

First we show that the bias of the uniform permutation of a string on parity tests is equal to the normalized Krawtchouk polynomials.

▷ **Claim 31.** Let W_t be the uniform distribution on $\{x \in \{-1, 1\}^n : \sum_{i=1}^n x_i = t\}$. For every subset $S \subseteq [n]$ of size ℓ , we have

$$|\mathbb{E}[W_t^{[n] \setminus S}]| = |\mathbb{E}[W_t^S]| = \frac{|\overline{K}(\ell, t)|}{\binom{n}{\ell}}.$$

Proof. The first equality follows from $|\sum_{|S|=n-\ell} z^S| = |z^{[n]} \sum_{|S|=\ell} z^S| = |\sum_{|S|=\ell} z^S|$. To prove the second inequality, first fix a string z with $\sum_{i=1}^n z_i = t$. Observe that by symmetry we have $\sum_{x: \sum_{i=1}^n x_i = t} x^S = \sum_{x: \sum_{i=1}^n x_i = t} x^{[S]}$ for any $S \subseteq [n]$ of size ℓ , and $\sum_{|S|=\ell} x^S = \sum_{|S|=\ell} z^S$ for any $x \in \{-1, 1\}^n$ with $\sum_{i=1}^n x_i = t$. Hence,

$$\binom{n}{\ell} \sum_{x: \sum_i x_i = t} x^{[S]} = \sum_{|S|=\ell} \sum_{x: \sum_i x_i = t} x^S = \sum_{x: \sum_i x_i = t} \sum_{|S|=\ell} x^S = \binom{n}{\ell} \sum_{|S|=\ell} z^S.$$

Rearranging gives

$$|\mathbb{E}[W_t^S]| = \frac{1}{\binom{n}{\ell}} \left| \sum_{x: \sum_i x_i = t} x^{[S]} \right| = \frac{1}{\binom{n}{\ell}} \left| \sum_{|S|=\ell} z^S \right| = \frac{|\overline{K}(\ell, t)|}{\binom{n}{\ell}}. \quad \triangleleft$$

4.1 Proof of Corollary 33

Corollary 33 is a straightforward corollary of Theorem 32, which we now prove.

▶ **Theorem 32.** Let D_{sym} be a symmetric n^{-20k} -biased distribution on $\{-1, 1\}^n$ and $z \in \{-1, 1\}^n$ be any string with $|\sum_{i=1}^n z_i| \leq n^{0.6}$. Then $|\mathbb{E}[f(z \cdot D_{\text{sym}})] - \mathbb{E}[f]| \leq O_k(n^{-0.3k})$.

Note that it is crucial that D_{sym} is symmetric, as small-bias distributions are closed under shifts; so every small-bias distribution D is also a shifted small-bias distribution.

18:16 Pseudorandomness, Symmetry, Smoothing: I

► **Corollary 33.** Let D_{sym} and D be two independent n^{-20k} -biased distributions on $\{-1, 1\}^n$, where D_{sym} is symmetric. Then $|\mathbb{E}[f(D_{\text{sym}} + D)] - \mathbb{E}[f]| \leq c_k(n^{-0.3k})$ for every symmetric function $f: \{-1, 1\}^n \rightarrow [-1, 1]$.

Also note that $D + D_{\text{sym}}$ itself is not necessarily a symmetric distribution.

Proof of Corollary 33. By Lemma 2, D is n^{-10k} -close to $(10k)$ -wise uniform. So by Corollary 28,

$$\Pr\left[\left|\sum_{i=1}^n D_i\right| \geq n^{0.6}\right] \leq \left(\frac{10kn}{n^{1.2}}\right)^{5k} + n^{-10k} \leq O_k(n^{-k}).$$

It follows from Theorem 32 that the error is $O_k(n^{-k}) + O_k(n^{-0.3k}) = O_k(n^{-0.3k})$. ◀

Proof of Theorem 32. Let $\varepsilon := n^{-20k}$ be the bias of D_{sym} and $t := n^{0.6}$. Define $G := \{x \in \{-1, 1\}^n : |\sum_{i=1}^n x_i| \leq t\}$. As D_{sym} is ε -biased, by Lemma 2, it is δ -close to $(30k)$ -wise uniform, where $\delta \leq n^{-4k}$. Applying Corollary 28 with our choice of $t = n^{0.6}$ in the definition of G , we have

$$\Pr[D_{\text{sym}} \notin G] \leq \left(\frac{30kn}{n^{1.2}}\right)^{15k} + \delta \leq O_k(n^{-3k}). \quad (11)$$

Let D'_{sym} be the distribution of D_{sym} conditioned on $D_{\text{sym}} \in G$. Note that D'_{sym} remains a symmetric distribution and by Fact 29 is $\Pr[D_{\text{sym}} \notin G]$ -close to D_{sym} , and thus is ε' -biased, where $\varepsilon' := \varepsilon + \Pr[D_{\text{sym}} \notin G] \leq O_k(n^{-3k})$. We now write $f := f_{\text{mid}} + f_{\text{ends}}$, where $f_{\text{mid}}(x) := \sum_{k < |S| < n-k} \widehat{f}(S)x^S$, and $f_{\text{ends}}(x) := f(x) - f_{\text{mid}}(x) = \sum_{|S| \in [0, k] \cup [n-k, n]} \widehat{f}(S)x^S$. By the triangle inequality, we have

$$\begin{aligned} |\mathbb{E}[f(z \cdot D_{\text{sym}})] - \mathbb{E}[f]| &\leq |\mathbb{E}[f(z \cdot D'_{\text{sym}})] - \mathbb{E}[f]| + \Pr[D_{\text{sym}} \notin G] \\ &\leq |\mathbb{E}[f_{\text{ends}}(z \cdot D'_{\text{sym}})] - \mathbb{E}[f]| + |\mathbb{E}[f_{\text{mid}}(z \cdot D'_{\text{sym}})]| + O_k(n^{-3k}). \end{aligned} \quad (12)$$

We now bound each of the two terms on the right hand side. As $z \cdot D'_{\text{sym}}$ is ε' -biased, we have

$$|\mathbb{E}[f_{\text{ends}}(z \cdot D'_{\text{sym}})] - \mathbb{E}[f]| \leq \sum_{|S| \in [1, k] \cup [n-k, n]} |\widehat{f}(S)| \varepsilon' \leq 2n^k \varepsilon' \leq O_k(n^{-2k}). \quad (13)$$

To bound $|\mathbb{E}[f_{\text{mid}}(z \cdot D'_{\text{sym}})]|$, let S be any subset of size ℓ . As f is symmetric, we have $\widehat{f}(S) = \widehat{f}([\ell])$. As D'_{sym} is also symmetric, we have $\mathbb{E}[D'_{\text{sym}}{}^S] = \varepsilon_\ell$ for some ε_ℓ which only depends on the size of S . Hence,

$$\left| \mathbb{E}[f_{\text{mid}}(z \cdot D'_{\text{sym}})] \right| \leq \left| \sum_{\ell=k+1}^{n-k-1} \sum_{|S|=\ell} \widehat{f}(S) \mathbb{E}[D'_{\text{sym}}{}^S] z^S \right| = \left| \sum_{\ell=k+1}^{n-k-1} \widehat{f}([\ell]) \varepsilon_\ell \sum_{|S|=\ell} z^S \right|.$$

As $|\sum_{|S|=\ell} z^S| = |z^{[n]} \sum_{|S|=n-\ell} z^S| = |\sum_{|S|=n-\ell} z^S|$, by Fact 26 and Claim 31 we have

$$\left| \sum_{\ell=k+1}^{n-k-1} \widehat{f}([\ell]) \varepsilon_\ell \sum_{|S|=\ell} z^S \right| \leq \sum_{\ell=k+1}^{n-k-1} \left(|\widehat{f}([\ell])| \cdot |\varepsilon_\ell| \cdot \left| \sum_{|S|=\ell} z^S \right| \right) \leq 2 \sum_{\ell=k+1}^{\lfloor n/2 \rfloor} \frac{\overline{K}(\ell, n^{0.6})^2}{\binom{n}{\ell}^{3/2}}.$$

We first bound the partial sum over ℓ from $n/4$ to $n/2$. Note that the binary entropy function $H(x) := x \log_2(1/x) + (1-x) \log_2(1/(1-x))$ is increasing on $[0, 1/2]$. In particular, we have

$H(1/4) \geq 4/5$ and so $\frac{3}{2}H(1/4) \geq 6/5$. By Stirling's approximation, we have $\binom{n}{\ell} \geq \frac{1}{n^2} 2^{nH(\frac{\ell}{n})}$ (see [24] for a proof). Applying Corollary 18 with these facts, we have

$$\sum_{\ell=\max\{n/4, k\}+1}^{\lfloor n/2 \rfloor} \frac{\overline{K}(\ell, n^{0.6})^2}{\binom{n}{\ell}^{3/2}} \leq \frac{n}{4} \cdot n^2 \cdot 2^{-n(\frac{3}{2}H(\frac{1}{4})-1-n^{-0.8})} \leq 2^{-n/10}. \quad (14)$$

We now bound the remaining sum (i.e., the partial sum from $\ell = k + 1$ to $n/4$). Using Corollary 16 and the inequality $\binom{n}{\ell} \leq (\frac{en}{\ell})^\ell$, we have

$$\sum_{\ell=k+1}^{n/4} \frac{\overline{K}(\ell, n^{0.6})^2}{\binom{n}{\ell}^{3/2}} \leq \sum_{\ell=k+1}^{n/4} \binom{n}{\ell}^{1/2} \left(\frac{\ell}{n} + \frac{n^{1.2}}{n^2}\right)^\ell \leq \sum_{\ell=k+1}^{n/4} \left(\frac{en}{\ell}\right)^{\ell/2} \left(\frac{\ell}{n} + \frac{1}{n^{0.8}}\right)^\ell.$$

Observe that each term in the sum is at most $1/2$ of its previous term, and so this sum is bounded by twice the first term, which is at most $O_k(n^{-0.3k})$. Therefore,

$$|\mathbb{E}[f_{\text{mid}}(z \cdot D'_{\text{sym}})]| \leq 2^{-n/10} + O_k(n^{-0.3k}) \leq O_k(n^{-0.3k}). \quad (15)$$

Plugging (13) and (15) in (12) completes the proof. \blacktriangleleft

4.2 General case

Theorem 32 is stated for a nearly-balanced shift. We now prove a general bound that holds for any shifts.

► Theorem 34. *There exists a constant C such that the following holds. Let D_{sym} be a symmetric ε -biased distribution on $\{-1, 1\}^n$ and $z \in \{-1, 1\}^n$ be any string. Let $s := |\sum_{i=1}^n z_i|$. For every positive integer k and every symmetric function $f: \{-1, 1\}^n \rightarrow [-1, 1]$, we have*

$$\left| \mathbb{E}[f(z \cdot D_{\text{sym}})] - \mathbb{E}[f] \right| \leq C \left(\left(\frac{11 \max\{s, \sqrt{kn}\}}{n} \right)^{k/2} + \left(\frac{e^3 n}{2k} \right)^{k/2} \varepsilon \right).$$

The following lower bound shows that the dependence on s in Theorem 34 is necessary.

▷ Claim 35. There exists a constant $c > 0$ such that the following holds. For every integer $m \geq 3$, there is a symmetric e^{-cn/m^2} -biased distribution D on $\{0, 1\}^n$ such that for every string $z \in \{0, 1\}^n$ of Hamming weight at most $\lfloor m/2 \rfloor - 1$, there exists a symmetric function $f: \{0, 1\}^n \rightarrow \{0, 1\}$ such that $f(D) = 0$ always and $\Pr[f(U) = 1] \geq 1/m - e^{-cn/m^2}$.

Proof of Claim 35. Let D be the uniform distribution on $\{x \in \{0, 1\}^n : \sum_i x_i \equiv 0 \pmod{m}\}$. It is known that D is $2^{-cn/m^2}$ -biased (see Claim 18 and Lemma 19 in [11] for a proof). Let z be any string of weight $|z| \leq \lfloor m/2 \rfloor - 1$. Consider the symmetric function $f(x) := \mathbb{1}(|x| \equiv \lfloor (m+1)/2 \rfloor \pmod{m})$. By the triangle inequality, we have $|D| - |z| \leq |D + z| \leq |D| + |z|$, and so $|D + z| \not\equiv \lfloor (m+1)/2 \rfloor \pmod{m}$.

On the other hand, it is known that $\Pr[\text{Bin}(n, 1/2) \equiv \lfloor (m+1)/2 \rfloor] \geq 1/m - e^{-cn/m^2}$ (see, again, Claim 18 in [11] for a proof). \triangleleft

Proof of Theorem 34. We may assume $k \leq n/16$ and $s \leq n/120$, as otherwise the bound given in the theorem is at least 1. Define $G := \{x \in \{-1, 1\}^n : |\sum_{i=1}^n x_i| \leq t\}$, where $t := t(n, k, s)$ is a parameter to be chosen. As D_{sym} is ε -biased, by Lemma 2, it is δ -close to $(2k)$ -wise uniform, where $\delta := (\frac{e^3 n}{2k})^k \varepsilon$. Applying Corollary 28, we have

$$\Pr[D_{\text{sym}} \notin G] \leq \sqrt{2} \cdot \left(\frac{2nk}{et^2} \right)^k + \delta. \quad (16)$$

18:18 Pseudorandomness, Symmetry, Smoothing: I

We write $f := f_{\text{mid}} + f_{\text{ends}}$, where $f_{\text{mid}}(x) := \sum_{k+1 < |S| < n-k} \widehat{f}(S)x^S$, and $f_{\text{ends}}(x) := f(x) - f_{\text{mid}}(x) = \sum_{|S| \in [0, k] \cup [n-k, n]} \widehat{f}(S)x^S$. For convenience, let $Z := z \cdot D_{\text{sym}}$. As Z is ε -biased, we have

$$|\mathbb{E}[f_{\text{ends}}(Z)] - \mathbb{E}[f]| \leq \sum_{|S| \in [1, k] \cup [n-k, n]} |\widehat{f}(S)| \varepsilon \leq 2 \left(\frac{e^3 n}{k} \right)^{k/2} \varepsilon \leq \delta.$$

By the triangle inequality, we have

$$\begin{aligned} & |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \in G)] - \mathbb{E}[f]| \\ & \leq |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \in G)] - \mathbb{E}[f_{\text{ends}}(Z)]| + |\mathbb{E}[f_{\text{ends}}(Z)] - \mathbb{E}[f]| \\ & \leq |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \in G)] - \mathbb{E}[f_{\text{ends}}(Z)]| + \delta \\ & = |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \notin G)]| + \delta. \end{aligned} \tag{17}$$

As Z is ε -biased,

$$\begin{aligned} \mathbb{E}[f_{\text{ends}}(Z)^2] &= \sum_{|S|, |T| \in [0, k] \cup [n-k, n]} \widehat{f}(S) \widehat{f}(T) \mathbb{E}[Z^{S \Delta T}] \\ &\leq \sum_{|S| \in [0, k] \cup [n-k, n]} \widehat{f}(S)^2 + \sum_{|S| \neq |T| \in [0, k] \cup [n-k, n]} |\widehat{f}(S)| |\widehat{f}(T)| \varepsilon \\ &\leq 1 + 2 \binom{n}{k} \varepsilon \leq 1 + \delta. \end{aligned}$$

By Cauchy–Schwarz,

$$|\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \notin G)]| \leq \mathbb{E}[f_{\text{ends}}(Z)^2]^{1/2} \Pr[D_{\text{sym}} \notin G]^{1/2} \leq 2 \Pr[D_{\text{sym}} \notin G]^{1/2}. \tag{18}$$

We now use (17) and (18) to bound the error as follows:

$$\begin{aligned} |\mathbb{E}[f(Z)] - \mathbb{E}[f]| &= |\mathbb{E}[f(Z) \mathbb{1}(D_{\text{sym}} \in G)] + \mathbb{E}[f(Z) \mathbb{1}(D_{\text{sym}} \notin G)] - \mathbb{E}[f]| \\ &\leq |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \in G)] - \mathbb{E}[f]| + |\mathbb{E}[f_{\text{mid}}(Z) \mathbb{1}(D_{\text{sym}} \in G)]| + \Pr[D_{\text{sym}} \notin G] \\ &\leq |\mathbb{E}[f_{\text{ends}}(Z) \mathbb{1}(D_{\text{sym}} \notin G)]| + |\mathbb{E}[f_{\text{mid}}(Z) \mathbb{1}(D_{\text{sym}} \in G)]| + \Pr[D_{\text{sym}} \notin G] + \delta \\ &\leq |\mathbb{E}[f_{\text{mid}}(Z) \mathbb{1}(D_{\text{sym}} \in G)]| + 3 \Pr[D_{\text{sym}} \notin G]^{1/2} + 2\delta. \end{aligned} \tag{19}$$

We will bound the first term in (19) by

$$|\mathbb{E}[f_{\text{mid}}(z \cdot D_{\text{sym}})]| \leq \begin{cases} O(1) \left(\frac{120k}{n} \right)^{k/4} & \text{if } s \leq \sqrt{kn} \text{ and } t = (kn^3)^{1/4} \\ O(1) \left(\frac{120s^2}{n^2} \right)^{k/4} & \text{if } s \geq \sqrt{kn} \text{ and } t = \left(\frac{k^2 n^4}{s^2} \right)^{1/4}. \end{cases} \tag{20}$$

Plugging (16) and (20) into (19) gives us an error of

$$O(1) \left(\left(\frac{120 \max\{s, \sqrt{kn}\}}{n} \right)^{k/2} + \delta \right)$$

as desired.

It remains to prove (20). Let S be any subset of size ℓ . As f is symmetric, we have $\widehat{f}(S) = \widehat{f}([S])$. Let D'_{sym} be the distribution of D_{sym} conditioned on $D_{\text{sym}} \in G$. Note that D'_{sym} is also symmetric, and so we have $\mathbb{E}[D'_{\text{sym}}] = \varepsilon_\ell$ for some ε_ℓ which only depends on the size of S . Hence,

$$\left| \mathbb{E}[f_{\text{mid}}(z \cdot D_{\text{sym}}) \mathbb{1}(D_{\text{sym}} \in G)] \right| \leq \left| \sum_{\ell=k+1}^{n-k-1} \sum_{|S|=\ell} \widehat{f}(S) \mathbb{E}[D'_{\text{sym}} z^S] \right| = \left| \sum_{\ell=k+1}^{n-k-1} \widehat{f}([\ell]) \varepsilon_{\ell} \sum_{|S|=\ell} z^S \right|.$$

As $|\sum_{|S|=\ell} z^S| = |z^{[n]} \sum_{|S|=n-\ell} z^S| = |\sum_{|S|=n-\ell} z^S|$, by Fact 26 and Claim 31 we have

$$\left| \sum_{\ell=k+1}^{n-k-1} \widehat{f}([\ell]) \varepsilon_{\ell} \sum_{|S|=\ell} z^S \right| \leq \sum_{\ell=k+1}^{n-k-1} \left(|\widehat{f}([\ell])| \cdot |\varepsilon_{\ell}| \cdot \left| \sum_{|S|=\ell} z^S \right| \right) \leq 2 \sum_{\ell=k+1}^{\lfloor n/2 \rfloor} \frac{\overline{K}(\ell, t) \overline{K}(\ell, s)}{\binom{n}{\ell}^{3/2}}.$$

We first bound the sum over ℓ from $n/4$ to $n/2$. Note that the binary entropy function $H(x) := x \log_2(1/x) + (1-x) \log_2(1/(1-x))$ is increasing on $[0, 1/2]$. In particular, we have $H(1/4) \geq 4/5$ and so $\frac{3}{2}H(1/4) \geq 6/5$. By Stirling's approximation, we have $\binom{n}{\ell} \geq \frac{1}{n^2} 2^{nH(\frac{\ell}{n})}$ (see [24] for a proof). Applying Corollary 18 with these facts along with $s \leq n/120$ and $t \leq (kn^3)^{1/4} \leq n/2$, we have

$$\sum_{\ell=\max\{n/4, k\}+1}^{\lfloor n/2 \rfloor} \frac{\overline{K}(\ell, t) \overline{K}(\ell, s)}{\binom{n}{\ell}^{3/2}} \leq \sum_{\ell=\max\{n/4, k\}+1}^{\lfloor n/2 \rfloor} \frac{1}{\binom{n}{\ell}^{3/2}} \cdot 2^{\frac{n}{2}(2H(\frac{1}{4}) + \frac{s^2}{n^2} + \frac{t^2}{n^2})} \quad (21)$$

$$\leq \frac{n}{4} \cdot n^2 \cdot 2^{-n(\frac{3}{2}H(\frac{1}{4}) - 1 - \frac{s^2+t^2}{2n^2})} \leq 2^{-n/15}. \quad (22)$$

We now bound the remaining sum (i.e. from $\ell = k+1$ to $n/4$). Using Corollary 16 and Claim 31, and the inequality $\binom{n}{\ell} \leq (\frac{en}{\ell})^{\ell}$, we have

$$\begin{aligned} \sum_{\ell=k+1}^{\lfloor n/2 \rfloor} \frac{\overline{K}(\ell, t) \overline{K}(\ell, s)}{\binom{n}{\ell}^{3/2}} &\leq \sum_{\ell=k+1}^{n/4} \binom{n}{\ell}^{1/2} \cdot \left(\frac{\ell}{n} + \frac{t^2}{n^2} \right)^{\ell/2} \left(\frac{\ell}{n} + \frac{s^2}{n^2} \right)^{\ell/2} \\ &= \sum_{\ell=k+1}^{n/4} \left(e \left(\frac{\ell}{n} + \frac{t^2}{n^2} + \frac{s^2}{n^2} + \frac{t^2 s^2}{n^3 \ell} \right) \right)^{\ell/2}. \end{aligned} \quad (23)$$

We now consider the two cases in (20).

Case 1: $s \leq \sqrt{nk}$ and $t = (n^3 k)^{1/4}$. In this case (23) is at most

$$\sum_{\ell=k+1}^{n/4} \left(e \left(\frac{\ell}{n} + \sqrt{\frac{k}{n}} + \frac{k}{n} + \sqrt{\frac{k}{n}} \right) \right)^{\ell/2} \leq \sum_{\ell=k+1}^{n/4} \left(e \left(\frac{\ell}{n} + 3\sqrt{\frac{k}{n}} \right) \right)^{\ell/2} \leq O(1) \left(\frac{120k}{n} \right)^{k/4},$$

where the last inequality follows because each term in the sum is at most 9/10 of its previous term, and so the sum is bounded by 10 times the first term. Combining this with (21) proves the first case in (20).

Case 2: $s \geq \sqrt{nk}$ and $t = (k^2 n^4 / s^2)^{1/4}$. In this case (23) is at most

$$\sum_{\ell=k+1}^{n/4} \left(e \left(\frac{\ell}{n} + \frac{k}{s} + \frac{s^2}{n^2} + \frac{s}{n} \right) \right)^{\ell/2} \leq \sum_{\ell=k+1}^{n/4} \left(e \left(\frac{\ell}{n} + \frac{3s}{n} \right) \right)^{\ell/2} \leq O(1) \left(\frac{120s^2}{n^2} \right)^{k/4},$$

where again the last inequality follows because each term in the sum is at most 9/10 of its previous term, and so the sum is bounded by 10 times the first term. Combining this with (21) proves the second case in (20). \blacktriangleleft

5 Proof of Theorem 14

In this section, we prove Theorem 14, which is based on the same idea that was used in the previous sections. The difference is that here we use that a typical shift is nearly balanced, and so $\bar{K}(\ell, 1^\top x)$ is small.

Proof of Theorem 14. Applying Cauchy–Schwarz, Parseval’s identity (to the function $g(u) := f(u \cdot D)$), and the assumption that $\mathbb{E}[D^S] = 0$ for $|S| \in [1, k] \cup [n - k, n]$, we have

$$\mathbb{E}_{\mathbf{u}} \left[\left| \mathbb{E}[f(\mathbf{u} \cdot D)] - \mathbb{E}[f] \right| \right]^2 \leq \mathbb{E}_{\mathbf{u}} \left[\left(\mathbb{E}[f(\mathbf{u} \cdot D)] - \mathbb{E}[f] \right)^2 \right] = \sum_{S: |S| \in (k, n-k)} \hat{f}(S)^2 \mathbb{E}[\chi_S(D^2)],$$

where D^2 is the sum of two independent copies of D , which is also k -wise uniform. Let $G := \{x \in \{-1, 1\}^n : |\sum_{i=1}^n x_i| \leq (\frac{kn^3}{2e})^{1/4}\}$, and D_G be the conditional distribution of D^2 supported on G . By Fact 29, the distribution D_G is $\Pr[D \notin G]$ -close to D^2 . As $\sum_{S \subseteq [n]} \hat{f}(S)^2 \leq 1$, we have

$$\left| \sum_{S: |S| \in (k, n-k)} \hat{f}(S)^2 \mathbb{E}[(D^2)^S] \right| \leq \left| \sum_{S: |S| \in (k, n-k)} \hat{f}(S)^2 \mathbb{E}[D_G^S] \right| + \Pr[D \notin G].$$

Applying Corollary 28 (to the even integer $k - 1$ or k), we have

$$\Pr[D \notin G] \leq \left(\frac{2k}{en} \right)^{\frac{k-1}{4}}. \quad (24)$$

We now bound the first term on the right hand side as follows. Fix a string $z \in G$. As $|\sum_{|S|=\ell} z^S| = |z^{[n]} \sum_{|S|=n-\ell} z^S| = |\sum_{|S|=n-\ell} z^S|$, by Fact 26,

$$\left| \sum_{S: |S| \in (k, n-k)} \hat{f}(S)^2 z^S \right| = \sum_{\ell=k+1}^{n-k-1} \hat{f}([\ell])^2 \left| \sum_{|S|=\ell} z^S \right| \leq 2 \sum_{\ell=k+1}^{n/2} \frac{1}{\binom{n}{\ell}} \left| \sum_{|S|=\ell} z^S \right|.$$

We separate the sum into two parts depending on $\ell \leq n/5$ and bound each of them individually. First, using Corollary 16, we have

$$\sum_{\ell=k+1}^{n/2} \frac{1}{\binom{n}{\ell}} \left| \sum_{|S|=\ell} z^S \right| \leq \sum_{\ell=k+1}^{n/5} \left(\frac{\ell}{n} + \sqrt{\frac{k}{2en}} \right)^{\ell/2} \leq 2 \left(2\sqrt{\frac{k}{2en}} \right)^{k/2} \leq 2 \left(\frac{2k}{en} \right)^{k/4}, \quad (25)$$

because each term in the sum is at most half its previous term, and so the sum can be bounded by twice the first term. For the remaining sum (from $\ell = \max\{k, n/5\} + 1$ to $n/2$), note that the binary entropy function $H(x) := x \log_2(1/x) + (1-x) \log_2(1/(1-x))$ is increasing on $[0, 1/2]$. In particular, we have $H(1/5) - \frac{1}{\sqrt{2e}} \geq 1/4$. By Stirling’s approximation, we have $\binom{n}{\ell} \geq \frac{1}{n^2} 2^{nH(\frac{\ell}{n})}$ (see [24] for a proof). Applying Corollary 18 with these facts, we have

$$\sum_{\ell=\max\{n/5, k\}+1}^{n/2} \frac{1}{\binom{n}{\ell}} \left| \sum_{|S|=\ell} z^S \right| \leq \sum_{\ell=\max\{n/5, k\}+1}^{n/2} n^2 \cdot 2^{-\frac{n}{2} \left(H(\frac{\ell}{n}) - \frac{1}{\sqrt{2e}} \right)} \leq 2^{-n/10}.$$

Combining this with (24) and (25) gives an error of $\left(\frac{2k}{en} \right)^{\frac{k-1}{4}} + 2 \left(\frac{2k}{en} \right)^{\frac{k}{4}} + 2^{-n/10} \leq 6 \left(\frac{2k}{en} \right)^{\frac{k-1}{4}}$. ◀

6 Bounds on Krawtchouk polynomials

In this section, we prove our upper and lower bounds on Krawtchouk polynomials (Corollary 16, Proposition 17, , and Claim 15). Corollary 16 follows directly from Lemma 36, which is a general upper bound on the elementary symmetric polynomials $\sum_{|S|=\ell} y^S$ that holds for arbitrary real tuples $y \in \mathbb{R}^n$, not only for $y \in \{-1, 1\}^n$.

► **Lemma 36.** *Let $y = (y_1, \dots, y_n) \in \mathbb{R}^n$. For every $1 \leq \ell \leq n$, we have*

$$\left| \sum_{S \subseteq [n]: |S|=\ell} y^S \right| \leq \binom{n}{\ell} \left(\frac{\ell-1}{n-1} \cdot \frac{\sum_{i=1}^n y_i^2}{n} + \left(1 - \frac{\ell-1}{n-1} \right) \cdot \frac{(\sum_{i=1}^n y_i)^2}{n^2} \right)^{\frac{\ell}{2}}.$$

with equality if and only if $y_1 = \dots = y_n$ or $\ell = 1$.

Specializing to $y \in \{-1, 1\}^n$, the elementary symmetric polynomial $\sum_{|S|=\ell} y^S$ is simply the degree- ℓ (shifted) Krawtchouk polynomial $\bar{K}(\ell, |y|)$. In this case, we always have $\sum_{i=1}^n y_i^2 = n$, and hence we obtain Corollary 16.

Corollary 16 appeared in [11] with an extra factor of c^ℓ . Lemma 36 shows that the same inequality holds even when y_1, \dots, y_n are arbitrary real numbers. A similar-looking but incomparable inequality, first proved in [33], showed that

$$\left| \sum_{S \subseteq [n]: |S|=\ell} y^S \right| \leq O\left(\frac{k}{\ell}\right)^{\frac{\ell}{2}} \max_{k' \in \{k, k+1\}} \left(\left| \sum_{S \subseteq [n]: |S|=k'} y^S \right| \right)^{\frac{\ell}{k'}}, \tag{26}$$

Using a different approach, Tao [59] recently sharpened this inequality to

$$\left| \sum_{S \subseteq [n]: |S|=\ell} y^S \right| \leq O\left(\frac{1}{\ell}\right)^{\frac{\ell}{2}} \max_{k' \in \{k, k+1\}} \left(\left| \sum_{S \subseteq [n]: |S|=k'} y^S \right| \right)^{\frac{\ell}{k'}}, \tag{27}$$

confirming a conjecture made on MathOverflow, see <https://mathoverflow.net/q/446254>. Note that specializing to the case $k = 1$, and using the inequality $|\sum_{1 \leq i < j \leq n} y_i y_j| \leq \frac{1}{2} \sum_{i=1}^n y_i^2$, both (26) and (27) imply a weaker form of Lemma 36. In the other direction, Tao [58] observed that one cannot replace the quantity $\sum_{i=1}^n y_i^2$ in Lemma 36 with $|\sum_{1 \leq i < j \leq n} y_i y_j|$, as otherwise when n is the square of an even number, for $y \in \{-1, 1\}^n$ such that $\sum_{i=1}^n y_i = \sqrt{n}$, we have $\sum_{1 \leq i < j \leq n} y_i y_j = 0$ and the inequality fails at $\ell = n$.

We note that Lemma 36 can be obtained by a slight modification of both proofs in [33, 59]. Here we follow the approach taken in [59], as it gives a sharper constant and the argument is cleaner.

6.1 Proof of Lemma 36

Our approach is based on [59], which relies on several basic properties of real-rooted polynomials. We say an $(n + 1)$ -tuple of real numbers (s_0, \dots, s_n) is *attainable* if the polynomial

$$\sum_{k=0}^n (-1)^k \binom{n}{k} s_k z^{n-k}$$

is monic with all roots real. By its real-rootedness, we can factor the polynomial as

$$\sum_{k=0}^n (-1)^k \binom{n}{k} s_k(y) z^{n-k} = \prod_{i=1}^n (z - y_i)$$

18:22 Pseudorandomness, Symmetry, Smoothing: I

for some real numbers y_1, \dots, y_n , where

$$s_k(y) = \frac{1}{\binom{n}{k}} \sum_{|S|=k} y^S = \frac{1}{\binom{n}{k}} \sum_{1 \leq i_1 < \dots < i_k \leq n} y_{i_1} \cdots y_{i_k}.$$

Conversely, given an n -tuple of real numbers $y = (y_1, \dots, y_n)$, we can define $s_k(y)$ as above to obtain an attainable tuple. We will use the following truncation property of attainable tuples.

► **Fact 37 (Truncation).** *Let (s_0, \dots, s_n) be an attainable tuple. Then (s_0, \dots, s_ℓ) is attainable for every $1 \leq \ell \leq n$.*

Proof. It suffices to show that (s_0, \dots, s_{n-1}) is attainable. Write $s_k := s_k(y_1, \dots, y_n)$ for some real numbers y_1, \dots, y_n . By Rolle's theorem, between every two consecutive real roots of a polynomial, there is a real root of its derivative. Thus the derivative of a real-rooted polynomial is also real-rooted. Therefore, the polynomial

$$\begin{aligned} \frac{1}{n} \cdot \frac{d}{dz} \sum_{k=0}^n (-1)^k \binom{n}{k} s_k(y) z^{n-k} &= \sum_{k=0}^{n-1} (-1)^k \frac{n-k}{n} \binom{n}{k} s_k(y) z^{n-1-k} \\ &= \sum_{k=0}^{n-1} (-1)^k \binom{n-1}{k} s_k(y) z^{n-1-k} \end{aligned}$$

is monic and real-rooted, showing that (s_0, \dots, s_{n-1}) is also attainable. ◀

► **Remark 38.** One should view s_ℓ as $s_\ell = \prod_{i=1}^{\ell} y'_i$ for some $y'_1, \dots, y'_\ell \in \mathbb{R}$, instead of $s_\ell = \binom{n}{\ell}^{-1} \sum_{|S|=\ell} y^S$ for some $y_1, \dots, y_n \in \mathbb{R}$ such that $s_n = \prod_{i=1}^n y_i$.

Lemma 36 relies on the following slight refinement in Tao's argument.

► **Lemma 39.** *Let (s_0, \dots, s_n) be an attainable tuple. Then for every $1 \leq \ell \leq n$,*

$$|s_\ell|^{\frac{2}{\ell}} \leq (\ell - 1) \cdot (s_1^2 - s_2) + s_1^2.$$

Proof. By the truncation property (Fact 37), it suffices to consider the case $\ell = n$. Write $s_k := s_k(y_1, \dots, y_n)$ for some $y = (y_1, \dots, y_n) \in \mathbb{R}^n$. By the AM-GM inequality, we have

$$|s_n(y)|^{\frac{2}{n}} = (y_1^2 \cdots y_n^2)^{\frac{1}{n}} \leq \frac{1}{n} \sum_{i=1}^n y_i^2.$$

By the Newton identity we have

$$\sum_{i=1}^n y_i^2 = \left(\sum_{i=1}^n y_i \right)^2 - 2 \sum_{1 \leq i < j \leq n} y_i y_j = n^2 s_1(y_1, \dots, y_n)^2 - 2 \binom{n}{2} s_2(y_1, \dots, y_n).$$

Therefore,

$$|s_n(y)|^{\frac{2}{n}} \leq n s_1(y)^2 - (n-1) s_2(y) = (n-1) (s_1(y)^2 - s_2(y)) + s_1(y)^2. \quad \blacktriangleleft$$

Lemma 36 immediately follows from Lemma 39 by un-normalizing s_ℓ, s_1 and s_2 .

Proof of Lemma 36. Let $S_k(y) := \binom{n}{k} s_k(y) = \sum_{|S|=k} y^S$. Applying Lemma 39, we have

$$\begin{aligned} |S_\ell|^{\frac{2}{\ell}} &\leq \binom{n}{\ell}^{\frac{2}{\ell}} \left((\ell-1)(s_1^2 - s_2) + s_1^2 \right) \\ &= \binom{n}{\ell}^{\frac{2}{\ell}} \left((\ell-1) \left(\frac{S_1^2}{n^2} - \frac{2S_2}{n(n-1)} \right) + \frac{S_1^2}{n^2} \right) \\ &= \binom{n}{\ell}^{\frac{2}{\ell}} \left((\ell-1) \left(\frac{S_1^2 - 2S_2}{n(n-1)} - \frac{S_1^2}{n^2(n-1)} \right) + \frac{S_1^2}{n^2} \right) \\ &= \binom{n}{\ell}^{\frac{2}{\ell}} \left(\frac{\ell-1}{n-1} \left(\frac{S_1^2 - 2S_2}{n} - \frac{S_1^2}{n^2} \right) + \frac{S_1^2}{n^2} \right) \\ &= \binom{n}{\ell}^{\frac{2}{\ell}} \left(\frac{\ell-1}{n-1} \left(\frac{S_1^2 - 2S_2}{n} \right) + \left(1 - \frac{\ell-1}{n-1} \right) \frac{S_1^2}{n^2} \right). \end{aligned}$$

Applying Newton's identity, i.e., $S_1^2 - 2S_2 = \sum_{i=1}^n y_i^2$, completes the proof. \blacktriangleleft

We now prove Proposition 17. We note that a similar argument also appears in [51, Lemma 2.1]. For completeness we provide a self-contained proof here.

Proof of Proposition 17. First note

$$(1+z)^{(n+t)/2} (1-z)^{(n-t)/2} = \sum_{\ell=0}^n \bar{K}(\ell, t) z^\ell.$$

Let $r = |z|$. The logarithmic function is known to be concave:

$$\alpha \log_2(u) + (1-\alpha) \log_2(v) \leq \log_2(\alpha u + (1-\alpha)v).$$

for any positive u and v . Using concavity and the observation $|1+z|^2 + |1-z|^2 = 2 + 2|z|^2 = 1 + z + \bar{z} + |z|^2 + 1 - z - \bar{z} + |z|^2 = 2 + 2|z|^2 = 2 + 2r^2$ gives

$$\begin{aligned} H(\alpha) + \log_2(|1+z|^{2\alpha} |1-z|^{2(1-\alpha)}) &= \alpha \log_2 \left(\frac{|1+z|^2}{\alpha} \right) + (1-\alpha) \log_2 \left(\frac{|1-z|^2}{1-\alpha} \right) \\ &\leq \log_2(|1+z|^2 + |1-z|^2) \\ &= \log_2(2 + 2r^2). \end{aligned}$$

For an integer ℓ with $0 \leq \ell \leq n$ consider the Laurent polynomial

$$p(z) = \frac{(1+z)^{(n+t)/2} (1-z)^{(n-t)/2}}{z^\ell}.$$

If $r^2 = \beta/(1-\beta)$, then we have

$$\begin{aligned} \log |p(z)| &= \frac{n}{2} \left(\log_2(|1+z|^{2\alpha} |1-z|^{2(1-\alpha)}) - \beta \log_2(|z|^2) \right) \\ &\leq \frac{n}{2} \left(\log_2(2 + 2r^2) - \beta \log_2(r^2) - H(\alpha) \right) \\ &= \frac{n}{2} \left(\log_2 \frac{2}{1-\beta} - \beta \log_2 \frac{\beta}{1-\beta} - H(\alpha) \right) \\ &= \frac{n}{2} \left(1 + H(\beta) - H(\alpha) \right). \end{aligned}$$

The coefficient of $1 = z^0$ in $p(z)$ is $\bar{K}(\ell, t)$, so it follows that $\bar{K}(\ell, t) = \int_0^1 p(re^{2\pi i\theta}) d\theta$. We conclude that

$$\log_2 |\bar{K}(\ell, t)| \leq \log_2 \left(\int_0^1 |p(re^{2\pi i\theta})| d\theta \right) \leq \max_{|z|=r} \log_2 |p(z)| \leq \frac{n}{2} \left(1 - H(\alpha) + H(\beta) \right). \quad \blacktriangleleft$$

6.2 Lower bound on Krawtchouk polynomials

In this section, we prove Claim 15. It follows from an inequality on Krawtchouk polynomials which appears to be well known in the coding theory literature [44, 36, 38, 37], and essentially follows from Newton's inequality.

For convenience we will work with the standard (non-shifted) definition of Krawtchouk polynomials $K(\ell, t) = \overline{K}(\ell, n - 2t)$. Note that in the claim below, we intentionally swap t and ℓ .

▷ **Claim 40.** $K(n/2 - t, \ell) \geq \binom{n}{n/2-t} (t/n)^\ell$ for $t \geq \sqrt{\ell(n - \ell)}$.

Claim 40 follows from the fact that $K(t, 0) = \binom{n}{t}$ and then applying the following lemma iteratively ℓ times.

► **Lemma 41** (Theorem 8 in [38]). *For ℓ, i such that $(n - 2i)^2 \geq 4\ell(n - \ell)$ (so that $s = \sqrt{(n - 2i)^2 - 4\ell(n - \ell)}$ is real and nonnegative),*

$$\frac{K(i, \ell + 1)}{K(i, \ell)} > \frac{n - 2i + s}{2(n - \ell)} \geq \frac{n - 2i}{2n}.$$

We can now prove Claim 15 using the following fact and translating the statement in terms of $\overline{K}(\cdot, \cdot)$.

► **Fact 42.** $\binom{n}{t} K(\ell, t) = \binom{n}{\ell} K(t, \ell)$.

Proof of Claim 15. We have

$$\overline{K}(\ell, t) = K\left(\ell, \frac{n}{2} - \frac{t}{2}\right) = \frac{\binom{n}{\ell}}{\binom{n}{\frac{n}{2} + \frac{t}{2}}} K\left(\frac{n}{2} - \frac{t}{2}, \ell\right) \geq \binom{n}{\ell} \left(\frac{t}{2n}\right)^\ell. \quad \triangleleft$$

References

- 1 Thomas D. Ahle. *Asymptotic Tail Bound and Applications*. Available at <https://thomasahle.com/papers/tails.pdf>, 2017.
- 2 Miklós Ajtai. Σ_1^1 -formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
- 3 Miklos Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant-depth circuits. *Advances in Computing Research - Randomness and Computation*, 5:199–223, 1989.
- 4 Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing k -wise and almost k -wise independence. In *ACM Symp. on the Theory of Computing (STOC)*, pages 496–505, 2007. doi:10.1145/1250790.1250863.
- 5 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized algorithm for the maximal independent set problem. *Journal of Algorithms*, 7:567–583, 1986.
- 6 Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k -wise independence versus k -wise independence. *Inform. Process. Lett.*, 88(3):107–110, 2003. doi:10.1016/S0020-0190(03)00359-4.
- 7 Louay Bazzi. Polylogarithmic independence can fool DNF formulas. In *48th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 63–73, 2007.
- 8 Louay Bazzi. Entropy of weight distributions of small-bias spaces and pseudobinomality. In *Computing and combinatorics*, volume 9198 of *Lecture Notes in Comput. Sci.*, pages 495–506. Springer, Cham, 2015. doi:10.1007/978-3-319-21398-9_39.
- 9 Louay Bazzi. Weight distribution of cosets of small codes with good dual properties. *IEEE Trans. Inform. Theory*, 61(12):6493–6504, 2015. doi:10.1109/TIT.2015.2487348.

- 10 Itai Benjamini, Ori Gurel-Gurevich, and Ron Peled. On k -wise independent distributions and boolean functions, 2012. [arXiv:1201.3261](#).
- 11 Jarosław Blasiok, Peter Ivanov, Yaonan Jin, Chin Ho Lee, Rocco A. Servedio, and Emanuele Viola. Fourier growth of structured \mathbb{F}_2 -polynomials and applications. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 207 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 53, 20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.
- 12 Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory Comput.*, 9:283–292, 2013. [doi:10.4086/toc.2013.v009a007](#).
- 13 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. on Computing*, 39(6):2464–2486, 2010.
- 14 Ravi Boppana, Johan Håstad, Chin Ho Lee, and Emanuele Viola. Bounded independence versus symmetric tests. *ACM Trans. Comput. Theory*, 11(4):Art. 21, 27, 2019. [doi:10.1145/3337783](#).
- 15 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *J. of the ACM*, 57(5), 2010.
- 16 Mark Bun and Thomas Steinke. Weighted polynomial approximations: limits for learning and pseudorandomness. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 40 of *LIPICs. Leibniz Int. Proc. Inform.*, pages 625–644. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015.
- 17 J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *J. of Computer and System Sciences*, 18(2):143–154, 1979.
- 18 Eshan Chattopadhyay, Jason Gaitonde, Chin Ho Lee, Shachar Lovett, and Abhishek Shetty. Fractional pseudorandom generators from any Fourier level. In *36th Computational Complexity Conference*, volume 200 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 10, 24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.
- 19 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom generators from polarizing random walks. *Theory Comput.*, 15:Paper No. 10, 26, 2019. [doi:10.4086/toc.2019.v015a010](#).
- 20 Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal. Pseudorandom generators from the second Fourier level and applications to AC^0 with parity gates. In *10th Innovations in Theoretical Computer Science*, volume 124 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 22, 15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.
- 21 Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. Improved pseudorandomness for unordered branching programs through local monotonicity. In *STOC’18—Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 363–375. ACM, New York, 2018. [doi:10.1145/3188745.3188800](#).
- 22 Lijie Chen, Xin Lyu, Avishay Tal, and Hongxun Wu. New PRGs for unbounded-width/adaptive-order read-once branching programs. In *50th International Colloquium on Automata, Languages, and Programming*, volume 261 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 39, 20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. [doi:10.4230/lipics.icalp.2023.39](#).
- 23 Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem or t -resilient functions (preliminary version). In *26th Symposium on Foundations of Computer Science*, pages 396–407, Portland, Oregon, 21–23 October 1985. IEEE.
- 24 Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006.
- 25 Harm Derksen, Peter Ivanov, Chin Ho Lee, and Emanuele Viola. Pseudorandomness, symmetry, smoothing: II, 2024.
- 26 Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM J. on Computing*, 39(8):3441–3462, 2010.

- 27 Dean Doron, Pooya Hatami, and William M. Hoza. Near-optimal pseudorandom generators for constant-depth read-once formulas. In *34th Computational Complexity Conference*, volume 137 of *LIPICs. Leibniz Int. Proc. Inform.*, pages 16:1–16:34. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICs.CCC.2019.16.
- 28 Dean Doron, Pooya Hatami, and William M. Hoza. Log-seed pseudorandom generators via iterated restrictions. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPICs*, pages 6:1–6:36. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.CCC.2020.6.
- 29 Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2018.
- 30 Dmitry Gavinsky, Shachar Lovett, and Srikanth Srinivasan. Pseudorandom generators for read-once acc⁰. In *IEEE Conf. on Computational Complexity (CCC)*, pages 287–297, 2012. doi:10.1109/CCC.2012.37.
- 31 Parikshit Gopalan, Daniel Kane, and Raghu Meka. Pseudorandomness via the discrete fourier transform. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 903–922, 2015. doi:10.1109/FOCS.2015.60.
- 32 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2012.
- 33 Parikshit Gopalan and Amir Yehudayoff. Concentration for limited independence via inequalities for the elementary symmetric polynomials. *Theory Comput.*, 16:Paper No. 17, 29, 2020. doi:10.4086/toc.2020.v016a017.
- 34 Elad Haramaty, Chin Ho Lee, and Emanuele Viola. Bounded independence plus noise fools products. *SIAM J. on Computing*, 47(2):295–615, 2018.
- 35 Pooya Hatami and William Hoza. Theory of unconditional pseudorandom generators. *Electron. Colloquium Comput. Complex.*, TR23-019, 2023.
- 36 Gil Kalai and Nathan Linial. On the distance distribution of codes. *IEEE Trans. Inform. Theory*, 41(5):1467–1472, 1995. doi:10.1109/18.412711.
- 37 Naomi Kirshner and Alex Samorodnitsky. A moment ratio bound for polynomials and some extremal properties of Krawchouk polynomials and Hamming spheres. *IEEE Trans. Inform. Theory*, 67(6, part 1):3509–3541, 2021. doi:10.1109/TIT.2021.3071597.
- 38 Ilya Krasikov. Nonnegative quadratic forms and bounds on orthogonal polynomials. *J. Approx. Theory*, 111(1):31–49, 2001. doi:10.1006/jath.2001.3570.
- 39 Ilya Krasikov and Simon Litsyn. Survey of binary Krawtchouk polynomials. In *Codes and association schemes (Piscataway, NJ, 1999)*, volume 56 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 199–211. Amer. Math. Soc., Providence, RI, 2001. doi:10.1090/dimacs/056/16.
- 40 Chin Ho Lee. Fourier bounds and pseudorandom generators for product tests, 2019.
- 41 Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13, 2017.
- 42 Chin Ho Lee and Emanuele Viola. More on bounded independence plus noise: Pseudorandom generators for read-once polynomials. *Theory of Computing*, 16:1–50, 2020. URL: <https://www.khoury.northeastern.edu/home/viola/papers/LV-rop.pdf>.
- 43 Vladimir I. Levenshtein. Krawtchouk polynomials and universal bounds for codes and designs in Hamming spaces. *IEEE Trans. Inform. Theory*, 41(5):1303–1321, 1995. doi:10.1109/18.412678.
- 44 Robert J. McEliece, Eugene R. Rodemich, Howard Rumsey, Jr., and Lloyd R. Welch. New upper bounds on the rate of a code via the Delsarte-MacWilliams inequalities. *IEEE Trans. Inform. Theory*, IT-23(2):157–166, 1977. doi:10.1109/tit.1977.1055688.

- 45 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *STOC'19—Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 626–637. ACM, New York, 2019. doi:10.1145/3313276.3316319.
- 46 Raghu Meka and David Zuckerman. Small-bias spaces for group products. In *Approximation, randomization, and combinatorial optimization*, volume 5687 of *Lecture Notes in Comput. Sci.*, pages 658–672. Springer, Berlin, 2009. doi:10.1007/978-3-642-03685-9_49.
- 47 J. Naor and M. Naor. Small-bias probability spaces: efficient constructions and applications. In *22nd ACM Symp. on the Theory of Computing (STOC)*, pages 213–223. ACM, 1990.
- 48 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.
- 49 Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.
- 50 Ryan O’Donnell and Yu Zhao. On Closeness to k -Wise Uniformity. 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 54:1–54:19, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.54.
- 51 Yury Polyanskiy. Hypercontractivity of spherical averages in Hamming space. *SIAM J. Discrete Math.*, 33(2):731–754, 2019. doi:10.1137/15M1046575.
- 52 C. Radhakrishna Rao. Factorial experiments derivable from combinatorial arrangements of arrays. *Suppl. J. Roy. Statist. Soc.*, 9:128–139, 1947.
- 53 Alexander A. Razborov. A simple proof of Bazzi’s theorem. *ACM Transactions on Computation Theory (TOCT)*, 1(1), 2009.
- 54 Omer Reingold, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for regular branching programs via Fourier analysis. In *Workshop on Randomization and Computation (RANDOM)*, pages 655–670, 2013.
- 55 Jad Silbak, Swastik Kopparty, and Ronen Shaltiel. Quasilinear time list-decodable codes for space bounded channels. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 302–333. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00028.
- 56 Thomas Steinke, Salil Vadhan, and Andrew Wan. Pseudorandomness and Fourier-growth bounds for width-3 branching programs. *Theory Comput.*, 13:Paper No. 12, 50, 2017. doi:10.4086/toc.2017.v013a012.
- 57 Avishay Tal. Tight bounds on the fourier spectrum of AC0. In *Conf. on Computational Complexity (CCC)*, pages 15:1–15:31, 2017. doi:10.4230/LIPIcs.CCC.2017.15.
- 58 Terence Tao. Personal communication, 2023.
- 59 Terence Tao. A Maclaurin type inequality, 2023. arXiv:2310.05328.
- 60 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/04000000010.
- 61 Emanuele Viola. On approximate majority and probabilistic time. *Computational Complexity*, 18(3):337–375, 2009.
- 62 Emanuele Viola. Correlation bounds against polynomials, a survey, 2022.
- 63 Emanuele Viola. Pseudorandom bits and lower bounds for randomized turing machines. *Theory of Computing*, 18(10):1–12, 2022.
- 64 Emanuele Viola. Mathematics of the impossible: The uncharted complexity of computation, 2023.

Information Dissemination via Broadcasts in the Presence of Adversarial Noise

Klim Efremenko ✉ 

Ben-Gurion University of the Negev, Beer-Sheva, Israel

Gillat Kol ✉ 

Princeton University, NJ, USA

Dmitry Paramonov ✉ 

Princeton University, NJ, USA

Ran Raz ✉ 

Princeton University, NJ, USA

Raghuvansh R. Saxena ✉

Tata Institute of Fundamental Research, Mumbai, India

Abstract

We initiate the study of *error correcting codes* over the multi-party *adversarial broadcast channel*. Specifically, we consider the classic *information dissemination problem* where n parties, each holding an input bit, wish to know each other's input. For this, they communicate in rounds, where, in each round, one designated party sends a bit to all other parties over a channel governed by an adversary that may corrupt a constant fraction of the received communication. We mention that the dissemination problem was studied in the *stochastic* noise model since the 80's.

While stochastic noise in multi-party channels has received quite a bit of attention, the case of adversarial noise has largely been avoided, as such channels cannot handle more than a $\frac{1}{n}$ -fraction of errors. Indeed, this many errors allow an adversary to completely corrupt the incoming or outgoing communication for one of the parties and fail the protocol. Curiously, we show that by eliminating these “trivial” attacks, one can get a simple protocol resilient to a constant fraction of errors. Thus, a model that rules out such attacks is both necessary and sufficient to get a resilient protocol.

The main shortcoming of our dissemination protocol is its length: it requires $\Theta(n^2)$ communication rounds whereas n rounds suffice in the absence of noise. Our main result is a matching lower bound of $\Omega(n^2)$ on the length of any dissemination protocol in our model. Our proof first “gets rid” of the channel noise by converting it to a form of “input noise”, showing that a noisy dissemination protocol implies a (noiseless) protocol for a version of the *direct sum gap-majority* problem. We conclude the proof with a tight lower bound for the latter problem, which may be of independent interest.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Radio Networks, Interactive Coding, Error Correcting Codes

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.19

Funding *Klim Efremenko*: Supported by European Research Council Grant No. 949707.

Gillat Kol: Supported by the National Science Foundation CAREER award CCF-1750443.

Ran Raz: Supported by a Simons Investigator Award and by the National Science Foundation grant No. CCF-2007462.

1 Introduction

We initiate the study of *error correcting codes* over the multi-party *adversarial broadcast channel*, where n parties take turns broadcasting a bit to all other parties, but an adversary may corrupt a constant fraction of the received bits. Multi-party broadcast channels were



© Klim Efremenko, Gillat Kol, Dmitry Paramonov, Ran Raz, and Raghuvansh R. Saxena;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 19; pp. 19:1–19:33



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



studied under various noise models in many recent works. However, almost all prior work assumed that the noise is stochastic, meaning that each sent message is corrupted with some small constant probability.

The reason the adversarial noise model has received considerably less attention is due to the fundamental limitation that any scheme, regardless of its rate, cannot withstand an adversarial noise rate exceeding $\frac{1}{n}$. To illustrate, with a budget of $\frac{1}{n}$ -fraction of corruptions, the adversary can corrupt all messages broadcast by the participant who communicates the least, thereby obstructing the other parties from successfully computing a function that relies on this individual's input. Likewise, within the same budget, the adversary can disrupt all messages received by one of the participants, preventing them from producing a correct output.

The starting point of this paper is the observation that by excluding the two simple adversarial attacks mentioned earlier, we can circumvent the nonexistence of protocols capable of withstanding adversarial corruptions beyond a fraction of $\frac{1}{n}$. Specifically, we consider the adversarial channel where the adversary can corrupt any number of messages, provided that they do not corrupt more than a θ -fraction of the messages received by each party and a θ -fraction of the messages sent by each party¹, for some constant $\theta > 0$. We call such adversaries θ -limited.

Indeed, consider the following simple protocol for the *information dissemination* problem, where the input to each party is a bit and all parties wish to know all inputs: in the first half of the protocol, each party broadcasts their input bit the same number of times, and then each party computes the majority of the bits they received from each of the other parties. In the second half of the rounds, each party broadcasts an error correcting code of all the majority bits they computed. It is not hard to show that this protocol is resilient to θ -limited adversaries for some constant θ .²

Although our protocol exhibits good error resilience, a notable drawback is its rate, which is at most $\frac{1}{n}$. This is because, in the second half of the protocol, each of the n parties broadcasts the encoding of all n of their majority bits with an error correcting code, resulting in a length of at least n^2 bits. This situation prompts the following question:

Is there an information dissemination protocol robust to θ -limited adversaries with constant rate, or at least $\omega(\frac{1}{n})$ rate?

1.1 Our Result

We answer this question in the negative, showing that the above simple protocol is essentially optimal.

► **Theorem 1** (Informal; see Theorem 6). *Every protocol for information dissemination that is resilient to a 0.01 -limited adversary has length $\Omega(n^2)$.*

¹ More formally, if a party broadcasts in t rounds, then the adversary may corrupt up to θtn out of the tn messages received by the parties in those t rounds.

² The argument is that for every $i \in [n]$, in each half of the protocol at most 2θ -fraction of the messages received by party i are corrupted. Thus, party i correctly decodes at least $(1 - \mathcal{O}(\theta))$ -fraction of the error correcting codes sent in the second half. This means that if party i outputs a wrong guess for player j 's input, then the bit for party j must be incorrect in at least $(\frac{1}{2} - \mathcal{O}(\theta))$ -fraction of the broadcast codes. This means that at least $\theta' = (\frac{1}{4} - \mathcal{O}(\theta))$ -fraction of the transmissions of party j in the first half were corrupted. By choosing θ such that $\theta' > 2\theta$, we get that the output of all the parties is correct. See Section 4.

Technique

The proof of Theorem 1 consists of two steps. The first step *converts the noise in the channel to noise in the inputs*. Roughly speaking, we show that any protocol for information dissemination in our noisy model implies a protocol for solving a direct-sum gap-majority problem in the absence of noise³. Here, one copy of the gap-majority problem, denoted GapMaj_n , is the following: each party gets a bit with the promise that at least 0.9-fraction of the parties received the same bit. The parties' goal is to all output the majority bit. In the direct sum gap-majority problem, denoted here GapMaj_n^m , each party gets m bits, and we are promised that, for all j , the j -th bits of all the parties are an instance for GapMaj_n . Consider that one can view the majority bit for each copy j as the “true j -th bit” and view the j -th input bit of each player as a *noisy version* of this bit. In this sense, this indeed converts the noise in the channel to a noise in the inputs.

We then proceed to prove a lower bound on the communication cost of GapMaj_n^m over the noiseless channel. We show:

► **Theorem 2** (Informal; see Theorem 10). *Every protocol for GapMaj_n^m with $m = \Theta(n)$, has length $\Omega(n^2)$.*

We note that our above definition of the direct sum problem is different from other definitions in the literature on one crucial point: *if the promise is violated, even for a single copy, any output is accepted*. In other words, as is usually the case, if all m copies satisfy the promise, the parties need to solve all copies. However, if the promise is violated for some of the copies, we don't require the protocol to solve the copies on which the promise does hold. Since our definition is easier to be satisfied by an algorithm, the lower bound in Theorem 2 is stronger. Because the relevant, known direct sum theorems only rule out algorithms that solve all the copies where the promise is satisfied, they are insufficient for our purpose (see more about this in Sections 1.2 and 2.2).

To prove our lower bound, we first note that Theorem 2 is essentially tight, as with $\mathcal{O}(n^2)$ communication, the parties can exchange their entire input. Moreover, as at least $\Omega(n)$ parties need to speak to solve the single-copy problem, Theorem 2 implies that the best protocol essentially solves each copy separately. Put differently, one can say that Theorem 2 is equivalent to the statement that trying to correlate the copies of gap-majority does not help the parties in solving the GapMaj_n^m problem. Interestingly, our proof establishes this in a pretty strong sense, showing that even trying to correlate three copies does not help the protocol, as it shows the result only assuming that the copies are *pairwise independent* at the end of the protocol.

In other words, we prove that the only way to make progress towards solving the GapMaj_n^m problem is to try to create a lot of correlations between pairs of copies. However, as this is only a constant factor away from giving a lot of information about individual copies (which is bounded by the overall communication), the number of these correlations can also be bounded by a constant times the overall communication, hence the lower bound. For more details, see Section 2.

³ We mention that this step is inspired by the beautiful work of [32] that gives a lower bound under stochastic noise by lower bounding a different problem where the noise is in the inputs. However, our implementation is largely different, see Section 2.1.

1.2 Related Work

Dissemination over the stochastic broadcast channel

El Gamal [25] initiated the study of the noisy broadcast model as a simple abstraction for the effect of noise on highly distributed wireless systems. The noise in his model was stochastic – in each round the bit received by each party is flipped with some constant probability $\epsilon > 0$, independently. El Gamal asked whether there is a communication-efficient information dissemination protocol over this channel. The answer came from Gallager [24], who gave an elegant $\mathcal{O}(n \log \log n)$ -round protocol, which was later proved to be optimal by the beautiful paper [32]. Variants of El Gamal’s stochastic noisy broadcast channel were studied in many follow up works [24, 48, 42, 23, 45, 32, 10, 17, 14, 15]. We mention that our initial motivation for the study in this paper was the question of whether communication-efficient protocols like Gallager’s were also possible in the presence of adversarial noise.

Interactive coding

In this work we consider the information dissemination problem, which is, perhaps, the most basic multi-party problem. It can be viewed as a generalization of the classical coding task to the multi-sender, multi-receiver setting (in traditional coding there is a single sender and a single receiver and the goal is to transfer a message from the sender to the receiver). It can be shown that a dissemination protocol with a certain resilience implies a protocol for any other problem, as the parties can first exchange their inputs and then compute the output themselves. Of course, this protocol is not always practical, as the input size may be much greater than the communication required to compute a solution to the problem.

The field of *interactive coding* aims to make this practical by converting (general) protocols designed to work over a noiseless channel to noise resilient protocols with a small overhead in the communication. The study of interactive codes was initiated by a seminal paper of Schulman [47] that considered two-party protocols and was the topic of many works since. Interactive codes for multi-party distributed channels were also studied, including codes for peer-to-peer networks [46, 31, 36, 43, 35, 2, 5, 1, 28, 9, 29, 30] and codes for various types of broadcast channels [24, 48, 42, 23, 45, 32, 10, 17, 11, 18, 20, 14, 44, 15, 16].

Peer-to-peer with adversarial noise

In this paper, we consider the broadcast channel under adversarial noise. The case of adversarial noise was previously considered in different peer-to-peer settings, where the parties are nodes in a graph and a node can send (potentially different) messages to its neighbors.

The work of [35] gives an interactive coding result in the synchronous, “fully utilized” model, where the communication is in rounds, and in each round each node sends a message to all other nodes. They show a scheme for converting any noiseless protocol to a protocol that is robust against $\Theta(\frac{1}{n})$ -fraction of adversarial errors with multiplicative overhead of $\mathcal{O}(|E| \log n / n)$ in the communication, where $|E|$ is the number of edges in the graph and n is the number of vertices. [36] consider the synchronous, non-fully-utilized model, and show that if the network graph contains the star topology, then a noiseless protocol can be converted to a protocol that is robust against $\Theta(\frac{1}{n})$ -fraction of adversarial errors and has length linear in the length of the original protocol. [43] improve the communication balance of the [36] scheme. [9] consider the asynchronous setting and give an interactive coding scheme with error resilience $\Theta(\frac{1}{n})$ and multiplicative overhead of $\mathcal{O}(n \log^2 n)$ in the communication.

In all the above results, the noise tolerance of $\Theta(\frac{1}{n})$ is optimal, up to constants. For example, it is noted by [36] that, “by investing $\frac{1}{n}$ -fraction of error an adversary can completely silence a party (the quietest party)”. Recall that we get around this attack by forcing the adversary to not corrupt more than a constant fraction of the bits sent/received by any party.

A more challenging peer-to-peer channel where the adversarial noise can insert, delete, or alter communicated messages is considered in [30]. However, the obtained noise tolerance is $\Theta(\frac{1}{|E|\log|E|})$, which is even smaller than $\Theta(\frac{1}{n})$. Another line of work considers *oblivious* adversaries in peer-to-peer models [1, 29, 30]. Oblivious adversaries are not allowed to see the content of the communication channel when making their decision of what messages to corrupt. See more about that in Section 1.3.

Direct sum

The direct sum problem in communication complexity asks whether the communication required to solve k independent copies of a communication task is k times the communication required for solving a single copy. This problem has a rich history and was studied in several different settings (e.g., in the deterministic, non-deterministic, randomized, and distributional settings) and for different types of problems (relations, complete functions).

Currently, non-deterministic communication complexity is the best understood model in this regard, and an “almost perfect” direct sum theorem is known. The work of [22] and [40] showed that solving k copies of a relation R takes almost k times the amount of non-deterministic communication. More formally, $N(R^k) \geq k(N(R) - \log n - \mathcal{O}(1))$, where n is the number of bits required to describe an input for R , $N(R')$ denotes the non-deterministic communication complexity of R' , and R^k is the problem of solving k instances of R simultaneously. Note that if R represents a partial (or promise) problem, then solving R^k means giving the correct output on all the copies where the promise is satisfied.

For deterministic communication complexity, denoted D , and *total* functions f , [22] show a weaker direct sum theorem $D(f^k) \geq k(\sqrt{D(f)}/2 - \log n - \mathcal{O}(1))$.

The direct sum problem (and related problems like the direct product and XOR lemmas) were extensively studied in the randomized settings and are known to be related to other questions in complexity theory, like parallel repetition theorems and interactive compression schemes, [12, 39, 3, 34, 41, 6, 4, 8, 7, 37, 38, 26, 27, 49, to cite a few]. We mention that [26, 27] show that perfect direct sum does not hold for randomized communication complexity, however, weak direct sum theorems are known to hold, see, e.g., [4].

1.3 Additional Discussion and Future Directions

In this work we study the power and limitations of θ -limited adversaries in the broadcast model. We next discuss some of our modeling decisions and suggest other related questions.

Non-adaptive vs. adaptive protocols

In this work we follow the footsteps of El Gamal [25] and the followup works and assume that the order of communication in the protocol is *predetermined* and is independent of the players’ inputs and the channel noise (and therefore also independent of the parties’ received transcripts). Such protocols are called *non-adaptive*. Non-adaptive protocols are widely studied as they model certain common types of wireless networks, prevent *signaling*⁴, and can trivially ensure that exactly one party is broadcasting in every round.

⁴ Signaling is the situation in which information is inferred from whether a certain party has broadcast or not, rather than from the content of their communicated message.

Inspired by the radio network models in distributed computing [13], many recent works consider *adaptive* models, where a party decides whether to broadcast or not based on their input and their received transcript, see, e.g., [33, 10, 17, 11, 18, 19, 20, 21, 16]. As hinted above, such a model is prone to collision rounds (where more than one party broadcasts) and silent rounds (where no party broadcasts).

Note, however, that all the above mentioned adaptive models assume stochastic noise, and that it is *unclear how to adapt the definition of θ -limited adversaries to adaptive settings*. The main issue is that, for every i , our limited adversaries are only allowed to corrupt a θ -fraction of the total number of the messages received in rounds where party i broadcasts. But for adaptive models this number may not be fixed. Extending the notion of θ -limited adversaries to adaptive protocols is an intriguing question that may be motivated by the fact that, in various settings, adaptive protocols were shown to be much “stronger” than non-adaptive ones, see, e.g., [33, 17, 19, 21].

Interactive coding with limited adversaries

In this paper, we study the dissemination problem with θ -limited adversaries. As discussed in Section 1.2, such dissemination protocols imply a protocol for solving any other communication task with θ -limited adversaries, but the blow-up in communication may be substantial. In other words, interactive codes (with bad rate) are possible with θ -limited adversaries. It can likely be shown that, in some cases, such blow-up cannot be avoided⁵. An interesting goal is to find the “minimum additional restrictions” to be posed on the adversary that would allow for interactive coding with low overhead.

Randomness in adversarial models

Our simple dissemination protocol and our lower bound in Theorem 1, as well as most of the study of error correcting codes over adversarial channels in the literature, assume the *deterministic* setting. One can also consider *randomized* settings, where the parties share a random string. Note, however, that if this string is known to the adversary at the beginning of the protocol, then the protocol is essentially deterministic. On the other hand, if the random string is unknown to the adversary for the entire duration of the protocol, then the parties may use parts of it as one-time pads and ensure that the adversary is *oblivious* to the contents of the messages. Such adversaries are known to be weak (at least in the peer-to-peer setting) and every noiseless protocol can be simulated in the presence of such adversaries with only a constant overhead in the communication, see, e.g., [1, 29, 29].

An interesting direction for future work is to consider “*intermediate models*” where, for example, fresh randomness is sampled in every round, the adversary gets to see it immediately after it is sampled, but the adversary does not get to see randomness in future rounds when deciding on what corruptions to make. Can we design communication-efficient protocols in such models?

⁵ Consider, for example, the pointer chasing protocol on a tree of depth n , where party i has an edge coming out from each of the vertices in level i in the tree and the parties wish to find the unique root-to-leaf path contained in the union of their edges. While the parties can exchange their huge inputs and get constant resilience, an attempt to simulate the noiseless chasing protocol directly will result in noise resilience $o(1)$. Indeed, the adversary that erases the communication to the second party in the first θ -fraction of the rounds, then erases the third party for the next θ -fraction of the rounds, *etc.*, is θ -limited, but prevents the parties from computing the correct output. We mention that [36] show similar limitations in the peer-to-peer setting.

The adversarial erasure channel

In this work we have allowed the adversary to *corrupt*, or flip, some of the received bits. Can better protocols be designed for the easier setting where the adversary is only allowed to erase some of the received bits?

Noise tolerance

What is the maximum noise tolerance of dissemination protocols in our model? That is, what is the largest fraction of errors that can be handled by dissemination protocols? What is the rate vs. tolerance tradeoff?

2 Proof Overview

In this section, we give a detailed overview of our proof for Theorem 1. For the rest of this section we set $\theta = 0.01$. Let GapMaj_n be the following n -party problem: each of the n parties gets an input bit with the promise that at least $(1 - 2\theta)$ -fraction of the input bits agree. The goal is for all parties to output the majority bit. Let GapMaj_n^m be the problem where each of the n parties gets m input bits with the promise that the j -th bit of all the parties is an instance of GapMaj_n . In addition, it is promised that for every player i , there exists a set consisting of $(1 - 2\theta)$ -fraction of the copies j , such that the bit of party i for copy j is the majority bit of copy j (that is, $(1 - 2\theta)$ -fraction of the bits of each party are “correct”).

Our proof consists of two main parts. We first show that any protocol for information dissemination in our noisy model implies a protocol for GapMaj_n^m with the same communication cost. Here and for the rest of the section we set $m = \Theta(n)$. As explained in Section 1.1, this means that we can convert the noise in the channel to a type of noise in the inputs. We then prove an $\Omega(n^2)$ lower bound on the communication cost of GapMaj_n^m .

2.1 Reducing Noiseless GapMaj_n^m to Noisy Dissemination

The reduction for simple protocols

We first explain why “simple” information dissemination protocols, structured like the simple dissemination protocol we described in Section 1 (also see Section 4), imply a protocol for GapMaj_n^n with a similar number of rounds. Later in the section, we show how to extend the reduction to general dissemination protocols. Consider a protocol Π where all parties broadcast the same number of times. Additionally, assume that the protocol Π consists of two phases: in the first phase, which is, say, the first half of the rounds, the parties take turns broadcasting their input bits. Then, in the second phase, consisting of the second half of the rounds, the messages broadcast by the parties are only functions of their received transcript and are independent of their private input (*i.e.*, players “forget” their inputs).

To best see the connection to GapMaj_n^n , consider such a two-phase protocol Π in the following weak noise model. In this model, the adversary is θ -limited, and, in addition, it is only allowed to corrupt the messages received in the first phase, while the messages broadcast in the second phase are always received correctly. Furthermore, the only type of corruptions allowed in the first phase are as follows: for every two parties i and j , party i receives only 0s from party j or receives only 1s for party j . Since party j only broadcasts their input bit, this means that either party i receives all their transmissions correctly (j ’s input is $b \in \{0, 1\}$ and i received only b bits), or party i received all the transmissions flipped (party j ’s input is b and i received only \bar{b} bits). Clearly, lower bounds in this weaker noise model imply similar lower bounds for our noise model.

Next, observe that in the first phase of Π , for every party i , at least $(1 - 2\theta)$ -fraction of the parties received (many repetitions of) the correct input of i . This is because the adversary can only corrupt θ -fraction of the total outgoing messages of party i , since party i broadcasts in $\frac{1}{n}$ fraction of the rounds, and since the first phase is half of the total communication. Similarly, since the adversary can only corrupt a θ -fraction of the total incoming messages of a party, every party i receives the correct input of at least $(1 - 2\theta)$ -fraction of the parties.

Next, we claim that in the second, noiseless, phase, the parties are left with solving an instance of GapMaj_n^n in the absence of noise. In this instance, the input of party i for the j -th copy of GapMaj_n is the input that player i received from player j in the first phase. The promise in the definition of GapMaj_n^n is indeed satisfied: for every j , at least $(1 - 2\theta)$ -fraction of the parties i have the majority bit as their input for copy j , and for every i , for at least $(1 - 2\theta)$ -fraction of the j 's, party i 's input for copy j is the true majority bit of the j -th copy.

Removing the assumption of same number of broadcasts

So far we have considered “simple” protocols. We next show how to handle general protocols. First, we wish to remove the assumption that each party broadcasts in the same number of rounds. Note that this assumption is needed for the above argument. For example, if party i broadcasts in all the rounds of the second phase, then since the adversary is allowed to corrupt a θ -fraction of the *total* received communication for rounds where this party broadcasts, the adversary can corrupt all the messages received from this party in the first phase.

To rectify this situation, we “reveal” the input of all parties that broadcast in at least $\frac{3}{n}$ -fraction of the rounds (this is 3 times the average communication). Note that since we are working in the non-adaptive model, the number of times that each party broadcasts is determined ahead of time. By Markov’s inequality, at least $\frac{2n}{3}$ parties speak in less than $\frac{3}{n}$ -fraction of the rounds and are not fixed by being revealed. Therefore, we end up with a dissemination protocol that only needs to disseminate the input bits of $m \geq \frac{2n}{3}$ parties to all n parties. Note that since our reduction converts the dissemination of the input of one of the parties to one copy of GapMaj_n , applying the reduction to disseminate the value of m parties results in an instance of GapMaj_n^m (instead of GapMaj_n^n).

Removing the rest of the assumptions

The other assumptions we made when considering simple protocols, were that the protocol had two phases of equal lengths. In the first phase, parties only broadcast their inputs, and then, in the second phase, they “forget” their inputs, meaning that the messages broadcast by a party are independent of their input.

To handle general protocols Π , we use the following clever observation used by [32] to analyze protocols under stochastic noise: a message sent by player i in round t of Π given that their received transcript for the $t - 1$ first rounds of Π is π , can be deduced from the following three pieces of information:

1. the input bit of party i ,
2. the message that party i would have sent had their input been a 0 (and their received transcript was π), and
3. the message that party i would have sent had their input been a 1 (and their received transcript was π).

This gives a way of converting any protocol Π to a two-phase protocol Π' of our desired structure: for rounds $t = 1, 2, \dots$, if player i broadcasts in round t of Π , add two rounds to the first phase of Π' where player i broadcasts their input. Additionally, add two rounds to

the second phase of Π' where in the first, party i sends the message they would have sent had their input been a 0, and in the second they send the message they would have sent had their input been a 1.

2.2 Communication Lower Bound for GapMaj_n^m

Our next goal is to prove a deterministic communication complexity lower bound for GapMaj_n^m , as is promised by Theorem 2. One straightforward approach would be to prove a lower bound for one copy of GapMaj_n and then use one of the known direct sum theorems⁶. While it is not hard to prove a deterministic, or even a non-deterministic, communication lower bound for GapMaj_n , and while a perfect direct sum theorem is known for non-deterministic communication complexity [22, 40], this still does not give us the required bound. The reason is that, as explained after Theorem 2, such direct sum theorems only rule out “strong” communication protocols that solve all the copies that satisfy the promise, whereas we also need to rule out protocols that only output correctly when the promise is satisfied for all copies.

We also mention that the randomized communication complexity of GapMaj_n^m is low, at least when constant error probability is allowed. Consider the following protocol: each party broadcasts $t \cdot \frac{m}{n}$ random bits from their input, for some $t \leq n$. So, the expected number of bits communicated per copy is t , implying a total communication of tm and, using the Chernoff bound, the success probability of the protocol is at least $1 - 2^{-\frac{t}{10}}$. By taking $t = \frac{n}{10}$, we get a protocol with $\frac{mn}{10}$ communication and success probability $1 - 2^{-\frac{n}{100}}$.

We prove a randomized lower bound, showing that the latter tradeoff is optimal up to constant factors in the exponent. Specifically, we show that the success probability cannot be as high as $1 - 2^{-100n}$. Note that this is stronger than the deterministic lower bound we need for the proof of Theorem 1 to go through.

Our hard distribution(s)

To show our randomized lower bound, we prove a distributional lower bound and use Yao’s minimax theorem. Consider the following distribution \mathcal{D} on inputs for GapMaj_n^m : for every i and j , party i gets the bit 0 for copy j with probability $1 - \theta'$, independently, where $\theta' = \theta^{200}$.⁷ It may seem at first that our distribution is “easy”, as the right answer is the all-zeros vector, except with exponentially small probability. However, while exponentially small, the error probability of this protocol is still too large (the error probability is the probability that the promise is satisfied, but the correct answer is not the all-zeros vector).

To show that, observe what happens when we fix the first copy (say) to be 1 for all the players, which is an event whose probability is exponentially small. As the copies are mutually independent, the distribution of the other copies is not affected by this conditioning. Thus, for each one of the remaining copies, they satisfy the promise except with an exponentially small probability. Using a union bound, we get that conditioned on this event, all the copies satisfy the promise except with an exponentially small probability. This means that conditioned on the event that the first copies is fixed to be all-ones, it is likely that the the input is counted in the error probability, implying that the error probability of this protocol

⁶ Since most direct sum theorems are for the two-party setting, one would first need to adapt the theorem to the multi-party setting.

⁷ With an exponentially small probability, an instance sampled from this distribution does not satisfy the promise in the definition of GapMaj_n^m . If this is the case, we’ll accept any output by the protocol.

19:10 Information Dissemination via Broadcasts in the Presence of Adversarial Noise

is *at least* exponentially small (greater than our allowed error probability of 2^{-100n}). Also, observe that, as we union bounded over all the copies, the exact same argument can be made even if the copies are only *pairwise independent* instead of mutually independent.

In fact, the same arguments can be used to show that GapMaj_n^m cannot be solved by a 0-communication protocol over a large set of other distributions. This set contains the following distributions:

1. Distributions where the probability that party i gets the bit 1 for copy j is between $\frac{\theta'}{2}$ and $2\theta'$, and, as in \mathcal{D} , all input bits are independent.
2. Distributions where for every i , the input bits of party i are pairwise-independent (while the inputs of different parties continue to be independent).

GapMaj_n^m cannot be solved over such distributions with 0-communication protocols, because, as in the case of the distribution \mathcal{D} , the all-zeros vector is the correct solution except with exponentially small probability (the argument for this fact does not use independence, and therefore still holds). Meanwhile, our above argument for showing that the correct solution is not the all-zeros vector with probability greater than the allowed error probability only relied on pairwise independence, so it still holds.

3. Distributions where the bits for the same player may not be pairwise-independent, but the distribution of each pair of input bits of the same party are close in total variation distance to a distribution that is independent.

Lower bound over \mathcal{D}

The arguments above only showed that protocols with 0-communication will not solve GapMaj_n^m when the inputs are sampled from any distribution in the large set of distributions above. However, the lower bound we desire is for protocols with $o(n^2)$ communication. For this, our approach at a high level is to show that if the inputs of the parties are sampled from the distribution \mathcal{D} , then after $o(n^2)$ rounds of communication, the distribution of the inputs of the parties conditioned on the observed transcript (with high probability over the transcript), stays inside the set of distributions above. As distributions in the set are hard for 0-communication protocols, it follows that the original distribution is hard for $o(n^2)$ communication protocols.

Let \mathcal{D}' be the distribution \mathcal{D} conditioned on the observed transcript (for a typical transcript that we omit from the notation for the purposes of this sketch). Our goal is to show that \mathcal{D}' is in the set of distributions defined by Items 1–3. This requires showing that, for any player, the marginal distribution for any pair of copies (and also for any single copy) is close to the corresponding marginal in \mathcal{D} in total variation distance. As it turns out to be easier to handle, we actually measure the distance between these marginals in terms of the KL-divergence (*a.k.a.*, relative entropy) and move to total variation distance using Pinsker's inequality later in the proof.

Our goal therefore, is to show that for every party and every bit or pair of bits held by this party, the marginal distribution for this bit or pair of bits in \mathcal{D} and in \mathcal{D}' are close in terms of KL-divergence. Unfortunately, it is easily seen that this goal is impossible: consider the protocol where player 1 sends the bit in the first copy (and nothing happens after that). This simple protocol already violates Item 1 as now the marginal of the first party's bit of the first copy in \mathcal{D}' is a point mass. Getting around this impossibility is the next main part of the proof and we do it in two steps.

Revealing information and the use of pairwise independence

First, we show that the chain rule of KL-divergence and the fact that the protocol has $o(n^2)$ communication implies that the number of bits for which Item 1 fails is $o(n^2)$. For instance, the protocol mentioned above satisfies this, as sending any bit requires a bit of communication. With this bound, whenever a bit violates Item 1 (or a pair of bits violates Item 3), we “fix” the concerned bit(s) (*i.e.*, reveal it to the players for free). The knowledge of these bits changes the marginal distribution of the remaining bits, and in particular, may cause more of them to violate Item 1, causing us to fix even more bits. Nonetheless, as explained below, we are able to show using the chain rule for KL-divergence that this iterative procedure of fixing bits will terminate after $o(n^2)$ bits are fixed.

Indeed, as the protocol has $o(n^2)$ communication, we get that the KL-divergence between \mathcal{D} and \mathcal{D}' before any of the bits are fixed is $o(n^2)$. Because we only fix bits or pairs of bits whose marginal distributions have KL-divergence $\Omega(1)$, every time a bit or pair of bits is fixed, the chain rule for KL-divergence implies that the KL-divergence of the distributions \mathcal{D} and \mathcal{D}' , when restricted to the unfixed bits, goes down by at least $\Omega(1)$. As the initial KL-divergence is $o(n^2)$, this implies that the total number of fixed bits is $o(n^2)$.

The second step is to ensure that even after $o(n^2)$ bits are fixed, we still have the property that 0-communication protocols cannot compute GapMaj_n^m . To this end, let us carefully examine the argument above. The crux of the argument above was that the same transcript (which is the empty transcript for 0-communication protocols) is generated by two sets of inputs for which the output of gap majority is different. As the output is determined by the transcript, this means that the output of the protocol must be incorrect for at least one of the two sets, giving us the lower bound. Specifically, we observed that a typical input from the distribution \mathcal{D} satisfied the promise of GapMaj_n^m and resulted in the output being the all-zeros vector. Moreover, once we fix one of the copies to be one for all the parties, the remaining copies can still be fixed to satisfy the promise and have the majority value in the copies be zero.

We claim that, except with small probability, we can make this exact argument even after $o(n^2)$ of the bits have been fixed. This is because if the total number of bits fixed is $o(n^2)$, then most of the copies have only $o(n)$ fixed bits. Thus, regardless of the values these bits are fixed to, their number is small enough to not affect the output of GapMaj_n , which is determined by the other bits (with high probability). It remains to consider the copies that have $\Theta(n)$ fixed bits.

For these copies, it is possible that the values of the fixed bits prohibit the promise of gap majority from being satisfied, *e.g.*, when $\frac{n}{2}$ of the bits are fixed to 0 and $\frac{n}{2}$ of the bits are fixed to 1. However, as our distribution \mathcal{D} is heavily biased towards 0 and likely to remain this way for a typical transcript, this is an unlikely event and can be ignored. What cannot be ignored on the other hand is the case where almost all, say 0.999-fraction, of the fixed bits are fixed to 0, and the remaining 0.001-fraction are fixed to 1. Because the number of fixed bits is large, when this happens, we can no longer fix the remaining bits in the copy to satisfy the promise and have the majority value be 1, affecting the second of the two properties we desire.

To tackle this, we recall the fact that the number of copies for which this can happen is small, as most of the copies will have $o(n)$ fixed bits. In our proof, we track these copies with a large number of fixed entries and fix all their remaining bits. This makes sure that any copy with even one unfixed bit, can be fixed to satisfy the promise and have the majority value be 1, while also making sure that the total number of fixed bits stays $o(n^2)$, as we only increase the number of fixed bits by a constant factor. Note that, as before, fixing any of the bits changes the marginal distribution of the remaining bits, and this fixing must therefore be done iteratively until there are no more bits that need to be fixed.

3 Model and Preliminaries

3.1 Concentration Inequalities

► **Lemma 3** (Multiplicative Chernoff bound). *Suppose X_1, \dots, X_n are independent random variables taking values in $[0, 1]$. Let X denote their sum and let $\mu = \mathbb{E}[X]$ denote the sum's expected value. Then,*

$$\begin{aligned} \Pr(X \geq (1 + \delta)\mu) &\leq e^{-\frac{\delta^2 \mu}{2 + \delta}}, & \forall 0 \leq \delta, \\ \Pr(X \leq (1 - \delta)\mu) &\leq e^{-\frac{\delta^2 \mu}{2}}, & \forall 0 \leq \delta \leq 1. \end{aligned}$$

In particular, we have that:

$$\begin{aligned} \Pr(X \geq (1 + \delta)\mu) &\leq e^{-\frac{\delta \mu}{3} \cdot \min(\delta, 1)}, & \forall 0 \leq \delta, \\ \Pr(|X - \mu| \geq \delta \mu) &\leq 2 \cdot e^{-\frac{\delta^2 \mu}{3}}, & \forall 0 \leq \delta \leq 1. \end{aligned}$$

3.2 Error Correcting Codes

We use the following standard result about the existence of error correcting codes.

► **Lemma 4.** *Let $\delta > 0$ and define $K_0 = \lceil 10/\delta^2 \rceil$. For all $n > 0$, there exists a function $\text{ECC}_{n,\delta} : \{0, 1\}^n \rightarrow \{0, 1\}^{K_0 n}$ such that for all $s \neq t \in \{0, 1\}^n$, we have*

$$\Delta(\text{ECC}_{n,\delta}(s), \text{ECC}_{n,\delta}(t)) > \left(\frac{1}{2} - \delta\right) \cdot K_0 n.$$

3.3 The Adversarial Broadcast Channel

Protocols

Our communication model is the multi-party adversarial broadcast channel. Throughout this paper, we use n to denote the number of parties. An n -party protocol in this model is defined by a tuple:

$$\Pi = \left(\left\{ \mathcal{X}^{(i)} \right\}_{i \in [n]}, \mathcal{Y}, T, \sigma, \{M_j\}_{j \in [T]}, \text{out} \right),$$

where

1. for all $i \in [n]$, $\mathcal{X}^{(i)}$ is set of inputs of party i . We also define $\mathcal{X} = \mathcal{X}^{(1)} \times \dots \times \mathcal{X}^{(n)}$.
2. \mathcal{Y} is set of outputs of a protocol.
3. $T = \|\Pi\| \in \mathbb{N}$ is the length (number of rounds) in the protocol.
4. $\sigma \in [n]^T$ is a vector indicating for all rounds $j \in [T]$, which is the (unique) party scheduled to speak in round j .
5. for all $j \in [T]$, $M_j : \mathcal{X}^{\sigma_j} \times \{0, 1\}^{j-1} \times (\{0, 1\}^*)^j \rightarrow \{0, 1\}$ is the message function used by party σ_j that uses his input, the bits he received in the first $j - 1$ rounds, and the randomness sampled in the first j rounds to output the bit he will broadcast in round j .
6. $\text{out} : \{0, 1\}^T \rightarrow \mathcal{Y}$ is the function that the parties use to compute the output of the protocol based on the bits they receive.

We will omit $\mathcal{X}^{(i)}$ and \mathcal{Y} when they are clear from context.

Adversaries

Let Π be a protocol as above. An adversary for Π is defined by a tuple $\text{Adv} = (\text{Adv}_{i,j})_{i \in [n], j \in [T]}$, where $\text{Adv}_{i,j} : \mathcal{X} \times (\{0, 1\}^*)^j \rightarrow \{0, 1\}$. Here, for all $i \in [n]$ and $j \in [T]$, the function $\text{Adv}_{i,j}$ takes as input the inputs of all the parties and the randomness sampled in the first j rounds and outputs 1 if he wants to flip (corrupt) the bit party i receives in round j , and outputs 0 otherwise. Our formulation thus, captures adversaries that have knowledge of all the parties' inputs and the randomness they sampled so far, but are unaware of the randomness they will sample in the future.

Protocol execution

We next describe the execution of a protocol Π in the presence of adversary Adv : Each party $i \in [n]$ starts with an input $x^{(i)} \in \mathcal{X}^{(i)}$. Let $x = (x^{(1)}, \dots, x^{(n)})$. The execution takes place in T rounds, maintaining the invariant that for all parties $i \in [n]$ and all rounds $j \in [T]$, party i has a transcript $\pi_{<j}^{(i)}$ before the execution of round j . In each round $j \in [T]$, the parties first sample a shared random string $r_j \in \{0, 1\}^*$. The player σ_j then computes $\pi_j = M_j(x^{(\sigma_j)}, \pi_{<j}^{(\sigma_j)}, r_{\leq j})$ and broadcasts it over the channel. All parties $i \in [n]$ then receive a (potentially corrupted) bit $\pi_j^{(i)} = \pi_j \oplus \text{Adv}_{i,j}(x, r_{\leq j})$ and append it to $\pi_{<j}^{(i)}$ to get $\pi_{\leq j}^{(i)}$. At the end of the protocol, all parties $i \in [n]$ output $\text{out}(\pi_{\leq T}^{(i)})$. We say that an execution is *noiseless* if $\text{Adv}_{i,j}$ always outputs 0, and we call this adversary the noiseless adversary.

Additional discussion of the model

We finish this section with a few remarks about the above definition: Note that when Π is executed in the presence of Adv , the output of any party $i \in [n]$ is determined by the parties' inputs $x = (x^{(1)}, \dots, x^{(n)})$ and the sampled randomness $r_{\leq T}$. Owing to this, we denote it using the notation $\Pi_{\text{Adv}, i}(x, r_{\leq T})$. We will omit writing Adv, i when the adversary is noiseless as in this case, all players compute the same transcripts, and therefore, also the same output. We also omit $r_{\leq T}$ from our notation when talking about deterministic protocols. Also, as mentioned above, we define our adversaries to have complete knowledge of the inputs of the parties and the randomness they sampled so far but they are not aware of any future randomness the parties might have. Due to their knowledge of the randomness sampled so far (and the inputs), the adversaries can also compute all the bits sent and received over the channel so far, and we do not explicitly include this in our notation. Moreover, as our upper bound is deterministic, we will only need this assumption in our lower bound, and it only makes our result stronger.

Next, note that, as defined, the output function out is the same for all parties and depends only on the transcript of the protocol and not on the inputs of the parties. This is without loss of generality, as we can always extend the protocol so that one of the parties computes and sends the output over the channel (using an error correcting code) and all the parties then decode it from the transcript. Finally, note that our definition allows us to easily measure the number of bits corrupted by the adversary. As we are interested only in adversaries that do not corrupt too many bits sent or received by any given party, we define, for all $i \in [n]$, the set $\sigma^{-1}(i) = \{j \in [T] : \sigma(j) = i\}$ to contain rounds where party i broadcast and also define:

► **Definition 5.** Let Π be a protocol and Adv be an adversary for Π . Let $\theta > 0$. We say that Adv is θ -limited, if for all $x = (x^{(1)}, \dots, x^{(n)})$, all $r \leq T$, and all $i \in [n]$, we have:

$$\sum_{j \in [T]} \text{Adv}_{i,j}(x, r \leq j) \leq \theta T,$$

$$\sum_{j \in \sigma^{-1}(i)} \sum_{i' \in [n]} \text{Adv}_{i',j}(x, r \leq j) \leq \theta n \cdot |\sigma^{-1}(i)|.$$

Computation over the model

Let Π be a protocol as above and $f : \mathcal{X} \rightarrow \mathcal{Y}$ be a (possibly partial) function. Let $\theta, p > 0$. We say that the protocol Π computes f with probability p resilient to θ -adversarial noise if for all inputs $x = (x^{(1)}, \dots, x^{(n)})$ in the domain of f and all θ -limited adversaries, we have:

$$\Pr(\forall i \in [n] : \Pi_{\text{Adv},i}(x, r \leq T) = f(x)) \geq p. \quad (1)$$

We omit writing “resilient to” when $\theta = 0$. As there is only one adversary that is 0-limited, we will also omit Adv from the subscript in this case.

The ID and GapMaj problems

We consider the n -party *Information Dissemination* function, denoted ID_n , that simply outputs its n -sized tuple of arguments. That is, $\text{ID}_n : \{0, 1\}^n \rightarrow \{0, 1\}^n$, where $\text{ID}_n(x_1, \dots, x_n) = (x_1, \dots, x_n)$.

We also define the partial function $\text{GapMaj}_{\epsilon,n}^{\delta,m}$ that is parametrized by numbers $\epsilon, \delta > 0$ and an integer m and is such that $\mathcal{X}^{(i)} = \mathcal{Y} = \{0, 1\}^m$ for all $i \in [n]$. $\text{GapMaj}_{\epsilon,n}^{\delta,m}$ is defined only if there exists a $\hat{x} \in \{0, 1\}^m$ such that

1. for all $i \in [n]$, the Hamming distance between \hat{x} and x_i (the input vector for party i) is at most δm ,
 2. for all $j \in [m]$, we have $\hat{x}_j \neq x_j^{(i)}$ for at most ϵn values of $i \in [n]$,
- and outputs the (unique, for small ϵ) vector \hat{x} . For notational convenience, we will interpret $\text{GapMaj}_{\epsilon,n}^{\delta,m}$ as outputting a set of possible values, where the set is the singleton set $\{\hat{x}\}$ if the conditions above are satisfied, and is $\{0, 1\}^m$ otherwise.

3.4 Our Result

We are now ready to state the formal version of Theorem 1.

► **Theorem 6.** For all $\theta > 0$, there exists $\kappa > 0$ such that for all integers n large enough, any protocol Π that computes ID_n with probability 1 resilient to θ -adversarial noise has length $\|\Pi\| > \kappa n^2$.

We will actually prove the following slightly stronger Theorem 7 that implies Theorem 6:

► **Theorem 7.** For all $\theta > 0$, there exists $\kappa > 0$ such that for all integers n large enough, any protocol Π that computes ID_n with probability $1 - \kappa^n$ resilient to θ -adversarial noise has length $\|\Pi\| > \kappa n^2$.

The proof of Theorem 7 has two main parts. The first part is a reduction showing that lower bounds for protocol computing Information Dissemination can be obtained from lower bounds from protocols computing Gap Majority. This is described in Section 5. The next part is a lower bound for protocols computing Gap Majority, which is written in Section 6.

4 Our Information Dissemination Protocol

In order to demonstrate that our result in Theorem 6 is tight, we provide a simple algorithm with length $\mathcal{O}(n^2)$, which is resilient to θ -adversarial noise, for all $\theta < 1/40$.

High level description

To summarize, the idea is to proceed in two phases. In the first phase, every player says their input bit $\mathcal{O}(n)$ times. Each other player then takes the majority value of what they heard. Thus, each player now has a guess for each player's input. Then, in the second phase, every player encodes the string of guesses they have using an error-correcting code, and broadcasts the result. Each player then takes all the error-correcting codes they've received, decodes them, and sets their guess for each player's input to be the majority value among the guesses they've decoded.

The reason this works is that in order to corrupt a player's output, the adversary needs to either corrupt a lot of the error-correcting codes received by that player, or they need to corrupt a lot of players' guesses about some specific party's input. In either case, the adversary ends up corrupting too many rounds of communication, thus showing that a θ -limited adversary cannot possibly corrupt even a single player's output, exactly as desired.

The formal protocol

Our protocol is given in Algorithm 1. It uses an error correcting code $\text{ECC}_{n,\delta}$ as promised by Lemma 4, where $\delta = 1/10$. We use K_0 as given in that lemma.

■ **Algorithm 1** The algorithm computing ID_n .

Input: Each party $k \in [n]$ has an input $x_k \in \{0, 1\}$.

Output: Each party $i \in [n]$ outputs a $\hat{x}_{i,1}, \dots, \hat{x}_{i,n}$, such that $\hat{x}_{i,k} = x_k$ for all $k \in [n]$.

- 1: **for** $k \in [n]$ **do**
 - 2: Party k broadcasts x_k $K_0 n$ times.
 - 3: Each party $j \in [n]$ sets $y_{j,k}$ equal to the majority value they received in the previous $K_0 n$ broadcasts.
 - 4: **end for**
 - 5: **for** $j \in [n]$ **do**
 - 6: Party j computes and broadcasts $\text{ECC}_{n,\delta}(y_{j,1}, \dots, y_{j,n})$. This takes $K_0 n$ broadcasts.
 - 7: Each party $i \in [n]$ sets $\hat{y}_{i,j,1}, \dots, \hat{y}_{i,j,n}$ to the minimize the distance of $\text{ECC}_{n,\delta}(\hat{y}_{i,j,1}, \dots, \hat{y}_{i,j,n})$ to the messages they received in the previous $K_0 n$ broadcasts.
 - 8: **end for**
 - 9: Each party $i \in [n]$ sets $\hat{x}_{i,k}$ to be the majority value between $\hat{y}_{i,1,k}, \dots, \hat{y}_{i,n,k}$ for all $k \in [n]$.
 - 10: Each party $i \in [n]$ outputs $\hat{x}_{i,1}, \dots, \hat{x}_{i,n}$.
-

► **Theorem 8.** *For all $\theta < 1/40$, the protocol in Algorithm 1 solves ID_n resilient to θ -adversarial noise.*

19:16 Information Dissemination via Broadcasts in the Presence of Adversarial Noise

Proof. We begin by noting that this protocol is deterministic. As such, the behavior of the protocol is completely determined by the inputs to the players and the adversary Adv , and that, for simplicity, adversaries can be assumed to just be a function of the inputs x_1, \dots, x_n .

Fix some adversary Adv . We wish to demonstrate that if there exists some input x_1, \dots, x_n such that executing the protocol in Algorithm 1 against Adv on inputs x_1, \dots, x_n results in a player outputting an incorrect output, then the adversary Adv is not θ -limited.

To see this, suppose that a player outputs some incorrect value. In particular, suppose that $\hat{x}_{i,k} \neq x_k$ for some $i \in [n]$ and $k \in [n]$. That means that the majority value among $\hat{y}_{i,1,k}, \dots, \hat{y}_{i,n,k}$ was not x_k .

That implies that there must exist some set $S \subseteq [n]$ such that $|S| \geq n/4$, and that one of the two following conditions must hold true:

1. For all $j \in S$, $\hat{y}_{i,j,k} \neq y_{j,k}$, or
2. for all $j \in S$, $y_{j,k} \neq x_k$.

We claim that in order for either of these cases to occur, the adversary Adv must not be θ -limited. First suppose that for all $j \in S$, $\hat{y}_{i,j,k} \neq y_{j,k}$. That implies that for each $j \in S$, during iteration j of the loop at Line 5, at least $0.2K_0n$ of the bits sent by player j are received incorrectly by player i , by the properties promised about $\text{ECC}_{n,\delta}$ in Lemma 4. That implies that over all the iterations of the loop at Line 5, player i hears at least $\frac{0.2}{4}K_0n^2 = \frac{1}{20}K_0n^2$ messages incorrectly. That means that at least $1/40$ of all $2K_0n^2$ messages sent during the protocol are misheard by player i . This, thus, shows that Adv is not θ -limited.

On the other hand, suppose that for all $j \in S$, $y_{j,k} \neq x_k$. That implies that for all $j \in S$, during iteration k of the loop at Line 1, at least $0.5K_0n$ of the bits sent by player k are received incorrectly by player j . That implies that there are at least $\frac{0.5}{4}K_0n^2 = 1/8K_0n^2$ corruptions in messages from player k to other players, of a total of $2K_0n^2$ messages received in rounds during which this player broadcasts. This, thus, shows that Adv is not θ -limited. \blacktriangleleft

5 Reducing Gap Majority to Information Dissemination

This section has the first part of our proof, which is a reduction from Gap Majority to Information Dissemination. Specifically, our reduction shows that noise resilient protocols for Information Dissemination imply noiseless protocols for Gap Majority, as formalized next:

► **Theorem 9.** *Let parameters $0 < \theta < \frac{1}{3}$, $0 < p < 1$ and $n \in \mathbb{N}$ be given. If there exists a protocol Π computing ID_n with probability p resilient to θ -adversarial noise, there exists another protocol Π' computing $\text{GapMaj}_{\theta,n}^{\theta,\theta n}$ with probability p such that $\|\Pi'\| \leq 2 \cdot \|\Pi\| + n$.*

Proof. Fix θ , p , n , and Π as in the theorem statement. Let $\Pi = (T, \sigma, \{M_j\}_{j \in [T]}, \text{out})$. Define the set $I' = \{i \in [n] \mid |\sigma^{-1}(i)| \leq \frac{T}{\theta n}\}$ to be the set of parties that do not broadcast too often. By Markov's inequality, note that $|I'| \geq (1 - \theta)n > \theta n$. We let $m = \theta n$, I be the first m elements of I' and assume without loss of generality that $I = [m]$. We are now ready to define the protocol Π' . We note that throughout the description of Π' and its analysis, we will treat vectors in $\{0, 1\}^m$ also as vectors in $\{0, 1\}^n$ by padding with an appropriate number of zeros.

■ **Algorithm 2** The algorithm Π' computing $\text{GapMaj}_{\theta,n}^{\theta,m}$.

Input: Party $i \in [n]$ has an input $x^{(i)} \in \{0,1\}^m$.

- 1: Each party $i \in [n]$ sets $\pi^{(i)} \leftarrow \varepsilon$, the empty string.
- 2: **for** $j \in [T]$ **do**
- 3: The parties together sample a random string $r_j \in \{0,1\}^*$.
- 4: Party σ_j sets $\tau_{j,b} \leftarrow M_j(b, \pi^{(\sigma_j)}, r_{\leq j})$ for all $b \in \{0,1\}$. $\triangleright |\pi^{(\sigma_j)}| = j - 1$.
- 5: Party σ_j broadcasts $\tau_{j,b}$ for all $b \in \{0,1\}$ to all other players.
- 6: Each party $i \in [n]$ extends $\pi^{(i)}$ by appending $\tau_{j,x_{\sigma_j}^{(i)}}$.
- 7: **end for**
- 8: Each party $i \in [n]$ outputs the first m bits of $\text{out}(\pi^{(i)})$.

Observe that, as written, the output function of the protocol Π' depends on the inputs of the parties. However, as mentioned in Section 3, this can be easily corrected by adding an extra n rounds where one of the parties broadcasts its output over the channel. Together with these n rounds, the $2T$ rounds in Line 5 imply that $\|\Pi'\| \leq 2 \cdot \|\Pi\| + n$. It remains to show that Π' computes $\text{GapMaj}_{\theta,n}^{\theta,m}$ with probability p . For this, we have to show Equation (1). We do this next.

Fix $x' = (x^{(1)}, \dots, x^{(n)})$ in the domain of $\text{GapMaj}_{\theta,n}^{\theta,m}$ as in Equation (1) and define $\hat{x} = \text{GapMaj}_{\theta,n}^{\theta,m}(x')$. As $\text{ID}_n(\hat{x}) = \hat{x}$ by definition, Equation (1) follows if we show a θ -limited adversary Adv for Π such that for all $r_{\leq T}$ and all $i \in [n]$, we have that:

$$\Pi_{\text{Adv},i}(\hat{x}, r_{\leq T}) = \Pi'_i(x', r_{\leq T}),$$

where, as usual, we pad the output of Π' with zeros to be of length n . Indeed, the above implies $\Pi_{\text{Adv},i}(\hat{x}, r_{\leq T}) = \hat{x} \iff \Pi'_i(x', r_{\leq T}) = \hat{x}$, and Equation (1) follows from the fact that Π computes ID_n with probability p resilient to θ -adversarial noise.

To start, we first note that it suffices to define a different Adv for every randomness $r_{\leq T}$ as the property we want is determined solely by the value of Adv on the randomness $r_{\leq T}$. Fix an arbitrary $r_{\leq T}$ and note that, as we already fixed x' , fixing $r_{\leq T}$ fixes the value of all variables in the execution of Algorithm 2. Henceforth, we abuse notation and use the name of the variable to also denote its fixed value at the end of the protocol. We define the adversary Adv as:

$$\text{Adv}_{i,j}(\hat{x}, r_{\leq j}) = \pi_j^{(i)} \oplus \tau_{j,\hat{x}_{\sigma_j}},$$

and we set it to 0 everywhere else. We now show why this adversary satisfies $\Pi_{\text{Adv},i}(\hat{x}, r_{\leq T}) = \Pi'_i(x', r_{\leq T})$ for all $i \in [n]$. Due to Line 8, this follows if we show that for all i , the transcript $\pi^{(i)}$ equals the transcript $\pi^{(i)}$ received by party i when Π is executed in the presence of Adv . As both $\pi^{(i)}$ and $\pi^{(i)}$ have length T , this follows if we show by induction that, for all $0 \leq j \leq T$, we have $\pi_{\leq j}^{(i)} = \pi_{\leq j}^{(i)}$. The base case $j = 0$ is straightforward. To prove the statement for $j > 0$, we assume it holds for $j - 1$ and prove that $\pi_j^{(i)} = \pi_j^{(i)}$. We have:

$$\begin{aligned} \pi_j^{(i)} &= M_j(\hat{x}_{\sigma_j}, \pi_{<j}^{(\sigma_j)}, r_{\leq j}) \oplus \text{Adv}_{i,j}(\hat{x}, r_{\leq j}) \\ &= M_j(\hat{x}_{\sigma_j}, \pi_{<j}^{(\sigma_j)}, r_{\leq j}) \oplus \pi_j^{(i)} \oplus \tau_{j,\hat{x}_{\sigma_j}} && \text{(Definition of Adv)} \\ &= M_j(\hat{x}_{\sigma_j}, \pi_{<j}^{(\sigma_j)}, r_{\leq j}) \oplus \pi_j^{(i)} \oplus \tau_{j,\hat{x}_{\sigma_j}} && \text{(Induction hypothesis)} \\ &= \pi_j^{(i)}. && \text{(Line 4)} \end{aligned}$$

19:18 Information Dissemination via Broadcasts in the Presence of Adversarial Noise

It remains to show that Adv is θ -limited. For this we show the two inequalities in Definition 5. As Adv is 0 everywhere else, it suffices to show it for the arguments $(\hat{x}, r_{\leq T})$. For the first inequality, we have for all $i \in [n]$ that:

$$\begin{aligned}
 \sum_{j \in [T]} \text{Adv}_{i,j}(\hat{x}, r_{\leq j}) &\leq \sum_{j \in [T]} \mathbb{1}(x_{\sigma_j}^{(i)} \neq \hat{x}_{\sigma_j}) && \text{(Definition of Adv and Line 6)} \\
 &= \sum_{i'=1}^n \mathbb{1}(x_{i'}^{(i)} \neq \hat{x}_{i'}) \cdot |\sigma^{-1}(i')| \\
 &= \sum_{i'=1}^m \mathbb{1}(x_{i'}^{(i)} \neq \hat{x}_{i'}) \cdot |\sigma^{-1}(i')| && \text{(The other coordinates are paddings)} \\
 &\leq \sum_{i'=1}^m \mathbb{1}(x_{i'}^{(i)} \neq \hat{x}_{i'}) \cdot \frac{T}{\theta n} && \text{(Definition of } I) \\
 &\leq \theta T. && \text{(Definition of } m \text{ and } \text{GapMaj}_{\theta,n}^{\theta,m})
 \end{aligned}$$

For the second inequality, we have for all $i \in [n]$ that:

$$\begin{aligned}
 \sum_{j \in \sigma^{-1}(i)} \sum_{i' \in [n]} \text{Adv}_{i',j}(\hat{x}, r_{\leq j}) &\leq \sum_{j \in \sigma^{-1}(i)} \sum_{i' \in [n]} \mathbb{1}(x_{\sigma_j}^{(i')} \neq \hat{x}_{\sigma_j}) && \text{(Definition of Adv and Line 6)} \\
 &\leq \sum_{j \in \sigma^{-1}(i)} \theta n && \text{(Definition of } \text{GapMaj}_{\theta,n}^{\theta,m}) \\
 &\leq \theta n \cdot |\sigma^{-1}(i)|. && \blacktriangleleft
 \end{aligned}$$

6 Lower Bound for Direct Sum Gap-Majority

The goal of this section is to show Theorem 7. As we already proved Theorem 9, it suffices to show the following result.

► **Theorem 10.** *For all $0 < \theta < \frac{1}{3}$, there exists $\kappa > 0$ such that for all $n > 0$ large enough and $m = \theta n$, any (possibly randomized) protocol Π computing $\text{GapMaj}_{\theta,n}^{\theta,\theta n}$ with probability $1 - \kappa^n$ satisfies $\|\Pi\| \geq \kappa n^2$.*

Indeed, Theorem 7 follows easily from Theorems 9 and 10. Moreover, as $\text{GapMaj}_{\theta,n}^{\theta,\theta n}$ is an easier problem than $\text{GapMaj}_{\theta,n}^{1,\theta n}$, Theorem 10 implies the following result about the direct-sum of gap-majority, that may be of independent interest.

► **Theorem 11.** *For all $0 < \theta < \frac{1}{3}$, there exists $\kappa > 0$ such that for all $n > 0$ large enough and $m = \theta n$, any (possibly randomized) protocol Π computing $\text{GapMaj}_{\theta,n}^{1,\theta n}$ with probability $1 - \kappa^n$ satisfies $\|\Pi\| \geq \kappa n^2$.*

Henceforth, we focus on proving Theorem 10, whose proof spans this entire section. Fix θ as in the theorem statement and define $\kappa = \theta^{1000}$. Let $n > 0$ be sufficiently large and define a distribution \mathcal{D} over inputs for $\text{GapMaj}_{\theta,n}^{\theta,m}$ as follows: For all players $i \in [n]$ and all $j \in [m]$, the bit $x_{i,j}$ is sampled independently of all other bits and is 1 with probability θ^{25} and 0 with probability $1 - \theta^{25}$. We will show that, any deterministic protocol Π with $\|\Pi\| < \kappa n^2$ satisfies:

$$\Pr_{X \sim \mathcal{D}} \left(\Pi(X) \in \text{GapMaj}_{\theta,n}^{\theta,m}(X) \right) \leq 1 - \kappa^n.$$

Theorem 10 then follows from Yao’s minimax principle. For brevity sake, we henceforth keep the distribution \mathcal{D} implicit. To show this bound, we shall focus on a testing version of Gap Majority, where the parties are only required to determine whether or not the output is the all zeros vector 0^m . Specifically, let $\text{flag} : \{0, 1\}^m \rightarrow \{0, 1\}$ be the indicator function that outputs 0 if and only if the Hamming weight of its input is at most $\frac{\theta m}{2}$ and 1 otherwise. Next, define the set-valued function $\text{GapMaj}_{\theta, n}^m$ as follows:

$$\text{GM-Test}_{\theta, n}^m(x) = \begin{cases} \{0, 1\}^m, & \text{if } \exists i \in [n] : \text{flag}(x_i) = 1 \\ \{0^m\}, & \text{else if } \text{GapMaj}_{\theta, n}^{\theta, m}(x) = \{0^m\} \\ \{0, 1\}^m \setminus \{0^m\}, & \text{else if } |\text{GapMaj}_{\theta, n}^{\theta, m}(x)| = 1 \\ \{0, 1\}^m, & \text{otherwise} \end{cases}. \quad (2)$$

This definition implies that $\text{GapMaj}_{\theta, n}^{\theta, m}(x) \subseteq \text{GM-Test}_{\theta, n}^m(x)$ for all x and thus, it suffices to show that any deterministic protocol Π with $\|\Pi\| < \kappa n^2$.

$$\Pr(\Pi(X) \in \text{GM-Test}_{\theta, n}^m(X)) \leq 1 - \kappa^n. \quad (3)$$

Fix a protocol Π as above and let $T = \kappa n^2$ so that $\|\Pi\| < T$ and $T' = T/\theta^{500}$. We first augment Π to get another protocol Π_{aug} that reveals some extra information about the parties’ inputs. The protocol Π_{aug} is defined below in Algorithm 3 where we use the symbol \perp to denote a special symbol saying “I skip”. Also, for a distribution D on the parties’ inputs and a subset $S \subseteq [n] \times [m]$, we use $D|_S$ to denote the marginal distribution of D over the coordinates in S . If we are writing a set, say $S = \{(i_1, j_1), (i_2, j_2)\}$, explicitly, we may omit the $\{\}$ and simply write $D|_{(i_1, j_1), (i_2, j_2)}$.

■ **Algorithm 3** The protocol Π_{aug} . All lines except Lines 8 and 11 executed by all the players. Any message sent is automatically appended to π_{aug} .

Input: Player i ’s input is a vector $x_i \in \{0, 1\}^m$.

- 1: Run Π to get a transcript $\pi \in \{0, 1\}^T$. Set $\pi_{\text{aug}} \leftarrow \pi$.
- 2: All players $i \in [n]$ speak. Player i sends $\text{flag}(x_i)$.
- 3: For all $i \in [n]$, $j \in [m]$, we set $R_{i, j} \leftarrow 0$.
- 4: **for** $t \in [T']$ **do**
- 5: Compute the sets:

$$S_{\text{cell}} = \{(i, j) \mid R_{i, j} = 0 \wedge \mathbb{D}((\mathcal{D} \mid \pi_{\text{aug}})|_{(i, j)} \parallel \mathcal{D}|_{(i, j)}) \geq \theta^{200}\},$$

$$S_{\text{pair}} = \{(i, j, j') \mid j \neq j' \wedge R_{i, j} = R_{i, j'} = 0 \wedge \mathbb{D}((\mathcal{D} \mid \pi_{\text{aug}})|_{(i, j), (i, j')} \parallel \mathcal{D}|_{(i, j), (i, j')}) \geq \theta^{100}\},$$

$$S_{\text{col}} = \{(i, j) \mid R_{i, j} = 0 \wedge |\{i' \in [n] \mid R_{i', j} = 1\}| \geq \theta^{100} \cdot n\}.$$
- 6: **if** $S_{\text{cell}} \cup S_{\text{col}} \neq \emptyset$ **then**
- 7: Let (i_1, j_1) be the smallest element in $S_{\text{cell}} \cup S_{\text{col}}$. Set $R_{i_1, j_1} \leftarrow 1$.
- 8: All players $i \in [n]$ speak. If $i \neq i_1$, they send (\perp, \perp) . Else, they send (x_{i_1, j_1}, \perp) .
- 9: **else if** $S_{\text{pair}} \neq \emptyset$ **then**
- 10: Let (i_2, j_2, j'_2) be the smallest element in S_{pair} . Set $R_{i_2, j_2}, R_{i_2, j'_2} \leftarrow 1$.
- 11: All players $i \in [n]$ speak. If $i \neq i_2$, they send (\perp, \perp) . Else, they send $(x_{i_2, j_2}, x_{i_2, j'_2})$.
- 12: **else**
- 13: All players send (\perp, \perp) .
- 14: **end if**
- 15: **end for**

Intuitively, the protocol Π_{aug} “cleans” the protocol Π by revealing the functions $\text{flag}(\cdot)$ and then, iteratively revealing all coordinates, pairs of coordinates, *etc.* for which the marginal distribution changed significantly. We describe this formally in the following section, but before that, a word on the notations we use.

Throughout this proof, we will use sans-serif letters, e.g., X to denote random variables and the corresponding lower case letters, e.g., x to denote their values. When it is clear from context, we may abbreviate the event $X = x$ as just x . Note that the protocol Π_{aug} is deterministic and the only randomness we have is the randomness of the distribution \mathcal{D} of inputs. All our random variables and probabilities are defined over this randomness.

For a variable var in Algorithm 3 and $t \in [T']$, we use var^t to denote the random variable (over the randomness of the inputs in \mathcal{D}) whose value equals the value of the variable at the end of iteration t of the loop in Line 4. When $t = 0$, we mean the corresponding value at the beginning of the loop, *i.e.*, after Line 3. We may omit writing the subscript when $t = T'$. We also define the additional set-valued variable $R = \{(i, j) \mid R_{i,j} = 1\}$ and use the same notation. Note that the set R can only grow.

6.1 Properties of Π_{aug}

In this subsection, we establish some useful properties of Π_{aug} .

► **Lemma 12.** *For all $0 \leq t \leq T'$, the value of π_{aug}^{t-1} determines⁸ the values of R^0, R^1, \dots, R^t .*

Proof. Proof by induction on t . The base case $t = 0$ is because $R^0 = \emptyset$ by definition. We prove the lemma for $t > 0$ assuming it holds for $t - 1$. Consider iteration t for the loop in Line 4 and let π_{aug}^{t-1} be an arbitrary value of the variable π_{aug} at the beginning of the iteration. As π_{aug}^{t-1} determines π_{aug}^{t-2} , we have by the induction hypothesis that it also determines the values of R^0, R^1, \dots, R^{t-1} . In particular, it determines R^{t-1} , the value of R at the beginning of this execution. Therefore, it also determines the value of the sets computed in Line 5. Now, using Lines 6, 8, 9, and 11, we get that it also determines the value of R^t , as desired. ◀

Observe from Algorithm 3 that that the variables $S_{\text{cell}}^t, S_{\text{pair}}^t, S_{\text{col}}^t, (i_1^t, j_1^t), (i_2^t, j_2^t, j_2^t)$ are all determined by π_{aug}^{t-1} and R^{t-1} . Thus, we get:

► **Corollary 13.** *For all $0 \leq t \leq T'$, the value of π_{aug}^{t-1} determines the values of $S_{\text{cell}}^t, S_{\text{pair}}^t, S_{\text{col}}^t, (i_1^t, j_1^t), (i_2^t, j_2^t, j_2^t)$.*

► **Lemma 14.** *For all $0 \leq t \leq T'$, the value of π_{aug}^t determines the values of $x_{i,j}$ for all $(i, j) \in R^t$.*

Proof. Proof by induction on t . The base case $t = 0$ is because $R^0 = \emptyset$ by definition. We prove the lemma for $t > 0$ assuming it holds for $t - 1$. Consider iteration t for the loop in Line 4. As π_{aug}^t determines π_{aug}^{t-1} , we have by the induction hypothesis that π_{aug}^t determines the values of $x_{i,j}$ for all $(i, j) \in R^{t-1}$. Moreover, from Lines 8 and 11, we have that, for all $(i, j) \in R^t \setminus R^{t-1}$, the value of $x_{i,j}$ is determined by π_{aug}^t . ◀

► **Lemma 15.** *For all $1 < t \leq T'$, if $S_{\text{cell}}^{t-1} = S_{\text{pair}}^{t-1} = S_{\text{col}}^{t-1} = \emptyset$, then $S_{\text{cell}}^t = S_{\text{pair}}^t = S_{\text{col}}^t = \emptyset$ (with probability 1).*

⁸ We define π_{aug}^{t-1} to be some dummy value when $t = 0$.

Proof. Recall from Corollary 13 that, for all $1 < t \leq T'$, the value of π_{aug}^{t-2} determines the values of S_{cell}^{t-1} , S_{pair}^{t-1} , S_{col}^{t-1} . Fix an arbitrary $1 < t \leq T'$ and an arbitrary π_{aug}^{t-2} such that $S_{\text{cell}}^{t-1} = S_{\text{pair}}^{t-1} = S_{\text{col}}^{t-1} = \emptyset$ and consider iteration $t-1$ of the loop in Algorithm 3. As Lines 8 and 11 are never executed in this iteration we get that $R^{t-2} = R^{t-1}$ and that π_{aug}^{t-2} determines π_{aug}^{t-1} . This means that both π_{aug}^{t-2} and π_{aug}^{t-1} determine each other implying that $\mathcal{D} \mid \pi_{\text{aug}}^{t-2} = \mathcal{D} \mid \pi_{\text{aug}}^{t-1}$. Combine this with $R^{t-2} = R^{t-1}$ and use Line 5 to finish the proof. ◀

► **Lemma 16.** For all $0 \leq j \leq \lfloor \pi_{\text{aug}}^{T'} \rfloor$, the random variables X_1, \dots, X_n are mutually independent conditioned on $\pi_{\text{aug}, \leq j}^{T'}$.

Proof. Proof by induction on j . The base case $j = 0$ is trivial. We prove the lemma for $j > 0$ assuming it holds for $j-1$. By the induction hypothesis, we have that X_1, \dots, X_n are mutually independent conditioned on $\pi_{\text{aug}, < j}^{T'}$. This means that for all $i \in [n]$, and all functions f and all values z in the range of f , we have that X_1, \dots, X_n are mutually independent conditioned on $\pi_{\text{aug}, < j}^{T'}, f(X_i) = z$. As conditioned on $\pi_{\text{aug}, < j}^{T'}$, the value of $\pi_{\text{aug}, j}^{T'}$ is just a function of exactly one of X_1, \dots, X_n , the lemma follows. ◀

► **Lemma 17.** We have:

$$\mathbb{E} \left[\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^0)_{|\overline{R^0}} \parallel \mathcal{D}_{|\overline{R^0}} \right) \right] = \mathbb{I}(X : \Pi_{\text{aug}}^0) \leq T + n \leq 2T.$$

Proof. The inequality follows from Fact 35 and Lemma 33. We now show the equality. As $R^0 = \emptyset$ and π_{aug}^0 is just π appended with the values $(\text{flag}(x_i))_{i \in [n]}$, we have:

$$\begin{aligned} \mathbb{E} \left[\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^0)_{|\overline{R^0}} \parallel \mathcal{D}_{|\overline{R^0}} \right) \right] &= \mathbb{E} \left[\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^0) \parallel \mathcal{D} \right) \right] \\ &= \sum_{\pi_{\text{aug}}^0} \sum_x \Pr(\pi_{\text{aug}}^0) \cdot \Pr(x \mid \pi_{\text{aug}}^0) \cdot \log \frac{\Pr(x \mid \pi_{\text{aug}}^0)}{\Pr(x)} \\ & \hspace{15em} \text{(Definition 37)} \\ &= \mathbb{I}(X : \Pi_{\text{aug}}^0). \hspace{10em} \text{(Lemma 36)} \end{aligned}$$

◀

Recall from Lemma 12 and Corollary 13 that fixing π_{aug}^{t-1} fixes the values of many variables in Algorithm 3. We now show:

► **Lemma 18.** For all $t \in [T']$ and all π_{aug}^{t-1} , we have:

$$\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^{t-1})_{|\overline{R^{t-1}}} \parallel \mathcal{D}_{|\overline{R^{t-1}}} \right) = \mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^{t-1})_{|\overline{R^t \setminus R^{t-1}}} \parallel \mathcal{D}_{|\overline{R^t \setminus R^{t-1}}} \right) + \mathbb{E} \left[\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^t)_{|\overline{R^t}} \parallel \mathcal{D}_{|\overline{R^t}} \right) \mid \pi_{\text{aug}}^{t-1} \right].$$

Proof. This essentially is just from the chain rule for KL-divergence. We give the details below. Note that:

$$\begin{aligned} &\mathbb{D} \left((\mathcal{D} \mid \pi_{\text{aug}}^{t-1})_{|\overline{R^{t-1}}} \parallel \mathcal{D}_{|\overline{R^{t-1}}} \right) \\ &= \sum_{x_{|\overline{R^{t-1}}}} \Pr(x_{|\overline{R^{t-1}}} \mid \pi_{\text{aug}}^{t-1}) \cdot \log \frac{\Pr(x_{|\overline{R^{t-1}}} \mid \pi_{\text{aug}}^{t-1})}{\Pr(x_{|\overline{R^{t-1}}})} \hspace{5em} \text{(Definition 37)} \\ &= \sum_{x_{|\overline{R^{t-1}}}} \Pr(x_{|\overline{R^{t-1}}} \mid \pi_{\text{aug}}^{t-1}) \cdot \log \frac{\Pr(x_{|\overline{R^t}} \mid x_{|\overline{R^t \setminus R^{t-1}}}, \pi_{\text{aug}}^{t-1}) \Pr(x_{|\overline{R^t \setminus R^{t-1}}} \mid \pi_{\text{aug}}^{t-1})}{\Pr(x_{|\overline{R^t}} \mid x_{|\overline{R^t \setminus R^{t-1}}}) \Pr(x_{|\overline{R^t \setminus R^{t-1}}})} \end{aligned}$$

$$\begin{aligned}
 &= \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) \\
 &\quad + \sum_{x_{|\overline{\mathbb{R}^{t-1}}} } \Pr\left(x_{|\overline{\mathbb{R}^{t-1}}} \mid \pi_{\text{aug}}^{t-1}\right) \cdot \log \frac{\Pr\left(x_{|\overline{\mathbb{R}^t}} \mid x_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}, \pi_{\text{aug}}^{t-1}\right)}{\Pr\left(x_{|\overline{\mathbb{R}^t}} \mid x_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right)}. \quad (\text{Definition 37})
 \end{aligned}$$

To continue, recall that all the coordinates of all players are mutually independent in the distribution \mathcal{D} . Moreover, we have from Lines 6, 8, 9, and 11 that conditioned on π_{aug}^{t-1} , the event $x_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}$ is the same as the corresponding event π_{aug}^t . We get:

$$\begin{aligned}
 &\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\overline{\mathbb{R}^{t-1}}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^{t-1}}}\right) \\
 &= \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) + \sum_{x_{|\overline{\mathbb{R}^t}}} \sum_{\pi_{\text{aug}}^t} \Pr\left(x_{|\overline{\mathbb{R}^t}}, \pi_{\text{aug}}^t \mid \pi_{\text{aug}}^{t-1}\right) \cdot \log \frac{\Pr\left(x_{|\overline{\mathbb{R}^t}} \mid \pi_{\text{aug}}^t\right)}{\Pr\left(x_{|\overline{\mathbb{R}^t}}\right)} \\
 &= \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) + \sum_{\pi_{\text{aug}}^t} \Pr\left(\pi_{\text{aug}}^t \mid \pi_{\text{aug}}^{t-1}\right) \cdot \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right) \\
 &\hspace{15em} (\text{Definition 37}) \\
 &= \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) + \mathbb{E}\left[\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right) \mid \pi_{\text{aug}}^{t-1}\right]. \quad \blacktriangleleft
 \end{aligned}$$

► **Lemma 19.** *Let $0 \leq t \leq T'$ and π_{aug}^0 be given. Let P^t be any set of transcripts π_{aug}^t that all have π_{aug}^0 as a prefix. It holds that:*

$$\sum_{\pi_{\text{aug}}^t \in P^t} \Pr\left(\pi_{\text{aug}}^t \mid \pi_{\text{aug}}^0\right) \cdot \nabla\left(\pi_{\text{aug}}^t\right) \leq \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^0\right)_{|\overline{\mathbb{R}^0}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^0}}\right),$$

where:

$$\nabla\left(\pi_{\text{aug}}^t\right) = \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right) + \sum_{t'=1}^t \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t'}\right)_{|\mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}} \parallel \mathcal{D}_{|\mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}}\right).$$

Proof. We prove the lemma by induction on t . The base case $t = 0$ is straightforward. We prove the lemma for $t > 0$ assuming it holds for $t - 1$. Fix a set P^t as in the lemma statement and define, for all $0 \leq t' < t$, the set $P^{t'}$ to be the set of all $\pi_{\text{aug}}^{t'}$ that are prefixes of a $\pi_{\text{aug}}^t \in P^t$. We have:

$$\begin{aligned}
 &\sum_{\pi_{\text{aug}}^t \in P^t} \Pr\left(\pi_{\text{aug}}^t \mid \pi_{\text{aug}}^0\right) \cdot \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right) \\
 &= \sum_{\pi_{\text{aug}}^{t-1} \in P^{t-1}} \Pr\left(\pi_{\text{aug}}^{t-1} \mid \pi_{\text{aug}}^0\right) \cdot \mathbb{E}\left[\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right) \mid \pi_{\text{aug}}^{t-1}\right] \\
 &= \sum_{\pi_{\text{aug}}^{t-1} \in P^{t-1}} \Pr\left(\pi_{\text{aug}}^{t-1} \mid \pi_{\text{aug}}^0\right) \cdot \\
 &\hspace{10em} \left(\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\overline{\mathbb{R}^{t-1}}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^{t-1}}}\right) - \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right)\right). \\
 &\hspace{15em} (\text{Lemma 18})
 \end{aligned}$$

To continue, we use the induction hypothesis on the first term. We get:

$$\sum_{\pi_{\text{aug}}^t \in P^t} \Pr\left(\pi_{\text{aug}}^t \mid \pi_{\text{aug}}^0\right) \cdot \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^t\right)_{|\overline{\mathbb{R}^t}} \parallel \mathcal{D}_{|\overline{\mathbb{R}^t}}\right)$$

$$\begin{aligned}
&\leq \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^0\right)_{|\overline{\mathcal{R}^0}} \parallel \mathcal{D}_{|\overline{\mathcal{R}^0}}\right) \\
&\quad - \sum_{\pi_{\text{aug}}^{t-1} \in P^{t-1}} \Pr\left(\pi_{\text{aug}}^{t-1} \mid \pi_{\text{aug}}^0\right) \cdot \sum_{t'=1}^t \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t'-1}\right)_{|\mathcal{R}^{t'} \setminus \mathcal{R}^{t'-1}} \parallel \mathcal{D}_{|\mathcal{R}^{t'} \setminus \mathcal{R}^{t'-1}}\right) \\
&\leq \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^0\right)_{|\overline{\mathcal{R}^0}} \parallel \mathcal{D}_{|\overline{\mathcal{R}^0}}\right) \\
&\quad - \sum_{\pi_{\text{aug}}^t \in P^t} \Pr\left(\pi_{\text{aug}}^t \mid \pi_{\text{aug}}^0\right) \cdot \sum_{t'=1}^t \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t'-1}\right)_{|\mathcal{R}^{t'} \setminus \mathcal{R}^{t'-1}} \parallel \mathcal{D}_{|\mathcal{R}^{t'} \setminus \mathcal{R}^{t'-1}}\right).
\end{aligned}$$

Rearranging gives the result. \blacktriangleleft

► **Corollary 20.** *Let π_{aug}^0 be given and $P^{T'}$ be any set of transcripts $\pi_{\text{aug}}^{T'}$ that all have π_{aug}^0 as a prefix. It holds that:*

$$\sum_{\pi_{\text{aug}}^{T'} \in P^{T'}} \Pr\left(\pi_{\text{aug}}^{T'} \mid \pi_{\text{aug}}^0\right) \cdot \sum_{t=1}^{T'} \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{|\mathcal{R}^t \setminus \mathcal{R}^{t-1}} \parallel \mathcal{D}_{|\mathcal{R}^t \setminus \mathcal{R}^{t-1}}\right) \leq \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^0\right)_{|\overline{\mathcal{R}^0}} \parallel \mathcal{D}_{|\overline{\mathcal{R}^0}}\right).$$

6.2 Many Clean Transcripts

Recall from Corollary 13 that fixing π_{aug} fixes the values of the values computed in Line 5. By definition, it also fixes the values computed in Line 2. Using this, we define the following events that are just some subsets of all possible $\pi_{\text{aug}}^{T'}$:

1. Define the event $\mathcal{E}_{\text{clean,flag}}$ to be the set of all $\pi_{\text{aug}}^{T'}$ such that for all $i \in [n]$, we have:

$$\text{flag}(x_i) = 0.$$

2. Define the event $\mathcal{E}_{\text{clean},x}$ to be the set of all $\pi_{\text{aug}}^{T'}$ such that for all $j \in [m]$, we have:

$$\Pr\left(\sum_{i=1}^n x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right) \leq 2^{-\theta^5 n}.$$

3. Define the event $\mathcal{E}_{\text{clean},S}$ to be the set of all $\pi_{\text{aug}}^{T'}$ such that:

$$S_{\text{cell}}^{T'} = S_{\text{pair}}^{T'} = S_{\text{col}}^{T'} = \emptyset.$$

The goal of this section is to show that a randomly sampled transcript $\pi_{\text{aug}}^{T'}$ is likely to be clean. Namely, if we define $\mathcal{E}_{\text{clean}} = \mathcal{E}_{\text{clean,flag}} \wedge \mathcal{E}_{\text{clean},x} \wedge \mathcal{E}_{\text{clean},S}$, we have:

► **Lemma 21.** *It holds that:*

$$\Pr(\overline{\mathcal{E}_{\text{clean}}}) < \frac{1}{2}.$$

Lemma 21 follows from Lemmas 22–24 proven below.

► **Lemma 22.** *It holds that:*

$$\Pr(\overline{\mathcal{E}_{\text{clean,flag}}}) \leq \theta^{30}.$$

Proof. We have:

$$\begin{aligned}
\Pr(\exists i \in [n] : \text{flag}(x_i) = 1) &\leq n \cdot \Pr(\text{flag}(x_1) = 1) && \text{(As } x_i \text{ are identically distributed)} \\
&\leq n \cdot 2^{-\theta^4 m} && \text{(Lemma 3)} \\
&\leq 2^{-\theta^5 m}. && \blacktriangleleft
\end{aligned}$$

► **Lemma 23.** *It holds that:*

$$\Pr(\overline{\mathcal{E}_{\text{clean},x}}) \leq \theta^{30}.$$

Proof. We have:

$$\begin{aligned} & \Pr\left(\exists j \in [m] : \Pr\left(\sum_{i=1}^n x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right) > 2^{-\theta^5 n}\right) \\ & \leq \sum_{j=1}^m \Pr\left(\Pr\left(\sum_{i=1}^n x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right) > 2^{-\theta^5 n}\right) && \text{(Union bound)} \\ & \leq 2^{\theta^5 n} \cdot \sum_{j=1}^m \mathbb{E}\left[\Pr\left(\sum_{i=1}^n x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right)\right] && \text{(Markov's Inequality)} \\ & \leq 2^{\theta^5 n} \cdot \sum_{j=1}^m \Pr\left(\sum_{i=1}^n x_{i,j} \geq \theta n\right) \\ & \leq 2^{\theta^5 n} \cdot m \cdot 2^{-\theta^4 n} && \text{(Lemma 3)} \\ & \leq 2^{-\theta^5 n}. \end{aligned}$$

► **Lemma 24.** *It holds that:*

$$\Pr(\overline{\mathcal{E}_{\text{clean},S}}) \leq \theta^{30}.$$

Proof. To start, define the event \mathcal{E} to be the set of all π_{aug}^0 such that

$\mathbb{D}\left((\mathcal{D} \mid \pi_{\text{aug}}^0)_{|\overline{R^0}} \parallel \mathcal{D}_{|\overline{R^0}}\right) \leq T \cdot \theta^{-40}$. By Markov's inequality and Lemma 17, we have $\Pr(\mathcal{E}) \leq \theta^{35}$. Using the chain rule, this implies that:

$$\Pr(\overline{\mathcal{E}_{\text{clean},S}}) \leq \Pr(\overline{\mathcal{E}}) + \Pr(\overline{\mathcal{E}_{\text{clean},S}} \mid \mathcal{E}) \leq \theta^{35} + \Pr(\overline{\mathcal{E}_{\text{clean},S}} \mid \mathcal{E}).$$

Thus, it suffices to show that the last term is bounded by θ^{35} . We will show this holds even under a stronger conditioning. Specifically, we fix an arbitrary π_{aug}^0 such that \mathcal{E} happens and show that $\Pr(\overline{\mathcal{E}_{\text{clean},S}} \mid \pi_{\text{aug}}^0) \leq \theta^{35}$. Fix any such π_{aug}^0 . We first claim that:

▷ **Claim 25.** For all $\pi_{\text{aug}}^{T'}$ that extend π_{aug}^0 for which $\mathcal{E}_{\text{clean},S}$ does not happen, we have:

$$\sum_{t=1}^{T'} \mathbb{D}\left((\mathcal{D} \mid \pi_{\text{aug}}^{t-1})_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) \geq T' \theta^{400}.$$

The lemma now follows as, defining $P^{T'}$ to be the set of all $\pi_{\text{aug}}^{T'}$ that extend π_{aug}^0 for which $\mathcal{E}_{\text{clean},S}$ does not happen, we get:

$$\begin{aligned} \Pr(\overline{\mathcal{E}_{\text{clean},S}} \mid \pi_{\text{aug}}^0) &= \sum_{\pi_{\text{aug}}^{T'} \in P^{T'}} \Pr\left(\pi_{\text{aug}}^{T'} \mid \pi_{\text{aug}}^0\right) \\ &\leq \frac{1}{T' \theta^{400}} \sum_{\pi_{\text{aug}}^{T'} \in P^{T'}} \Pr\left(\pi_{\text{aug}}^{T'} \mid \pi_{\text{aug}}^0\right) \cdot \sum_{t=1}^{T'} \mathbb{D}\left((\mathcal{D} \mid \pi_{\text{aug}}^{t-1})_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D}_{|\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) && \text{(Claim 25)} \\ &\leq \frac{1}{T' \theta^{400}} \mathbb{D}\left((\mathcal{D} \mid \pi_{\text{aug}}^0)_{|\overline{R^0}} \parallel \mathcal{D}_{|\overline{R^0}}\right) && \text{(Corollary 20)} \\ &\leq \frac{T}{T' \theta^{440}} && \text{(Choice of } \pi_{\text{aug}}^0) \\ &\leq \theta^{50}. \end{aligned}$$

It remains to show Claim 25

Proof of Claim 25. Fix an arbitrary $\pi_{\text{aug}}^{T'}$ as in the statement of the claim. As $\mathcal{E}_{\text{clean},S}$ does not happen, we have that at least one of $S_{\text{cell}}^{T'}$, $S_{\text{pair}}^{T'}$, $S_{\text{col}}^{T'}$ is non-empty. Applying Lemma 15, we get that for all $t \in [T']$, at least one of S_{cell}^t , S_{pair}^t , S_{col}^t is non-empty. Due to Lines 6 and 9, this means that in all iterations $t \in [T']$, the parties either execute Line 8 or they execute Line 11. Let $Z_{\text{col}} \subseteq [T']$ be the set of all iterations t where parties execute Line 8 and $(i_1^t, j_1^t) \in S_{\text{col}}^t \setminus S_{\text{cell}}^t$. We claim that $|Z_{\text{col}}| \leq T' \cdot (1 - \theta^{120})$.

We prove this claim later, but assuming it for now, we have that either the parties execute Line 11 or they execute Line 8 and $(i_1^t, j_1^t) \in S_{\text{cell}}^t$. In either case, note from Line 5 that $\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D} \mid_{\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right) \geq \theta^{200}$. This means that:

$$T' \theta^{120} \leq T' - |Z_{\text{col}}| \leq \theta^{-200} \cdot \sum_{t \in [T'] \setminus Z_{\text{col}}} \mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{t-1}\right)_{\mathbb{R}^t \setminus \mathbb{R}^{t-1}} \parallel \mathcal{D} \mid_{\mathbb{R}^t \setminus \mathbb{R}^{t-1}}\right).$$

As the KL-divergence is non-negative, we have the claim. It remains to show that $|Z_{\text{col}}| \leq T' \cdot (1 - \theta^{120})$. For this, consider the following relation $M \subseteq Z_{\text{col}} \times ([T'] \setminus Z_{\text{col}})$. For $t \in Z_{\text{col}}$ and $t' \in [T'] \setminus Z_{\text{col}}$, we have $(t, t') \in M$ if and only if there exists $i' \in [n]$ such that $(i', j_1^t) \in \mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}$. We will show that:

- (a) For all $t \in Z_{\text{col}}$, there are at least $\theta^{100}n$ values of $t' \in [T'] \setminus Z_{\text{col}}$ such that $(t, t') \in M$.
- (b) For all $t' \in [T'] \setminus Z_{\text{col}}$, there are at most $2n$ values of $t \in Z_{\text{col}}$ such that $(t, t') \in M$.

Using Items a and b, we get:

$$|Z_{\text{col}}| \cdot \theta^{100}n \leq |M| \leq (T' - |Z_{\text{col}}|) \cdot 2n.$$

It follows that $T' \cdot (1 - \theta^{120})$. It remains to show Items a and b. For Item a, fix an arbitrary $t \in Z_{\text{col}}$ and, for all $0 \leq t' \leq T'$, define the value $Y(t') = \left| \left\{ i' \in [n] \mid \mathbb{R}_{i', j_1^t}^{t'} = 1 \right\} \right|$. Observe that from Algorithm 3 that

1. $Y(0) = 0$.
2. For all $t' \in [T']$, we have $Y(t' - 1) \leq Y(t') \leq Y(t' - 1) + 1$.
3. If $t' \in [T']$ is such that $Y(t') \leq Y(t' - 1) + 1$, there exists $i' \in [n]$ such that $(i', j_1^t) \in \mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}$.

To see why Item a follows, consider the smallest $t^* \in Z_{\text{col}}$ such that $j_1^t = j_1^{t^*}$. As $t^* \in Z_{\text{col}}$, we have $(i_1^{t^*}, j_1^{t^*}) \in S_{\text{col}}^{t^*}$. By Line 5, this means that $Y(t^* - 1) \geq \theta^{100} \cdot n$. Together with Items 1 and 2, this means that there are at least $\theta^{100}n$ values of $t' \in [t^* - 1]$ such that $Y(t') = Y(t' - 1) + 1$. Moreover, none of these values are in Z_{col} as otherwise, Item 3 implies that $j_1^{t'} = j_1^t$, a contradiction to the choice of t^* . Using Item 3 again, it follows that all these values satisfy $(t, t') \in M$, as desired.

For Item b, fix an arbitrary $t' \in [T'] \setminus Z_{\text{col}}$ and suppose for contradiction that there are at least $2n + 1$ values of $t \in Z_{\text{col}}$ such that $(t, t') \in M$. Each of these $2n + 1$ values of t has a corresponding value of (i_1^t, j_1^t) which are all distinct (due to the fact that Line 5 only adds (i, j) to S_{col} if $\mathbb{R}_{i,j} = 0$). As all $i_1^t \in [n]$, the fact that there are $2n + 1$ distinct values of (i_1^t, j_1^t) imply that these values must contain at least 3 distinct values of j_1^t . This implies that there are at least three distinct values of j_1^t such that there exists $i' \in [n]$ for which $(i', j_1^t) \in \mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}$. This is a contradiction as Algorithm 3 guarantees that $|\mathbb{R}^{t'} \setminus \mathbb{R}^{t'-1}| \leq 2$. \triangleleft

6.3 Properties of Clean Transcripts

Lemma 21 shows that the probability that a transcript is not clean is small. Thus, a randomly sampled transcript is likely to be clean. We now establish some properties of clean transcripts. Throughout this subsection, we fix a transcript $\pi_{\text{aug}}^{T'}$ for which $\mathcal{E}_{\text{clean}}$ happens. This also fixes the values of $R^0, \dots, R^{T'}$ and other variables that are determined by $\pi_{\text{aug}}^{T'}$. As $\mathcal{E}_{\text{clean},S}$, $\mathcal{E}_{\text{clean},x}$ and $\mathcal{E}_{\text{clean},\text{flag}}$ happen, we have $S_{\text{cell}}^{T'} = S_{\text{pair}}^{T'} = S_{\text{col}}^{T'} = \emptyset$ and for all $j \in [m]$, it holds that $\Pr\left(\sum_{i \in [n]} x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right) \leq 2^{-\theta^5 n}$ and that $\text{flag}(x_i) = 0$ for all $i \in [n]$. Moreover, as the size R increases by at most 2 in any iteration, we have that:

$$\left| R^{T'} \right| \leq 2T' \leq \theta^{250} mn. \quad (4)$$

We claim that:

► **Lemma 26.** *For all $j \in [m]$, we have:*

$$\left| R^{T'} \cap ([n] \times \{j\}) \right| < n \implies \left| R^{T'} \cap ([n] \times \{j\}) \right| < \theta^{100} \cdot n.$$

Proof. Fix such a j . As $S_{\text{cell}}^{T'} = S_{\text{pair}}^{T'} = S_{\text{col}}^{T'} = \emptyset$, we have by Lines 6 and 9 that $R^{T'} = R^{T'-1}$. Also, as $S_{\text{col}}^{T'} = \emptyset$, we have by Line 5 that:

$$\left| R^{T'-1} \cap ([n] \times \{j\}) \right| < \theta^{100} \cdot n \vee \forall i' \in [n] : (i', j) \in R^{T'-1}.$$

As $R^{T'} = R^{T'-1}$, we are done. ◀

Due to Lemma 26, we can partition the values $j \in [m]$ into two sets as follows:

$$\begin{aligned} J_{\text{fix}} &= \left\{ j \in [m] \mid \left| R^{T'} \cap ([n] \times \{j\}) \right| = n \right\}. \\ J_{\text{unfix}} &= \left\{ j \in [m] \mid \left| R^{T'} \cap ([n] \times \{j\}) \right| < \theta^{100} \cdot n \right\}. \end{aligned} \quad (5)$$

Recall from Lemma 14 that $\pi_{\text{aug}}^{T'}$ determines the values of $x_{i,j}$ for all $(i,j) \in R^{T'}$. We have:

► **Lemma 27.** *For all $j \in J_{\text{fix}}$, it holds that $\sum_{i=1}^n x_{i,j} < \theta n$.*

Proof. For all $j \in J_{\text{fix}}$, we have by Equation (5) that $(i,j) \in R^{T'}$ for all $i \in [n]$. By Lemma 14, this means that $\pi_{\text{aug}}^{T'}$ determines $x_{i,j}$ for all $i \in [n]$. The lemma then follows as we have $\Pr\left(\sum_{i \in [n]} x_{i,j} \geq \theta n \mid \pi_{\text{aug}}^{T'}\right) \leq 2^{-\theta^5 n}$. ◀

► **Lemma 28.** *We have:*

1. *For all $(i,j) \notin R^{T'}$ and all $b \in \{0,1\}$,*

$$\Pr\left(x_{i,j} = b \mid \pi_{\text{aug}}^{T'}\right) \geq \theta^{26}.$$

2. *For all $i \in [n]$, $j \neq j' \in [m]$ such that $(i,j), (i,j') \notin R^{T'}$, and all $b \in \{0,1\}$:*

$$\Pr\left((x_{i,j}, x_{i,j'}) = (1, b) \mid \pi_{\text{aug}}^{T'}\right) \leq \theta^{20} \cdot \Pr\left(x_{i,j'} = b \mid \pi_{\text{aug}}^{T'}\right).$$

Proof. Recall that $S_{\text{cell}}^{T'} = S_{\text{pair}}^{T'} = S_{\text{col}}^{T'} = \emptyset$ and also recall from Corollary 13 that these sets are determined by $\pi_{\text{aug}}^{T'-1}$. The fact that these sets are empty imply that $\pi_{\text{aug}}^{T'-1}$ also determines $\pi_{\text{aug}}^{T'}$, which means that they both determine one another. It follows that the distributions $\mathcal{D} \mid \pi_{\text{aug}}^{T'-1}$ and $\mathcal{D} \mid \pi_{\text{aug}}^{T'}$ are identical. We now prove each part in turn.

1. As $R^{T'-1} \subseteq R^{T'}$, we have $(i, j) \notin R^{T'-1}$. Recall that $S_{\text{cell}}^{T'} = \emptyset$ implying from Line 5 that $\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'-1}\right)_{|(i,j)} \parallel \mathcal{D}_{|(i,j)}\right) < \theta^{200}$. As the distributions $\mathcal{D} \mid \pi_{\text{aug}}^{T'-1}$ and $\mathcal{D} \mid \pi_{\text{aug}}^{T'}$ are identical, we get $\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'}\right)_{|(i,j)} \parallel \mathcal{D}_{|(i,j)}\right) < \theta^{200}$. By Fact 39, this means that $\left\|\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'}\right)_{|(i,j)} - \mathcal{D}_{|(i,j)}\right\|_{\text{TV}} < \theta^{100}$ which by Definition 38 and the definition of \mathcal{D} implies the result.
2. As $R^{T'-1} \subseteq R^{T'}$, we have $(i, j), (i, j') \notin R^{T'-1}$. Recall that $S_{\text{pair}}^{T'} = \emptyset$ implying from Line 5 that $\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'-1}\right)_{|(i,j),(i,j')} \parallel \mathcal{D}_{|(i,j),(i,j')}\right) < \theta^{100}$. As the distributions $\mathcal{D} \mid \pi_{\text{aug}}^{T'-1}$ and $\mathcal{D} \mid \pi_{\text{aug}}^{T'}$ are identical, we get $\mathbb{D}\left(\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'}\right)_{|(i,j),(i,j')} \parallel \mathcal{D}_{|(i,j),(i,j')}\right) < \theta^{100}$. By Fact 39, this means that $\left\|\left(\mathcal{D} \mid \pi_{\text{aug}}^{T'}\right)_{|(i,j),(i,j')} - \mathcal{D}_{|(i,j),(i,j')}\right\|_{\text{TV}} < \theta^{50}$. It follows that:

$$\begin{aligned}
\Pr\left((x_{i,j}, x_{i,j'}) = (1, b) \mid \pi_{\text{aug}}^{T'}\right) &\leq \Pr((x_{i,j}, x_{i,j'}) = (1, b)) + \theta^{50} && \text{(Definition 38)} \\
&\leq \theta^{25} \cdot \Pr(x_{i,j'} = b) + \theta^{50} \\
&\leq \theta^{24} \cdot \Pr(x_{i,j'} = b) \\
&\leq \theta^{24} \cdot \left(\Pr(x_{i,j'} = b \mid \pi_{\text{aug}}^{T'}) + \theta^{50}\right) && \text{(Definition 38)} \\
&\leq \theta^{20} \cdot \Pr(x_{i,j'} = b \mid \pi_{\text{aug}}^{T'}). && \text{(Item 1)}
\end{aligned}$$

◀

Moreover, we have from Equation (4) that $J_{\text{unfix}} \neq \emptyset$. Fix an arbitrary $j^* \in J_{\text{unfix}}$. We now use j^* to define some important sets of the parties' inputs. Let x' be an input in the support of \mathcal{D} . We say that x' is relevant if for all $j \neq j^* \in [m]$, we have $\sum_{i=1}^n x_{i,j} \leq \theta n$. We define \mathcal{X}_{rel} to be the set of all relevant inputs. We say that x' sets j^* to zero (respectively, one) if for all $i \in [n]$, we have $x'_{i,j^*} = x_{i,j^*}$ if $(i, j^*) \in R^{T'}$ (recall from Lemma 14 that $\pi_{\text{aug}}^{T'}$ determines the values of $x_{i,j}$ for all $(i, j) \in R^{T'}$) and $x'_{i,j^*} = 0$ (resp. $x'_{i,j^*} = 1$) otherwise. We let $\mathcal{X}_{\text{zero}}$ and \mathcal{X}_{one} be the set of all inputs that set j^* to zero and one respectively. We claim that:

► **Lemma 29.** *For all $x' \in \mathcal{X}_{\text{rel}} \cap \mathcal{X}_{\text{one}}$ such that $\Pr(x' \mid \pi_{\text{aug}}^{T'}) > 0$, we have $\text{GM-Test}_{\theta,n}^m(x') = \{0, 1\}^m \setminus \{0^m\}$. For all $x' \in \mathcal{X}_{\text{rel}} \cap \mathcal{X}_{\text{zero}}$ such that $\Pr(x' \mid \pi_{\text{aug}}^{T'}) > 0$, we have $\text{GM-Test}_{\theta,n}^m(x') = \{0^m\}$.*

Proof. We only prove the former as the latter is analogous. For this, fix $x' \in \mathcal{X}_{\text{rel}} \cap \mathcal{X}_{\text{one}}$ and examine the cases in Equation (2). Recall that $\Pr(x' \mid \pi_{\text{aug}}^{T'}) > 0$ implies that $\text{flag}(x'_i) = 0$ for all $i \in [n]$. We finish the proof by showing that:

$$\text{GapMaj}_{\theta,n}^{\theta,m}(x') = \left\{ \left(\underbrace{0, \dots, 0}_{j^*-1 \text{ times}}, 1, \underbrace{0, \dots, 0}_{m-j^* \text{ times}} \right) \right\}. \quad (6)$$

Indeed, we have from $x' \in \mathcal{X}_{\text{rel}}$ that $\sum_{i=1}^n x'_{i,j} \leq \theta n$ for all $j \neq j^* \in [m]$. We also have from $j^* \in J_{\text{unfix}}$ and $x' \in \mathcal{X}_{\text{one}}$ that $\sum_{i=1}^n x'_{i,j} \geq n - \theta^{100}n$. Moreover, we also have from $\text{flag}(x'_i) = 0$ for all $i \in [n]$ that the Hamming weight of x'_i is at most $\frac{\theta m}{2}$ for all $i \in [n]$. Combine these to get Equation (6). ◀

► **Lemma 30.** For all $\mathcal{X} \in \{\mathcal{X}_{\text{one}}, \mathcal{X}_{\text{zero}}\}$, it holds that:

$$\Pr\left(\mathbf{X} \in \mathcal{X} \cap \mathcal{X}_{\text{rel}} \mid \pi_{\text{aug}}^{T'}\right) \geq \theta^{30n}.$$

Proof. We show the result for $\mathcal{X} = \mathcal{X}_{\text{one}}$ as the proof for $\mathcal{X}_{\text{zero}}$ is analogous. For $i \in [n]$, define $x_{i,j^*}^* = x_{i,j^*}$ if $(i, j^*) \in \mathbf{R}^{T'}$ (recall from Lemma 14 that $\pi_{\text{aug}}^{T'}$ determines the values of $x_{i,j}$ for all $(i, j) \in \mathbf{R}^{T'}$) and $x_{i,j^*}^* = 0$ otherwise. Thus, we have:

$$\begin{aligned} \Pr\left(\mathbf{X} \in \mathcal{X}_{\text{one}} \mid \pi_{\text{aug}}^{T'}\right) &= \Pr\left(\forall i \in [n] : x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right) \\ &= \prod_{i=1}^n \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right), \end{aligned} \quad (7)$$

where the last step uses Lemma 16. We also have

$$\begin{aligned} &\Pr\left(\mathbf{X} \in \mathcal{X}_{\text{one}} \cap \overline{\mathcal{X}_{\text{rel}}} \mid \pi_{\text{aug}}^{T'}\right) \\ &= \Pr\left(\forall i \in [n] : x'_{i,j^*} = x_{i,j^*}^* \wedge \exists j \neq j^* \in [m] : \sum_{i=1}^n x'_{i,j} > \theta n \mid \pi_{\text{aug}}^{T'}\right) \\ &\leq \sum_{j \neq j^* \in [m]} \Pr\left(\forall i \in [n] : x'_{i,j^*} = x_{i,j^*}^* \wedge \sum_{i=1}^n x'_{i,j} > \theta n \mid \pi_{\text{aug}}^{T'}\right) \quad (\text{Union bound}) \\ &\leq \sum_{j \neq j^* \in [m]} \sum_{\substack{Z \subseteq [n] \\ |Z| = \theta n}} \Pr\left(\forall i \in [n] : x'_{i,j^*} = x_{i,j^*}^* \wedge \forall i \in Z : x'_{i,j} = 1 \mid \pi_{\text{aug}}^{T'}\right) \quad (\text{Union bound}) \\ &\leq \sum_{j \neq j^* \in [m]} \sum_{\substack{Z \subseteq [n] \\ |Z| = \theta n}} \prod_{i \in Z} \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right) \prod_{i \in Z} \Pr\left((x'_{i,j}, x_{i,j^*}^*) = (1, x_{i,j^*}^*) \mid \pi_{\text{aug}}^{T'}\right) \\ &\leq \sum_{j \neq j^* \in [m]} \sum_{\substack{Z \subseteq [n] \\ |Z| = \theta n}} \theta^{20 \cdot \theta n} \cdot \prod_{i=1}^n \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right). \end{aligned} \quad (\text{Lemma 16})$$

Now, note that there are at most $n \cdot \left(\frac{3}{\theta}\right)^{\theta n} \leq \left(\frac{4}{\theta}\right)^{\theta n}$ terms in the sum (as $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$). We get using $\theta < 1/2$ that:

$$\begin{aligned} &\Pr\left(\mathbf{X} \in \mathcal{X}_{\text{one}} \cap \overline{\mathcal{X}_{\text{rel}}} \mid \pi_{\text{aug}}^{T'}\right) \\ &\leq \left(\frac{4}{\theta}\right)^{\theta n} \cdot \theta^{20 \cdot \theta n} \cdot \prod_{i=1}^n \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right) \\ &\leq \frac{1}{2} \cdot \prod_{i=1}^n \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right). \end{aligned}$$

Combining with Equation (7), we get:

$$\begin{aligned} \Pr\left(\mathbf{X} \in \mathcal{X}_{\text{one}} \cap \mathcal{X}_{\text{rel}} \mid \pi_{\text{aug}}^{T'}\right) &\geq \frac{1}{2} \cdot \prod_{i=1}^n \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right) \\ &= \frac{1}{2} \cdot \prod_{i:(i,j^*) \notin \mathbf{R}^{T'}} \Pr\left(x'_{i,j^*} = x_{i,j^*}^* \mid \pi_{\text{aug}}^{T'}\right) \\ &\geq \theta^{30n}. \end{aligned} \quad (\text{As } \pi_{\text{aug}}^{T'} \text{ determines } x_{i,j} \text{ for all } (i, j) \in \mathbf{R}^{T'}) \quad (\text{Lemma 28, Item 1})$$

◀

6.4 Finishing the Proof

We are now ready to finish the proof of Theorem 10.

Proof of Theorem 10. Recall that it suffices to show Equation (3). We have:

$$\begin{aligned}
 \Pr(\Pi(X) \in \text{GM-Test}_{\theta,n}^m(X)) &\leq \Pr(\overline{\mathcal{E}_{\text{clean}}}) + \Pr(\mathcal{E}_{\text{clean}}) \cdot \Pr(\Pi(X) \in \text{GM-Test}_{\theta,n}^m(X) \mid \mathcal{E}_{\text{clean}}) \\
 &\hspace{15em} \text{(Union bound)} \\
 &= 1 - \Pr(\mathcal{E}_{\text{clean}}) \cdot \Pr(\Pi(X) \notin \text{GM-Test}_{\theta,n}^m(X) \mid \mathcal{E}_{\text{clean}}) \\
 &\leq 1 - \frac{1}{2} \cdot \Pr(\Pi(X) \notin \text{GM-Test}_{\theta,n}^m(X) \mid \mathcal{E}_{\text{clean}}). \quad \text{(Lemma 21)}
 \end{aligned}$$

Thus, it suffices to lower bound the last probability by θ^{30n} . We will show this holds even under a stronger conditioning. Specifically, we fix an arbitrary $\pi_{\text{aug}}^{T'}$ such that $\mathcal{E}_{\text{clean}}$ happens and show that $\Pr(\Pi(X) \notin \text{GM-Test}_{\theta,n}^m(X) \mid \pi_{\text{aug}}^{T'}) \geq \theta^{30n}$. Fix any such $\pi_{\text{aug}}^{T'}$ and recall that fixing $\pi_{\text{aug}}^{T'}$ also fixes the output of the protocol. As the sets in the two cases of Lemma 29 are disjoint, we have:

$$\begin{aligned}
 &\Pr(\Pi(X) \notin \text{GM-Test}_{\theta,n}^m(X) \mid \pi_{\text{aug}}^{T'}) \\
 &\geq \min\left(\Pr(\Pi(X) \in \mathcal{X}_{\text{rel}} \cap \mathcal{X}_{\text{one}} \mid \pi_{\text{aug}}^{T'}), \Pr(\Pi(X) \in \mathcal{X}_{\text{rel}} \cap \mathcal{X}_{\text{zero}} \mid \pi_{\text{aug}}^{T'})\right) \\
 &\geq \theta^{30n}. \hspace{15em} \text{(Lemma 30)}
 \end{aligned}$$

◀

References

- 1 Abhinav Aggarwal, Varsha Dani, Thomas P Hayes, and Jared Saia. Distributed computing with channel noise. *arXiv preprint*, 2016. [arXiv:1612.05943](#).
- 2 Noga Alon, Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Reliable communication over highly connected noisy networks. In *Symposium on Principles of Distributed Computing (DISC)*, pages 165–173. ACM, 2016.
- 3 Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.
- 4 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. *SIAM Journal on Computing*, 42(3):1327–1363, 2013.
- 5 Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Symposium on Theory of Computing (STOC)*, pages 999–1010. ACM, 2016.
- 6 Mark Braverman and Anup Rao. Information equals amortized communication. In Rafail Ostrovsky, editor, *Symposium on Foundations of Computer Science (FOCS)*, pages 748–757. IEEE Computer Society, 2011.
- 7 Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct product via round-preserving compression. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 232–243. Springer, 2013.
- 8 Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct products in communication complexity. In *Symposium on Foundations of Computer Science (FOCS)*, pages 746–755. IEEE, 2013.
- 9 Keren Censor-Hillel, Ran Gelles, and Bernhard Haeupler. Making asynchronous distributed computations robust to noise. *Distributed Computing*, 32:405–421, 2019.

- 10 Keren Censor-Hillel, Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Broadcasting in noisy radio networks. In *Symposium on Principles of Distributed Computing (PODC)*, pages 33–42, 2017.
- 11 Keren Censor-Hillel, Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Erasure correction for noisy radio networks. In *International Symposium on Distributed Computing (DISC)*, 2019.
- 12 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Symposium on Foundations of Computer Science (FOCS)*, pages 270–278. IEEE, 2001.
- 13 Imrich Chlamtac and Shay Kutten. On broadcasting in radio networks—problem analysis and protocol design. *IEEE Trans. Communications*, 33(12):1240–1246, 1985.
- 14 Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Computation over the noisy broadcast channel with malicious parties. In *Innovations in Theoretical Computer Science Conference, (ITCS)*, volume 185, pages 82:1–82:19, 2021.
- 15 Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Tight bounds for general computation in noisy broadcast networks. In *Symposium on Foundations of Computer Science (FOCS)*, pages 634–645, 2021.
- 16 Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Protecting single-hop radio networks from message drops. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 261 of *LIPICs*, pages 53:1–53:20, 2023.
- 17 Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive coding over the noisy broadcast channel. In *Symposium on Theory of Computing (STOC)*, pages 507–520. ACM, 2018.
- 18 Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Radio network coding requires logarithmic overhead. In *Foundations of Computer Science (FOCS)*, pages 348–369, 2019.
- 19 Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive error resilience beyond $2/7$. In *Symposium on Theory of Computing (STOC)*. ACM, 2020.
- 20 Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Noisy beeps. In Yuval Emek and Christian Cachin, editors, *Symposium on Principles of Distributed Computing (PODC)*, pages 418–427, 2020.
- 21 Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Optimal error resilience of adaptive message exchange. In *Symposium on Theory of Computing (STOC)*, pages 1235–1247. ACM, 2021.
- 22 Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM Journal on computing*, 24(4):736–750, 1995.
- 23 Uriel Feige and Joe Kilian. Finding OR in a noisy broadcast network. *Information Processing Letters*, 73(1-2):69–75, 2000.
- 24 Robert G. Gallager. Finding parity in a simple broadcast network. *IEEE Transactions on Information Theory*, 34(2):176–180, 1988.
- 25 Abbas El Gamal. Open problems presented at the 1984 workshop on specific problems in communication and computation sponsored by bell communication research. “*Open Problems in Communication and Computation*”, by Thomas M. Cover and B. Gopinath (editors). Springer-Verlag, 1987.
- 26 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication for boolean functions. *Journal of the ACM*, 63(5):46:1–46:31, 2016.
- 27 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of communication and external information. *SIAM Journal on computing*, 50(3), 2021.
- 28 Ran Gelles and Yael T Kalai. Constant-rate interactive coding is impossible, even in constant-degree networks. *IEEE Transactions on Information Theory*, 65(6):3812–3829, 2019.

- 29 Ran Gelles, Yael Tauman Kalai, and Govind Ramnarayan. Efficient multiparty interactive coding—part i: Oblivious insertions, deletions and substitutions. *IEEE Transactions on Information Theory*, 67(6):3411–3437, 2021.
- 30 Ran Gelles, Yael Tauman Kalai, and Govind Ramnarayan. Efficient multiparty interactive coding—part ii: Non-oblivious noise. *IEEE Transactions on Information Theory*, 68(7):4723–4749, 2022.
- 31 Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient and explicit coding for interactive communication. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 768–777. IEEE, 2011.
- 32 Navin Goyal, Guy Kindler, and Michael Saks. Lower bounds for the noisy broadcast problem. *SIAM Journal on Computing*, 37(6):1806–1841, 2008.
- 33 Bernhard Haeupler. Interactive channel capacity revisited. In *Foundations of Computer Science (FOCS)*, pages 226–235. IEEE, 2014.
- 34 Prahladh Harsha, Rahul Jain, David McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. In *Conference on Computational Complexity (CCC)*, pages 10–23. IEEE, 2007.
- 35 William M. Hoza and Leonard J. Schulman. The adversarial noise threshold for distributed protocols. In *Symposium on Discrete Algorithms (SODA)*, pages 240–258, 2016.
- 36 Abhishek Jain, Yael Tauman Kalai, and Allison Bishop Lewko. Interactive coding for multiparty protocols. In *Symposium on Theory of computing (STOC)*, pages 1–10, 2015.
- 37 Rahul Jain. New strong direct product results in communication complexity. *Journal of the ACM*, 62(3):1–27, 2015.
- 38 Rahul Jain, Attila Pereszlényi, and Penghui Yao. A direct product theorem for two-party bounded-round public-coin communication complexity. *Algorithmica*, 76:720–748, 2016.
- 39 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A direct sum theorem in communication complexity via message compression. In *Colloquium on Automata, Languages, and Programming (ICALP)*, pages 300–315. Springer, 2003.
- 40 Mauricio Karchmer, Eyal Kushilevitz, and Noam Nisan. Fractional covers and communication complexity. *SIAM Journal on Discrete Mathematics*, 8(1):76–92, 1995.
- 41 Hartmut Klauck. A strong direct product theorem for disjointness. In *Symposium on Theory of Computing (STOC)*, pages 77–86, 2010.
- 42 Eyal Kushilevitz and Yishay Mansour. Computation in noisy radio networks. *SIAM Journal on Discrete Mathematics (SIDMA)*, 19(1):96–108, 2005.
- 43 Allison Lewko and Ellen Vitercik. Balancing communication for multi-party interactive coding. *arXiv preprint*, 2015. [arXiv:1503.06381](https://arxiv.org/abs/1503.06381).
- 44 Manuj Mukherjee and Ran Gelles. Multiparty interactive coding over networks of intersecting broadcast links. *IEEE Journal on Selected Areas in Information Theory*, 2(4):1078–1092, 2021.
- 45 Ilan Newman. Computing in fault tolerance broadcast networks. In *Computational Complexity Conference (CCC)*, pages 113–122, 2004.
- 46 Sridhar Rajagopalan and Leonard J. Schulman. A coding theorem for distributed computation. In *Symposium on the Theory of Computing (STOC)*, pages 790–799, 1994.
- 47 Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733. IEEE, 1992.
- 48 Andrew Chi-Chih Yao. On the complexity of communication under noise. *invited talk in the 5th ISTCS Conference*, 1997.
- 49 Huacheng Yu. Strong xor lemma for communication with bounded rounds. In *Symposium on Foundations of Computer Science (FOCS)*, pages 1186–1192. IEEE, 2022.

A

 Information Theory Preliminaries

Recall that we use sans-serif letters to denote random variables. We reserve E to denote an arbitrary event. All random variables will be assumed to be discrete and we shall adopt the convention $0 \log \frac{1}{0} = 0$. When it is clear from context, we may abbreviate the event $X = x$ as just x . All logarithms are taken with base 2.

A.1 Entropy

► **Definition 31** (Entropy). *The (binary) entropy of X is defined as:*

$$\mathbb{H}(X) = \sum_{x \in \text{supp}(X)} \Pr(x) \cdot \log \frac{1}{\Pr(x)}.$$

The entropy of X conditioned on E is defined as:

$$\mathbb{H}(X | E) = \sum_{x \in \text{supp}(X)} \Pr(x | E) \cdot \log \frac{1}{\Pr(x | E)}.$$

► **Definition 32** (Conditional Entropy). *We define the conditional entropy of X given Y and E as:*

$$\mathbb{H}(X | Y, E) = \sum_{y \in \text{supp}(Y)} \Pr(y | E) \cdot \mathbb{H}(X | y, E).$$

Henceforth, we shall omit writing the $\text{supp}(\cdot)$ when it is clear from context.

► **Lemma 33.** *It holds for all X and E that:*

$$0 \leq \mathbb{H}(X | E) \leq \log(|\text{supp}(X)|).$$

The second inequality is tight if and only if X conditioned on E is the uniform distribution over $\text{supp}(X)$.

A.2 Mutual Information

► **Definition 34** (Mutual Information). *The mutual information between X and Y is defined as:*

$$\mathbb{I}(X : Y) = \mathbb{H}(X) - \mathbb{H}(X | Y) = \mathbb{H}(Y) - \mathbb{H}(Y | X).$$

The mutual information between X and Y conditioned on Z is defined as:

$$\mathbb{I}(X : Y | Z) = \mathbb{H}(X | Z) - \mathbb{H}(X | YZ) = \mathbb{H}(Y | Z) - \mathbb{H}(Y | XZ).$$

► **Fact 35.** *We have $0 \leq \mathbb{I}(X : Y | Z) \leq \mathbb{H}(X)$.*

► **Lemma 36.** *We have:*

$$\mathbb{I}(X : Y | Z) = \sum_{x,y,z} \Pr(x, y, z) \cdot \log \frac{\Pr(x, y | z)}{\Pr(x | z) \Pr(y | z)}.$$

A.3 KL Divergence

► **Definition 37** (KL Divergence). *If μ, ν are two distributions over the same (finite) set Ω , the Kullback-Leibler (KL) Divergence between μ and ν is defined as:*

$$\mathbb{D}(\mu \parallel \nu) = \sum_{\omega \in \Omega} \mu(\omega) \cdot \log \frac{\mu(\omega)}{\nu(\omega)}.$$

For a finite non-empty set S , we shall use $\mathcal{U}(S)$ to denote the uniform distribution over S . We omit S from the notation when it is clear from the context. We use $\text{dist}(X \mid E)$ to denote the distribution of the random variable X conditioned on the event E .

A.4 Total Variation Distance




► **Definition 38** (Total variation distance). *Let μ, ν be two distributions over the same (finite) set Ω . The total variation distance between μ and ν is defined as:*

$$\|\mu - \nu\|_{\text{TV}} = \max_{\Omega' \subseteq \Omega} \sum_{\omega \in \Omega'} \mu(\omega) - \nu(\omega).$$

► **Fact 39** (Pinsker's inequality). *Let μ, ν be two distributions over the same set Ω . It holds that:*

$$\|\mu - \nu\|_{\text{TV}} \leq \sqrt{\frac{1}{2} \cdot \mathbb{D}(\mu \parallel \nu)}.$$

Lower Bounds for Set-Multilinear Branching Programs

Prerona Chatterjee   

School of Computer Sciences, NISER Bhubaneswar, India

Deepanshu Kush   

Department of Computer Science, University of Toronto, Canada

Shubhangi Saraf   

Department of Computer Science, University of Toronto, Canada

Department of Mathematics, University of Toronto, Canada

Amir Shpilka   

Blavatnik School of Computer Science, Tel-Aviv University, Israel

Abstract

In this paper, we prove super-polynomial lower bounds for the model of *sum of ordered set-multilinear algebraic branching programs*, each with a possibly different ordering (\sum smABP). Specifically, we give an explicit nd -variate polynomial of degree d such that any \sum smABP computing it must have size $n^{\omega(1)}$ for d as low as $\omega(\log n)$. Notably, this constitutes the first such lower bound in the low degree regime. Moreover, for $d = \text{poly}(n)$, we demonstrate an exponential lower bound. This result generalizes the seminal work of Nisan (STOC, 1991), which proved an exponential lower bound for a single ordered set-multilinear ABP.

The significance of our lower bounds is underscored by the recent work of Bhargav, Dwivedi, and Saxena (TAMC, 2024), which showed that super-polynomial lower bounds against a sum of ordered set-multilinear branching programs – for a polynomial of sufficiently low degree – would imply super-polynomial lower bounds against general ABPs, thereby resolving Valiant’s longstanding conjecture that the permanent polynomial can not be computed efficiently by ABPs. More precisely, their work shows that if one could obtain such lower bounds when the degree is bounded by $O(\log n / \log \log n)$, then it would imply super-polynomial lower bounds against general ABPs.

Our results strengthen the works of Arvind & Raja (Chic. J. Theor. Comput. Sci., 2016) and Bhargav, Dwivedi & Saxena (TAMC, 2024), as well as the works of Ramya & Rao (Theor. Comput. Sci., 2020) and Ghoshal & Rao (International Computer Science Symposium in Russia, 2021), each of which established lower bounds for related or restricted versions of this model. They also strongly answer a question from the former two, which asked to prove super-polynomial lower bounds for general \sum smABP.

2012 ACM Subject Classification Theory of computation \rightarrow Algebraic complexity theory

Keywords and phrases Lower Bounds, Algebraic Branching Programs, Set-multilinear polynomials

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.20

Related Version *Full Version*: <https://ecc.weizmann.ac.il/report/2023/212/>

Full Version: <https://arxiv.org/abs/2312.15874>

Funding *Prerona Chatterjee*: This work was done as a postdoctoral student at Tel Aviv University, where the research was funded by the Azrieli International Postdoctoral Fellowship, the Israel Science Foundation (grant number 514/20) and the Len Blavatnik and the Blavatnik Family foundation.

Shubhangi Saraf: Research partially supported by a Sloan research fellowship and an NSERC Discovery Grant.

Amir Shpilka: Research leading to these results has received funding from the Israel Science Foundation (grant number 514/20) and from the Len Blavatnik and the Blavatnik Family foundation.



© Prerona Chatterjee, Deepanshu Kush, Shubhangi Saraf, and Amir Shpilka;

licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 20; pp. 20:1–20:20

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Acknowledgements Parts of this work were done while the first author was visiting TIFR Mumbai and ICTS-TIFR Bengaluru, and she would like to thank Venkata Susmita Biswas, Ramprasad Saptharishi, Prahladh Harsha and Jaikumar Radhakrishnan for the hospitality. The first author would also like to thank Anamay Tengse for useful discussions.

1 Introduction

1.1 Background on Algebraic Complexity

In his seminal work ([40]) in 1979, Valiant proposed an algebraic framework to study the computational complexity of computing *polynomials*. *Algebraic Complexity Theory* is this study of the complexity of computational problems which can be described as computing a multivariate polynomial $P(x_1, \dots, x_N)$ over some elements x_1, \dots, x_N lying in a fixed field. Several fundamental computational tasks such as computing the determinant, permanent, matrix product, etc., can be represented using this framework. The natural computational models that we investigate in this setting are models such as *algebraic circuits*, *algebraic branching programs*, and *algebraic formulas*.

An *algebraic circuit* over a field \mathbb{F} for a multivariate polynomial $P(x_1, \dots, x_N)$ is a directed acyclic graph (DAG) whose internal vertices (called gates) are labeled as either $+$ (sum) or \times (product), and leaves (vertices of in-degree zero) are labeled by the variables x_i or constants from \mathbb{F} . A special output gate (the root of the DAG) represents the polynomial P . If the DAG happens to be a tree, such a resulting circuit is called an *algebraic formula*. The size of a circuit or formula is the number of nodes in the DAG. We also consider the product-depth of the circuit, which is the maximum number of product gates on a root-to-leaf path. The class VP (respectively, VF) is then defined to be the collection of all polynomials having at most polynomially large degree which can be computed by polynomial-sized circuits (respectively, formulas).

The class VP is synonymous to what we understand as efficiently computable polynomials. The class VNP, whose definition is similar to the boolean class NP, is in some sense a notion of what we deem as *explicit*. Much like the problem of proving circuit size lower bounds for explicit boolean functions, the problem of proving them for explicit *polynomials* (i.e., showing $VP \neq VNP$) has also remained elusive for many decades. However, because the latter only deals with formal *symbolic* computation as opposed to modelling *semantic* truth-table constraints, it is widely believed to be easier to resolve than its boolean counterpart. In fact, it is even known to be a *pre-requisite* to the $P \neq NP$ conjecture in the non-uniform setting ([8]).

An *algebraic branching program* (ABP) is a layered DAG with two special nodes in it: a start-node and an end-node. All edges of the ABP go from layer $\ell - 1$ to layer ℓ for some ℓ (say start-node is the unique node in layer 0 and end-node is the unique node in the last layer) and are labeled by a linear polynomial. Every directed path γ from start-node to end-node computes the monomial P_γ , which is the product of all labels on the path γ . The ABP computes the polynomial $P = \sum_\gamma P_\gamma$, where the sum is over all paths γ from start-node to end-node. Its size is simply the number of nodes in the DAG, its *depth* is the length of the longest path from the start-node to the end-node, and *width* is the maximum number of nodes in any layer. The class VBP is then defined to be the collection of all polynomials (with polynomially-bounded degree) which can be computed by polynomial-sized branching programs. ABPs are known to be of intermediate complexity between formulas and circuits; in other words, we know the inclusions $VF \subseteq VBP \subseteq VP \subseteq VNP$.

It is conjectured that all of these inclusions are strict, and resolving any of these conjectures would represent a dramatic advancement in algebraic complexity theory, and even more broadly, in circuit complexity overall. Valiant's original hypothesis in [40] pertains to showing a super-polynomial separation between the complexity of computing the determinant and the permanent polynomials. This is known to be equivalent to the $\text{VBP} \neq \text{VNP}$ conjecture, i.e., showing super-polynomial size lower bounds against ABPs computing *explicit* polynomials. At present, the best known lower bound against ABPs is only quadratic ([9]), and it appears as though we are quite distant from addressing this conjecture. On the other hand, as we now elaborate, while not directly improving upon this quadratic bound, this paper makes significant progress towards a different line of attack aimed at resolving Valiant's conjecture.

1.2 Set-Multilinearity: A Key Syntactic Restriction

One key advantage that algebraic models offer over their boolean counterparts is that of *syntactic* restrictions. A recurring theme in algebraic complexity theory is to first efficiently convert general models of computation (such as circuits or formulas) to special kinds of syntactically-restricted models, show strong lower bounds against these restricted models, and then recover non-trivial lower bounds against the original general models owing to the efficiency of this conversion. This phenomenon is termed *hardness escalation*. In this subsection, we describe one crucial example of a syntactic restriction in detail, that of *set-multilinearity*.

A polynomial is said to be *homogeneous* if each monomial has the same total degree and *multilinear* if every variable occurs at most once in any monomial. Now, suppose that the underlying variable set is partitioned into d sets X_1, \dots, X_d . Then the polynomial is said to be *set-multilinear* with respect to this variable partition if each monomial in P has *exactly* one variable from each set. Note that a set-multilinear polynomial is both multilinear and homogeneous, and has degree precisely d if it is set-multilinear over d sets. Next, we define different models of computation corresponding to these variants of polynomials classes. An algebraic formula/branching program/circuit is set-multilinear with respect to a variable partition (X_1, \dots, X_d) if each internal node in the formula/branching program/circuit computes a set-multilinear polynomial.¹ Multilinear and homogeneous formulas/branching programs/circuits are defined analogously.

We now describe several important hardness escalation results, each reducing general models to corresponding *set-multilinear* models.

Constant depth circuits

The recent celebrated breakthrough work of Limaye, Srinivasan, and Tavenas ([27]) establishes super-polynomial lower bounds for general algebraic circuits for *all* constant-depths, a problem that was open for many decades. In order to show this, it is first shown that general low-depth algebraic formulas can be converted to set-multilinear algebraic formulas of low depth as well, and without much of a blow-up in size (as long as the degree is small). Subsequently, strong lower bounds are established for low-depth set-multilinear circuits (of small enough degree), which when combined with the first step yields the desired lower bound for general constant-depth circuits.

¹ Of course, a non-root node need not be set-multilinear with respect to the *entire* variable partition. Nevertheless, here we demand that it must be set-multilinear with respect to some *subset* of the collection $\{X_1, \dots, X_d\}$.

Formulas

Raz [33] showed that if an N -variate set-multilinear polynomial of degree d has an algebraic formula of size s , then it also has a set-multilinear formula of size $\text{poly}(s) \cdot (\log s)^d$. In particular, for a set-multilinear polynomial P of degree $d = O(\log N / \log \log N)$, it follows that P has a formula of size $\text{poly}(N)$ if and only if P has a set-multilinear formula of size $\text{poly}(N)$. Thus, having $N^{\omega_a(1)}$ set-multilinear formula size lower bounds for such a low degree would imply super-polynomial lower bounds for general formulas. A recent line of work by Kush and Saraf ([25, 26]) can be viewed as an attempt to prove general formula lower bounds via this route.

Algebraic Branching Programs

In fact, in the context of ABPs as well, the very recent work of Bhargav, Dwivedi, and Saxena ([5]) reduces the problem of showing lower bounds against general ABPs to proving lower bounds against sums of ordered set-multilinear ABPs (again, as long as the degree is small enough). Ordered set-multilinear ABPs are, in fact, historically well-studied models and extremely well-understood. However, despite their apparent simplicity, the work [5] implies that understanding their *sums* – a model that is far less studied – is at the forefront of understanding Valiant’s conjecture. We state their result formally as Theorem 1.5 in Section 1.3.

First however, as this is also the main model considered in this paper, we begin by formally defining *ordered* set-multilinear ABPs and outlining their importance.

► **Definition 1.1** (Ordered smABP). *Given a variable partition (X_1, \dots, X_d) , we say that a set-multilinear branching program of depth d is said to be ordered with respect to an ordering (or permutation) $\sigma \in S_d$ if for each $\ell \in [d]$, all edges of the ABP from layer $\ell - 1$ to layer ℓ are labeled using a linear form over the variables in $X_{\sigma(\ell)}$. It is simply said to be ordered if there exists an ordering σ such that it is ordered with respect to σ .*

At this point, it is essential to take note of the terminology in this context: in this paper, a general (or “unordered”) set-multilinear branching program refers to an ABP for which each internal node computes a polynomial that is set-multilinear with respect to *some* subset of the global partition, whereas an *ordered* set-multilinear branching program is more specialized and has the property that any two nodes in the same layer compute polynomials that are set-multilinear with respect to the *same* partition.

► **Remark 1.2.** This notion of ordered set-multilinear branching programs turns out to be equivalent to the more commonly used notions of (i) “read-once oblivious algebraic branching programs (ROABPs)”, as well as (ii) “non-commutative algebraic branching programs” (see, for example, [12]). This relationship, especially with the former model, is described in more detail later in Section 1.4.

► **Definition 1.3** (\sum smABP). *Given a polynomial $P(X)$ that is set-multilinear with respect to the variable partition $X = (X_1, \dots, X_d)$, we say that $\sum_{i=1}^t A_i$ is a \sum smABP computing P if indeed $\sum_{i=1}^t A_i(X) = P(X)$, and each A_i is an ordered set-multilinear branching program i.e., each A_i is ordered with respect to some $\sigma_i \in S_d$. We call t (i.e., the number of summands in a \sum smABP) its support size and define its max-width and total-width to be the maximum over the width of each A_i and the sum of the width of each A_i , respectively.*

We have known *exponential* width lower bounds against a *single* ordered set-multilinear ABP since the foundational work of Nisan. In [29], he showed that there are explicit polynomials (in fact, in VP) which require any ordered set-multilinear ABP computing

them to be of exponentially large width. Viewed differently, this work even shows that in the non-commutative setting, $\text{VBP} \neq \text{VP}^2$. More crucially however, this work introduced a powerful technique – a notion known as the *partial derivative method* – that has been instrumental in the bulk of the major advancements in algebraic complexity theory over the past three decades (such as [30, 32, 36, 19, 23, 20, 24, 27, 39], see also [38, 37]).

Despite the considerable development of the partial derivative technique over the course of these works (and many more) for proving strong lower bounds against various algebraic models, relatively little is known about a *general sum* of ordered set-multilinear ABPs – a simple and direct generalization of the original model considered by Nisan. There is some progress in the literature towards this goal but which still requires additional structural restrictions on either the max-width or the support size or the size of each part in the variable partition. The work [3] of Arvind and Raja shows that any \sum smABP of support size t computing the $n \times n$ permanent polynomial requires max-width (and therefore, total-width) at least $2^{\Omega(n/t)}$. Note that for this bound to be super-polynomial, the support size needs to be heavily restricted i.e., t must be sub-linear. On the other hand, the work [5] also shows a super-polynomial lower bound in this context: it implies that no \sum smABP of polynomially-bounded total-width can compute the iterated matrix multiplication (IMM) polynomial. However, their work requires the additional assumption that the max-width of such an \sum smABP is $n^{o(1)}$, that is *sub-polynomial* in the number of variables.

Apart from these, Ramya and Rao ([31]) use the partial derivative method to show an exponential lower bound against the related model of sum of ROABPs in the multilinear setting, as well as some other structured multilinear ABPs. Their lower bounds are for a multilinear polynomial that is computable by a small multilinear circuit. Ghoshal and Rao ([13]) partially extend their work by proving an exponential lower bound, for a polynomial that is computable even by a small multilinear ABP, against sums of ROABPs that have polynomially bounded width. Notably, these results can be viewed as lower bounds against the \sum smABP model where each variable set in the variable partition has size 2 (that is, the total number of variables is $2d$). This is because a multilinear polynomial and any multilinear model computing it (such as circuit, formula, or branching program) can be converted, in a generic manner, to a set-multilinear polynomial and the corresponding set-multilinear model respectively, with each variable set having size 2 (also see Section 1.5 for a discussion). However, from the perspective of hardness escalation of [5] that is described above – and which is indeed the focus of our work – the setting of d that is far more interesting is when it is allowed to be considerably smaller than n . More precisely, the framework of [5] requires $d = O(\log n / \log \log n)$ (stated formally as Theorem 1.5 below). A detailed discussion about the results in [31], [13] and how they compare with our work can be found in Section 1.5.

1.3 Our Results

Our main result in this paper is a super-polynomial lower bound against an unrestricted sum of ordered set-multilinear branching programs, for a hard polynomial with “small” degree. We first state this result formally below, and then subsequently explain the connection with the hardness escalation result of [5] that is alluded to in the previous subsections.

► **Theorem 1.4** (“Low”-Degree \sum smABP Lower Bounds). *Let $d \leq n$ be growing parameters satisfying $d = \omega(\log n)$. There is a $\Theta(dn)$ -variate degree d set-multilinear polynomial $F_{n,d}$ in VP such that $F_{n,d}$ cannot be computed by a \sum smABP of total-width $\text{poly}(n)$.*

² We briefly explain the connection between ordered set-multilinear ABPs and non-commutative computation in Section 1.4.

Next, we formally state the aforementioned hardness escalation result of [5]. In words, in order to show $\text{VBP} \neq \text{VP}$, it suffices to show lower bounds for any \sum smABP computing a polynomial P whose degree is at most *about logarithmic* in the number of variables. Towards this goal, our main result above (Theorem 1.4) shows a super-polynomial lower bound for any \sum smABP computing an explicit set-multilinear polynomial, whose degree is *barely super-logarithmic* in the number of variables. In this sense, it approaches the resolution of Valiant’s conjecture.

► **Theorem 1.5** (Hardness Escalation of [5]). *Let n, d be growing parameters with $d = O(\log n / \log \log n)$. Let $P_{n,d}$ be a $\Theta(dn)$ -variate degree d set-multilinear polynomial in VP (respectively, VNP). If $P_{n,d}$ cannot be computed by a \sum smABP of total-width $\text{poly}(n)$, then $\text{VBP} \neq \text{VP}$ (respectively, $\text{VBP} \neq \text{VNP}$).*

Next, we also give an explicit set-multilinear polynomial (with polynomially-large degree) such that any \sum smABP computing it must require *exponential* total-width. This strongly answers a question left open in both [3] and [5].

► **Theorem 1.6** (Exponential Lower Bounds for \sum smABP). *There is a set-multilinear polynomial $F_{n,n}$ in VP , in $\Theta(n^2)$ variables and of degree $\Theta(n)$, such that any \sum smABP computing $F_{n,n}$ requires total-width $\exp(\Omega(n^{1/3}))$.*

Theorem 1.6 and Theorem 1.4 are also true when $F_{n,d}$ (as defined in Section 3.3) is replaced by the appropriate Nisan-Wigderson polynomial $NW_{n,d}$ (as defined in Section 3.2), which is known to be in VNP . In fact, we first indeed established them for the Nisan-Wigderson polynomial, and then used some of the ideas presented in a recent work by Kush and Saraf ([26]) to make the hard polynomial lie in VP .³

With additional effort, and building upon the machinery⁴ of [26] (which, in turn, uses the techniques developed in [10]), we can almost recover the same lower bounds as in Theorem 1.6 and Theorem 1.4 for a set-multilinear polynomial even in VBP . We preferred to first state Theorem 1.6 and Theorem 1.4 in the manner above because (i) the proof is less intricate and in fact, even serves as a prelude to the proof of the latter, and (ii) to draw a direct comparison and contrast with the hardness escalation statement (Theorem 1.5). We now state these results for when the hard polynomial is the VBP polynomial and then describe two intriguing consequences.

► **Theorem 1.6’**. *There is a fixed constant $\delta \geq 1/100$ and a set-multilinear polynomial $G_{n,n}$ in VBP , in $\Theta(n^2)$ variables and of degree $\Theta(n)$, such that any \sum smABP computing $G_{n,n}$ requires total-width $\exp(\Omega(n^\delta))$.*

► **Theorem 1.4’**. *Let $d \leq n$ be growing parameters satisfying $d = \omega(\log n)$. There is a $\Theta(dn)$ -variate, degree $\Theta(d)$ set-multilinear polynomial $G_{n,d}$ in VBP such that $G_{n,d}$ cannot be computed by a \sum smABP of total-width $\text{poly}(n)$.*

The first intriguing consequence of proving the statements above is that we are able to show that the ABP set-multilinearization process given in [5] is nearly tight, as $G_{n,d}$ is known to have a small set-multilinear branching program and yet, any \sum smABP computing it must have large total-width. To make this point effectively, we first state the following key ingredient in the proof of Theorem 1.5, and subsequently state our tightness result.

³ We also acknowledge that an exponential lower bound – with weaker quantitative parameters – for the related model of multilinear ROABPs was obtained in [31]. For a comparison of this model with the \sum smABP model, see Sections 1.4 and 1.5.

⁴ This is explained in more detail in Section 1.6.

► **Lemma 1.7** (ABP Set-Multilinearization in [5]). *Let $P_{n,d}$ be a polynomial of degree d that is set-multilinear with respect to the partition $X = (X_1, \dots, X_d)$ where $|X_i| \leq n$ for all $i \in [d]$. If $P_{n,d}$ can be computed by an ABP of size s , then it can also be computed by a \sum smABP of max-width s and total-width $2^{O(d \log d)} s$.*

► **Theorem 1.8** (Near-Tightness of ABP Set-Multilinearization). *For large enough integers $\omega(\log n) = d \leq n$, there is a polynomial $G_{n,d}(X)$ which is set-multilinear over the variable partition $X = (X_1, \dots, X_d)$ with each $|X_i| \leq n$, and such that:*

- *it has a branching program of size $\text{poly}(n)$,*
- *but any \sum smABP of max-width $\text{poly}(n)$ computing $G_{n,d}$ requires total-width $2^{\Omega(d)}$.*

The second intriguing consequence is the fact that Theorem 1.8 can also be viewed as an exponential separation between the model of (general) small-width set-multilinear branching programs and the model of sums of small-width ordered set-multilinear branching programs. Moreover, we can improve this bound much further in the case of a single ordered set-multilinear branching program. More precisely, in Theorem 1.9 below, we answer a question posed in [26] about the relative strength of an unordered and (a single) ordered set-multilinear branching program, by obtaining a *near-optimal* separation. A priori, as is shown in [26] and as mentioned earlier in the introduction, if these two models coincided (i.e., if a general set-multilinear ABP could be simulated by a small and ordered one), then it would have led to super-polynomial lower bounds for general algebraic formulas.

► **Theorem 1.9** (Near-Optimal Separation between Ordered and Unordered smABPs). *There is a polynomial $G_{n,d}(X)$ which is set-multilinear over the variable partition $X = (X_1, \dots, X_d)$ with each $|X_i| \leq n$, and such that:*

- *it has a set-multilinear branching program of size $\text{poly}(n, d)$,*
- *but any ordered set-multilinear branching program computing $G_{n,d}$ requires width $n^{\Omega(d)}$.*

Note that $G_{n,d}$ has at most n^d monomials and so, it trivially has an ordered set-multilinear ABP of width n^d . Therefore, the lower bound above is essentially optimal.

1.4 The ROABP Perspective

One can also view all of our results described in Section 1.3 through the lens of another well-studied model in algebraic complexity theory, namely *read-once oblivious algebraic branching programs* (ROABPs).

► **Definition 1.10** (ROABP). *For integers n, d and a permutation $\sigma \in S_n$, an ABP over the variables x_1, \dots, x_n is said to be a read-once oblivious algebraic branching program (ROABP) in the order σ of individual degree d if for each $\ell \in [n]$, all edges from layer $\ell - 1$ to ℓ are labelled by univariate polynomials in $x_{\sigma(i)}$ of degree at most d .*

ROABPs were first introduced in this form by Forbes and Shpilka in [12], where it is also noted that proving lower bounds against ordered set-multilinear ABPs (as in Definition 1.1) is equivalent to proving lower bounds against ROABPs as well as non-commutative ABPs.

Suppose $f \in [X_1, \dots, X_d]$ is a set-multilinear polynomial with respect to $X_1 \sqcup \dots \sqcup X_d$ with $X_i = \{x_{i,1}, \dots, x_{i,n}\}$. Then we can define an associated polynomial $g_f \in [x_1, \dots, x_d]$ as follows.

$$g_f(x_1, \dots, x_d) = \sum_{\mathbf{e} \in [n]^d} \prod_{i=1}^n x_i^{e_i} \cdot \text{coefficient of } x_{i,e_i}.$$

Now let us assume that g_f can be computed by an ROABP of size s that is ordered with respect to $\sigma \in S_n$. Then a set-multilinear ABP ordered with respect to σ can be constructed using it, by simply replacing $x_i^{e_i}$ by x_{i,e_i} and erasing any degree zero components on each edge. It is easy to check that this computes f and we can use the lower bound against ordered set-multilinear ABPs for f to prove a lower bound against ROABPs for g_f . Conversely, given $g \in [x_1, \dots, x_n]$, we can define $f_g \in [X_1, \dots, X_n]$ with $X_i = \{x_{i,0}, x_{i,1}, \dots, x_{i,d}\}$ by replacing $x_i^{e_i}$ with x_{i,e_i} . We could then use an ordered set-multilinear ABP computing f_g to construct an ROABP (in the same order) computing g by using the inverse transformation, thereby proving that lower bounds against ROABPs imply lower bounds against ordered set-multilinear ABPs. Furthermore, the computation that an ROABP (or an ordered set-multilinear ABP) performs can be seen to be *non-commutative*. This is because the variables (or linear forms) along a path get multiplied in the *same* order σ as that of the ROABP (or ordered set-multilinear ABP).

As a consequence, exponential lower bounds follow for a single ROABP from the work of Nisan ([29]), and also from later works ([18, 21]). Using the transformation described above, our lower bounds (Theorem 1.6 and Theorem 1.6') can also be viewed as exponential lower bounds for the model of sum of ROABPs. The work of Ramya & Rao [31] also prove (weaker) exponential lower bounds against this model for a multilinear polynomial computable by multilinear circuits. In a follow-up work, Ghoshal & Rao [13] prove an exponential lower bound against sums of ROABPs with the additional restriction that the summand ROABPs have polynomially-bounded width for a multilinear polynomial computable by multilinear ABPs. On the other hand, the works of Arvind & Raja ([3]) and Bhargav, Dwivedi & Saxena ([5]) provide lower bounds in certain restricted versions of this model. Along with these, the work of Anderson, Forbes, Saptharishi, Shpilka, and Volk ([2]) also implies an exponential lower bound for a restricted version (for the sum of k ROABPs when $k = o(\log n)$).

Finally, we note that ROABPs have been studied extensively in the context of another central problem in algebraic complexity theory: that of polynomial identity testing (PIT). The PIT question for a general algebraic model \mathcal{M} is the following: Given access to an n -variate polynomial f of degree at most d that can be computed in the model \mathcal{M} of (an appropriate measure of) complexity at most s , determine whether $f \equiv 0$ in $\text{poly}(n, d, s)$ time. When one is given access to the model computing f explicitly, this flavour of PIT is called white-box PIT, and when one is merely provided query access to f , it is called black-box PIT.

The solution to the PIT problem for ROABPs in the white-box setting follows from a result by Raz and Shpilka ([34] – where it is stated in the equivalent language of non-commutative computation). However, the corresponding problem in the black-box setting remains open to this date, with the best-known time bound in the black-box setting still being only $s^{O(\log s)}$ due to the work by Forbes and Shpilka ([12]), who additionally assumed that the ordering of the ROABP is known. This was matched later by Agrawal, Gurjar, Korwar, and Saxena ([1]) in the unknown order setting, improving upon the work of Forbes, Saptharishi and Shpilka ([11]). Guo and Gurjar improved the result further by improving the dependence on the width [14]. Additionally, there have been various improvements to this result in restricted settings ([15, 17, 6]) and some other works that study PIT for a small sum of ROABPs ([16, 7, 14]). When the number of summands is super-constant, the question of even white-box PIT remains wide open.

1.5 Related Work

In this subsection, we discuss two closely related papers, namely those of Ramya & Rao [31] and Ghoshal & Rao [13]⁵, which study the model of sum of ROABPs in the multilinear setting. In [31, Theorem 1], the authors show that there exists an explicit multilinear polynomial (computable by a small multilinear circuit) such that any sum of ROABPs computing it has exponential size. In [13, Theorem 2], the authors show a similar lower bound for an explicit multilinear polynomial (computable by a small multilinear ABP) – albeit, in the restricted setting where the summand ROABPs have polynomially-bounded width.

Using the transformation described in Section 1.4, one can then view these lower bounds as ones against the \sum smABP model in the special case that each bucket in the variable partition has size 2. (To see how a multilinear polynomial say over the variable set x_1, \dots, x_d can be set-multilinearized trivially, here is a sketch: for each variable x_i , have a variable set X_i comprising of two fresh variables $x_{i,0}$ and $x_{i,1}$ in the new set-multilinear polynomial; here, the latter is to signify the “presence” of x_i in any monomial of the original multilinear polynomial, whereas the former is to signify its “absence”.) Additionally, it is not hard to see that the set-multilinearized version of the hard polynomials (in the manner just described) used in [31, Theorem 1] and [13, Theorem 2] are efficiently computable by small set-multilinear circuits and set-multilinear ABPs respectively. We note, however, that even so, our result in the high-degree setting where the hard polynomial is in VP (Theorem 1.6) is quantitatively better than [31, Theorem 1]. Additionally, our result in the high-degree setting where the hard polynomial is in VBP (Theorem 1.6’) is *both* quantitatively as well as qualitatively better than [13, Theorem 2] – the latter since we do not assume any bound on the width of the individual summand ordered set-multilinear ABPs. More crucially, our techniques enable us to prove super-polynomial bounds even when the degree is vastly smaller than the number of variables – in particular, when d is as low as $\omega(\log n)$ (Theorem 1.4 and Theorem 1.4’) – which is the more interesting regime of parameters due to the work of [5].

Ramya and Rao [31] also study another model, which they call *sum of α -set-multilinear ABPs*. They define α -set-multilinear ABPs to be ABPs with N^α layers, where N is the number of variables in the polynomial being computed. Any edge between layer $\ell - 1$ and ℓ in an α -set-multilinear ABP is labelled by an arbitrary multilinear polynomial over X_ℓ , where $X = X_1 \sqcup \dots \sqcup X_{N^\alpha}$ is a partition of the variable set. Then, for $\alpha \geq 1/10$, they establish exponential lower bounds against sum of α -set-multilinear ABPs for a polynomial that is multilinear, but which is *not* set-multilinear under the variable partition that the model respects. Hence, even though this model is more general than ordered set-multilinear ABPs, this result [31, Theorem 3] is also not comparable with ours as our hard polynomial is set-multilinear. Again, more crucially, the result [31, Theorem 3] does not handle the “low-degree” regime – a setting in which our techniques allow us to prove lower bounds.

1.6 Proof Overview

The organization of this subsection is as follows: we first describe the basics of the partial derivative method and summarize its typical application in proving lower bounds against a generic set-multilinear model of computation. Next, we briefly describe Nisan’s original partial derivative method from [29] to prove lower bounds specifically against a single ordered set-multilinear branching program. We then describe an alternative approach that yields a

⁵ We thank Ben Lee Volk and Utsab Ghosal for pointing out these papers to us after the release of an initial pre-print of this article, which erroneously claimed that it was the first to show super-polynomial lower bounds in the sum of ROABPs model.

20:10 Lower Bounds for Set-Multilinear Branching Programs

slightly weaker bound for the same model, but nevertheless is versatile enough that we can generalize it considerably more in order to prove Theorem 1.6 and Theorem 1.4. Finally, we describe the additional ideas needed in order to situate the hard polynomial in these theorems in VBP and in the process, establish the tightness result for ABP set-multilinearization (Theorem 1.8).

Partial Derivative Measure Basics

The high-level idea is to work with a measure that we show to be “small” for all polynomials computed by a specified model of computation – the model against which we wish to prove lower bounds. If we can also show that there is a “hard” polynomial for which the measure is in fact “large”, then it follows that this polynomial cannot be computed by the specified model. These *partial derivative measures*, after the initial work ([29]) by Nisan, were further developed by Nisan and Wigderson in [30], who used them to prove some constant-depth set-multilinear formula lower bounds. Since then, variations of these measures have also been used to prove various other stronger set-multilinear formula lower bounds (e.g., [27, 39, 28, 4, 25, 26]).

Given a variable partition (X_1, \dots, X_d) , the idea is to label each set of variables X_i as “+1” or “-1” according to *some* rule (called a “word”) $w \in \{-1, 1\}^d$. Let \mathcal{P}_w and \mathcal{N}_w denote the set of positive and negative indices (or coordinates) respectively, and let $\mathcal{M}_w^{\mathcal{P}}$ and $\mathcal{M}_w^{\mathcal{N}}$ denote the sets of all set-multilinear monomials over \mathcal{P}_w and \mathcal{N}_w respectively. For a polynomial f that is set-multilinear over the given variable partition (X_1, \dots, X_d) , the measure then is simply the rank of the “partial derivative matrix” $\mathcal{M}_w(f)$, whose rows are indexed by the elements of $\mathcal{M}_w^{\mathcal{P}}$ and columns indexed by $\mathcal{M}_w^{\mathcal{N}}$, and the entry of this matrix corresponding to a row m_1 and a column m_2 is the coefficient of the monomial $m_1 \cdot m_2$ in f .

For a subset $S \subseteq [d]$, let w_S denote the sum of those coordinates of w that lie in S . In other words, $|w_S|$ measures the amount of “bias” that the rule w exhibits when restricted to the S coordinates. Note that the rank of $\mathcal{M}_w(f)$ can never exceed $n^{(d-|w_{[d]}|)/2}$. Furthermore, we have that the rank measure is *multiplicative*: if f and g are polynomials that are set-multilinear over *disjoint* subsets of the global partition (X_1, \dots, X_d) , then the rank of $\mathcal{M}_w(f \cdot g)$ is the product of the ranks of $\mathcal{M}_w(f)$ and $\mathcal{M}_w(g)$. These two observations, combined with the sub-additivity of rank, provide a recipe for showing lower bounds against any given set-multilinear model of computation: the overall idea is to carefully split up the original model into smaller, multiplicatively disjoint parts and then argue the existence of a rule for which enough of these parts exhibit high bias. This process allows us to prove that the measure is small for the model of computation. Therefore, one can conclude that any explicit polynomial for which the measure is provably high – which needs to be established separately – can not be computed by this model. It is known ([25, 26]) that there is a set-multilinear polynomial $NW_{n,d}$ in VNP (see Section 3.2) as well as a set-multilinear polynomial $F_{n,d}$ in VP (see Section 3.3) for which the matrices $\mathcal{M}_w(NW_{n,d}), \mathcal{M}_w(F_{n,d})$, have full-rank, whenever $|\mathcal{P}_w| = |\mathcal{N}_w|$.

Nisan’s original lower bound

Let us first summarize how Nisan’s original partial derivative method from [29], as alluded to in Section 1.2, can be applied in this context to obtain lower bounds against the size of a single ordered set-multilinear ABP (ordered smABP) computing the aforementioned “full-rank” polynomials. Given any set-multilinear branching program A ordered with respect

to some permutation $\sigma \in S_d$ computing $F_{n,d}$, the idea is to pick a word w such that the $+1$ labels in w precisely correspond to the “left half” of the ordering σ , and the -1 labels correspond to the “right half”. One can then observe that the rank of $\mathcal{M}_w(F_{n,d}) = \mathcal{M}_w(A)$ serves as a lower bound on the number of nodes s in the middle layer of the ABP, yielding a near-optimal $n^{\Omega(d)}$ lower bound: this is because the matrix $\mathcal{M}_w(A)$ is easily seen to be the product of an $n^{d/2} \times s$ and an $s \times n^{d/2}$ matrix.

We now sketch an alternate proof: rather than constructing a word dependent on the ordering of variable sets X_i in the ordered smABP A as above, choose a uniformly random⁶ word w from $\{-1, 1\}^d$. We demonstrate that, with positive probability, the rank of $\mathcal{M}_w(A)$ is bounded by $s \cdot n^{d/2 - \Omega(\sqrt{d})}$, where s is the width of the middle layer in A : Standard anti-concentration bounds imply that, with at least constant probability, the bias in the left and right halves of A is $\Omega(\sqrt{d})$. Since A can be expressed as a sum of s polynomials $f_i \cdot g_i$ for $i \in [s]$, where each f_i and g_i are ordered smABPs with respect to disjoint subsets of the global partition, we encounter a loss of a factor of $n^{\Omega(\sqrt{d})}$ in the rank of the product polynomial $\mathcal{M}_w(f_i \cdot g_i)$ due to the bias of w . This, combined with the sub-additivity of rank, shows the desired bound of $s \cdot n^{d/2 - \Omega(\sqrt{d})}$ on the rank of $\mathcal{M}_w(A)$. Finally, we exploit the full-rank property of $F_{n,d}$ with respect to such words to establish a lower bound of $n^{\Omega(\sqrt{d})}$ on the width s of a single ordered smABP computing $F_{n,d}$. Notably, this bound is indeed slightly worse than what one can obtain by manually defining a rule w deterministically, which ensures a *maximal* bias of $d/2$ in each half of A as described in the paragraph above.

Generalization of the alternative argument

The alternative argument described above yields an exponential lower bound even for a *sum* of ordered smABPs, assuming the number of summands is small. Consider a \sum smABP of the form $\sum_{i=1}^t A_i$, of max-width s , computing $F_{n,d}$. For each summand A_i , the analysis above provides an upper bound of $s \cdot n^{d/2 - \Omega(\sqrt{d})}$ on the rank of $\mathcal{M}_w(A_i)$ with constant probability. If the number of summands t is a small enough constant, the union bound ensures the existence of a word w such that the rank of $\mathcal{M}_w(\sum A_i)$ is at most $t \cdot s \cdot n^{d/2 - \Omega(\sqrt{d})}$. Thus⁷, we obtain an exponential lower bound on $t \cdot s$ since this \sum smABP computes a full-rank polynomial. However, because of the use of the union bound in this manner, this method faces an inherent limitation – it is unable to handle more than a very small number of summands, even if we lower the bias demand from each half (e.g., from $\Omega(\sqrt{d})$ to $\Omega(\sqrt[4]{d})$) or a smaller polynomial in d). In fact, one can construct a sum of d ordered smABPs (by starting with a single smABP ordered arbitrarily and considering the d cyclic shifts of this ordering) such that any unbiased word w (i.e., $w_{[d]} = 0$) has the property that for at least one of the summands, the left and right halves will have no bias! Evidently then, in order to prove lower bounds against an *unrestricted* number of summands, we need another method to analyze the rank of the summands. Nonetheless, a conceptual takeaway from the exercise above is that selecting a rule w that is oblivious to the orderings of individual summands (and in particular, a *random* rule) still lets us derive strong lower bounds for the sum of *multiple* ordered smABPs.

Suppose instead of slicing an ordered smABP A down the middle, we slice it into three roughly equal pieces. Then, it is possible to write the polynomial computed by A as a sum over s^2 terms, each of the form $f_i \cdot g_i \cdot h_i$ where for each i , each of f_i, g_i, h_i depends on $d/3$

⁶ We also need to suitably condition on the event that the word w is symmetric (i.e., $|\mathcal{P}_w| = |\mathcal{N}_w|$) in order to use the full-rank property of the hard polynomial – the probability of this event is $\Theta(\frac{1}{\sqrt{d}})$. For ease of exposition, we omit the technical details in this sketch.

⁷ See footnote 6.

disjoint variable sets of the global partition. We can then perform a similar analysis as above to show enough bias across these 3 pieces, thereby obtaining a rank deficit. More precisely, we can conclude that for a single ordered smABP A , again with a constant probability, the rank of $\mathcal{M}_w(A)$ is at most $s^2 \cdot n^{d/2 - \Omega(\sqrt{d})}$. When we slice the ABP into 3 pieces in this way, it is not immediately clear where the gain is. In fact, for a single ordered smABP, this method actually gives a worse lower bound on s due to the presence of the factor of s^2 . Where we gain is in the magnitude of the *probability* with which we can guarantee that a single ordered smABP has a rank deficit – we will now describe how this observation allows us to take a union bound over many more summands.

In order to illustrate this trade-off more clearly, we will partition the ordered smABP A into many more pieces. Suppose we slice it into $q \approx \sqrt{d}$ pieces, each of size roughly $r = d/q \approx \sqrt{d}$ (this is just one setting of parameters; q and r are suitably optimized in the final proof). Thus, the polynomial that A computes can be written as a sum of at most s^{q-1} terms, where each term is a product of q polynomials – each set-multilinear over a disjoint subset of the global partition, where each piece has size r . When a word w is chosen randomly, each such piece again exhibits a bias of about $\Omega(\sqrt{r})$ with constant probability. The crucial observation then is that by known concentration bounds, it can be shown that with probability *exponentially* close to 1, the sum of the biases across all the q pieces is $\Omega(q\sqrt{r}) = \text{poly}(d)$. For a single ordered smABP A , this shows that the rank of $\mathcal{M}_w(A)$ is at most $s^q \cdot n^{-\Omega(q\sqrt{r})}$, which is still enough to show an exponential lower bound on s , even though it is worse than what we obtained by slicing into fewer pieces.

The key advantage in implementing this analysis is that it provides a way to argue that for a random word w , $\mathcal{M}_w(A)$ has low rank for a single ordered smABP A – with probability *exponentially* close to 1. In particular, this allows us to union bound over exponentially many ordered smABPs and show that even if we have an \sum smABP computing $F_{n,d}$ of exponential support size, with high probability, each summand will have a rank deficit. Then, again using the sub-additivity of rank, we can conclude that the sum has a rank deficit as well.

This method of analyzing the rank of an ordered smABP by partitioning it into *numerous* pieces and tactfully using concentration bounds is novel, and conceptually the most essential aspect of the proof. As we demonstrated above, this method of analysis indeed gives a worse bound for a single smABP. However, while mildly sacrificing what we can prove about the rank of a single ordered smABP, we are able to leverage it to still prove something meaningful about the rank of a *sum* with a much larger number of summands.

Our partial derivative measure draws inspiration from previously known lower bounds in the context of multilinear and set-multilinear *formulas* ([32, 25]). One noteworthy distinction lies in the analysis of the measure: whereas the partitioning is present intrinsically in those formula settings, in our setting of ABPs, we deliberately introduce the partitioning at the expense of a notable increase in the number of summands or the total-width (and therefore, in the number of events we union bound over). The substantial advantage gained in utilizing this partitioning for rank analysis justifies the tolerable increase in the total-width.

Tightness of ABP set-multilinearization

In order to make the hard polynomial in Theorems 1.6 and 1.4 lie in VBP, one might wonder if we can get away with using the same rank measure (i.e., rank of the matrix $M_w(\cdot)$ for a uniformly random word $w \in \{-1, 1\}^d$) that was used in the analysis above for the VP polynomial $F_{n,d}$. However, as far as we know, full-rank polynomials (in the sense described above) may also require super-polynomial sized set-multilinear ABPs. Thus, in order to prove a separation between (general) set-multilinear ABPs and (sums of) ordered set-multilinear ABPs, we seek a property that is weaker than being full-rank and yet is still useful enough for

proving lower bounds against our model. For this, we rely upon the *arc-partition* framework that is developed in [26] in order to prove near-optimal set-multilinear formula lower bounds (building upon the initial ingenious construction given in [10] for the multilinear context), tailor the framework to our \sum smABP model, and use a more delicate concentration bound analysis in order to prove our results.

An *arc-partition* is a special kind of symmetric word w from $\{-1, 1\}^d$: we will now describe a distribution over $\{-1, 1\}^d$; the words that will have positive probability of being obtained in this distribution will be called arc-partitions. The distribution is defined according to the following (iterative) sampling algorithm. Position the d variable sets on a cycle with d nodes so that there is an edge between i and $i + 1$ modulo d . Start with the arc $[L_1, R_1] = \{1, 2\}$ (an arc is a connected path on the cycle). At step $t > 1$ of the process, maintain a partition of the arc $[L_t, R_t]$. “Grow” this partition by first picking a pair uniformly at random out of the three possible pairs $\{L_t - 2, L_t - 1\}$, $\{L_t - 1, R_t + 1\}$, $\{R_t + 1, R_t + 2\}$, and then choosing a labelling (or partition) Π on this pair i.e., assigning one of them “+1” and the other “-1” uniformly at random. After $d/2$ steps, we have chosen a partition (i.e., a word w from $\{-1, 1\}^d$) of the d variable sets into two disjoint, equal-size sets of variables \mathcal{P} and \mathcal{N} . It is known from [26] that there exist set-multilinear polynomials $G_{n,d}$ (as defined in Section 3.4) that are *arc-full-rank* i.e., $\mathcal{M}_w(G_{n,d})$ is full-rank for every arc-partition w . Analogous to the proofs of Theorems 1.6 and 1.4, we establish our \sum smABP lower bounds by showing that with high probability, every \sum smABP has an appropriately large rank deficit with respect to the arc-partition distribution. However, as we now briefly explain, this analysis turns out to be significantly more intricate.

Similar to the analysis as in the VP case, we partition an ordered smABP A into q pieces of size r each, and write the polynomial that it computes as a sum of at most s^q terms. Again, the task is to show that an arc-partition w exhibits a large total bias across the q pieces: more precisely, we show that if the pieces are labelled as S_1, \dots, S_q , then with probability exponentially close to 1, the sum $\sum_{i=1}^q |w_{S_i}|$ (i.e., the total bias of w across these pieces) is $\Omega(qr^\varepsilon)$, which is polynomially large in d for an appropriate setting of q, r . This then yields the desired rank deficit similar to the VP analysis (albeit with mildly worse parameters).

The bias lower bound is established in the following sequence of steps:

- View the partition (S_1, \dots, S_q) of $[d]$ as a fixed “coloring” of the latter. We say that a *pair* – as sampled in the construction of an arc-partition described above – “violates” a color S if exactly one of the elements of the pair is colored by the set S . Then, we show that with probability exponentially close to 1, “many” colors must have “many” violations: more precisely, that at least a constant fraction of the colors (i.e., $\Omega(q)$ many) have at least $r^{2\varepsilon}$ many violations each (for some small constant $\varepsilon > 0$). Such a “many violations” lemma is also established in [26] in the context of proving set-multilinear formula lower bounds. We show that this lemma, in fact, holds for a much wider range of parameters than was previously known; this extension is indeed necessary for our use. The proof of this strengthened many violations lemma is deferred to the appendix.
- We then use the strengthened many violations lemma to argue that even though w is not chosen uniformly at random and as such, its coordinates are not truly independent, it possesses “enough” inherent independence that a similar concentration bound as in the VP analysis is applicable. More precisely, we show that with high probability, there is an ordering of a set of $\Omega(q)$ colors such that each such color has at least $r^{2\varepsilon}$ violations and a more nuanced application of standard concentration bounds shows that w exhibits a total bias of at least $\Omega(qr^\varepsilon)$.

2 Relative Rank and its Properties

We first describe the notation that we need to define the measures that we use to prove our results described in Section 1.3. Instead of directly working with the rank of the partial derivative matrix, we work with the following normalized form.

► **Definition 2.1.** Let $w = (w_1, w_2, \dots, w_d)$ be a tuple (or word) of non-zero real numbers. For a subset $S \subseteq [t]$, we shall refer to the sum $\sum_{i \in S} w_i$ by w_S , and by $w|_S$, we will refer to the tuple obtained by considering only the elements of w that are indexed by S . Given a word $w = (w_1, \dots, w_d)$, we denote by $\overline{X}(w)$ a tuple of d sets of variables $(X(w_1), \dots, X(w_d))$ where $|X(w_i)| = 2^{|w_i|}$.⁸ We denote by $\mathbb{F}_{\text{sm}}[\mathcal{T}]$ the set of set-multilinear polynomials over the tuple of sets of variables \mathcal{T} .

► **Definition 2.2** (Relative Rank Measure of [27]). Let $\overline{X} = (X_1, \dots, X_d)$ be a tuple of sets of variables such that $|X_i| = n_i$ and let $f \in \mathbb{F}_{\text{sm}}[\overline{X}]$. Let $w = (w_1, w_2, \dots, w_d)$ be a tuple (or word) of non-zero real numbers such that $2^{|w_i|} = n_i$ for all $i \in [d]$. Corresponding to a word w , define $\mathcal{P}_w := \{i \mid w_i > 0\}$ and $\mathcal{N}_w := \{i \mid w_i < 0\}$. Let $\mathcal{M}_w^{\mathcal{P}}$ be the set of all set-multilinear monomials over the subset of the variable sets X_1, X_2, \dots, X_d precisely indexed by \mathcal{P}_w , and similarly let $\mathcal{M}_w^{\mathcal{N}}$ be the set of all set-multilinear monomials over these variable sets indexed by \mathcal{N}_w .

Define the ‘partial derivative matrix’ matrix $\mathcal{M}_w(f)$ whose rows are indexed by the elements of $\mathcal{M}_w^{\mathcal{P}}$ and columns indexed by the elements of $\mathcal{M}_w^{\mathcal{N}}$ as follows: the entry of this matrix corresponding to a row m_1 and a column m_2 is the coefficient of the monomial $m_1 \cdot m_2$ in f . We define

$$\text{relrk}_w(f) := \frac{\text{rank}(\mathcal{M}_w(f))}{\sqrt{|\mathcal{M}_w^{\mathcal{P}}| \cdot |\mathcal{M}_w^{\mathcal{N}}|}} = \frac{\text{rank}(\mathcal{M}_w(f))}{2^{\frac{1}{2} \sum_{i \in [d]} |w_i|}}.$$

The following is a simple result that establishes various useful properties of the relative rank measure.

▷ **Claim 2.3** ([27]).

1. (Imbalance) Say $f \in \mathbb{F}_{\text{sm}}[\overline{X}(w)]$. Then, $\text{relrk}_w(f) \leq 2^{-|w_{[d]}|/2}$.
2. (Sub-additivity) If $f, g \in \mathbb{F}_{\text{sm}}[\overline{X}(w)]$, then $\text{relrk}_w(f + g) \leq \text{relrk}_w(f) + \text{relrk}_w(g)$.
3. (Multiplicativity) Say $f = f_1 f_2 \cdots f_t$ and assume that for each $i \in [t]$, $f_i \in \mathbb{F}_{\text{sm}}[\overline{X}(w|_{S_i})]$, where (S_1, \dots, S_t) is a partition of $[d]$. Then

$$\text{relrk}_w(f) = \prod_{i \in [t]} \text{relrk}_{w|_{S_i}}(f_i).$$

3 The Hard Polynomial

We now describe the different hard polynomials we use for our results.

3.1 Inner Product Gadget

The following observation is used crucially to construct the hard polynomials in VP as well as VBP.

⁸ In particular, $2^{|w_i|} \in \mathbb{N}$.

► **Observation 3.1** ([26]). Let $n = 2^k$ and $X_1 = \{x_{1,1}, \dots, x_{1,n}\}$ and $X_2 = \{x_{2,1}, \dots, x_{2,n}\}$ be two disjoint sets of variables. Then, for any symmetric word $w \in \{k, -k\}^2$ (i.e., where $w_1 + w_2 = 0$) and for the inner product “gadget” $f = X_1 \cdot X_2 = \sum_{i=1}^n x_{1,i}x_{2,i}$, $\text{relrk}_w(f) = 1$ i.e., $\mathcal{M}_w(f)$ is full-rank.

3.2 A Hard Set-multilinear Polynomial in VNP

As is done in previous lower bounds using the NW polynomials (for example, see [22]), we will identify the set of the first n integers as elements of ${}_n$ via an arbitrary correspondence $\phi : [n] \rightarrow {}_n$. If $f(z) \in {}_n[z]$ is a univariate polynomial, then we abuse notation to let $f(i)$ denote the evaluation of f at the i -th field element via the above correspondence i.e., $f(i) := \phi^{-1}(f(\phi(i)))$. To simplify the exposition, in the following definition, we will omit the correspondence ϕ and identify a variable $x_{i,j}$ by the point $(\phi(i), \phi(j)) \in {}_n \times {}_n$.

► **Definition 3.2** (Nisan-Wigderson Polynomials). For a prime power n , let ${}_n$ be a field of size n . For an integer $d \leq n$ and the set X of nd variables $\{x_{i,j} : i \in [n], j \in [d]\}$, we define the degree d homogeneous polynomial $NW_{n,d}$ over any field as

$$NW_{n,d}(X) = \sum_{\substack{f(z) \in {}_n[z] \\ \deg(f) < d/2}} \prod_{j \in [d]} x_{f(j),j}.$$

▷ **Claim 3.3** ([25]). For an integer $n = 2^k$ and $d \leq n$, let $w \in \{k, -k\}^d$ with $w_{[d]} = 0$. Then $\text{relrk}_w(NW_{n,d}) = 1$ i.e., $\mathcal{M}_w(NW_{n,d})$ has full rank.

Proof. Fix $n = 2^k$ and d , so that we can also write NW for $NW_{n,d}$, and let $n' = d/2$. The condition on w implies that $|\mathcal{P}_w| = |\mathcal{N}_w| = n'$. Observe that $\mathcal{M}_w(NW)$ is a square matrix of dimension $|\mathcal{M}_w^P| = |\mathcal{M}_w^N| = n^{n'}$. Consider a row of $\mathcal{M}_w(NW)$ indexed by a monomial $m_1 = x_{i_1, j_1} \cdots x_{i_{n'}, j_{n'}} \in \mathcal{M}_w^P$. m_1 can be thought of as a map from $S = \{j_1, \dots, j_{n'}\}$ to ${}_n$ which sends j_ℓ to i_ℓ for each $\ell \in [n']$. Next, by interpolating the pairs $(j_1, i_1), \dots, (j_{n'}, i_{n'})$, we know that there exists a unique polynomial $f(z) \in {}_n(z)$ of degree $< n'$ for which $f(j_\ell) = i_\ell$ for each $\ell \in [n']$. As a consequence, there is a unique “extension” of the monomial $x_{i_1, j_1} \cdots x_{i_{n'}, j_{n'}}$ that appears as a term in NW , which is precisely $m_1 \cdot \prod_{j \in \mathcal{N}_w} x_{f(j), j}$. Therefore, all but one of the entries in the row corresponding to m_1 must be zero, and the remaining entry must be 1. Applying the same argument to the columns of $\mathcal{M}_w(NW)$, we deduce that $\mathcal{M}_w(NW)$ is a permutation matrix, and so has full rank. ◁

3.3 A Hard Set-multilinear Polynomial in VP

Let d be an even integer and let $X = (X_1, \dots, X_d)$ be a collection of sets of variables where each $|X_i| = n$, and similarly, let $Y = (Y_1, \dots, Y_d)$ be a distinct collection of sets of variables where each $|Y_i| = n$. We shall refer to the Y -variables as the *auxiliary* variables. For i and $j \in \{1, \dots, d\}$, let $X_i \cdot X_j$ denote the inner-product quadratic form $\sum_{k=1}^n x_{ik}x_{jk}$. Here, we shall assume that $X_i = \{x_{i,1}, \dots, x_{i,n}\}$ and $Y_i = \{y_{i,1}, \dots, y_{i,n}\}$.

For two integers $i \in \mathbb{N}$ and $j \in \mathbb{N}$, we denote $[i, j] = \{k \in \mathbb{N} : i \leq k \text{ and } k \leq j\}$ and call such a set an *interval*. For every interval $[i, j] \subseteq [d]$, we define a polynomial $f_{i,j}(X, Y) \in \mathbb{F}_{\text{sm}}[X_i, \dots, X_j, Y_i, \dots, Y_j]$ as follows:

$$f_{i,j} = \begin{cases} y_{i,j}y_{j,i}(X_i \cdot X_j) & \text{if } j = i + 1 \\ 0 & \text{if } j - i \text{ is even} \\ y_{i,j}y_{j,i}(X_i \cdot X_j) \cdot f_{i+1, j-1} + \sum_{r=i+1}^{j-1} f_{i,r}f_{r+1, j} & \text{otherwise} \end{cases}$$

These $f_{i,j}$ in present form were defined in [26], but were in turn inspired from an earlier work of Raz and Yehudayoff ([35]) in the multilinear context. [26] shows that they have the following full-rank property that will be instrumental for us.

► **Lemma 3.4** ([26]). *Let $n = 2^k$ and $d \leq n$ be an even integer. Over any field of characteristic zero, the polynomial $F_{n,d} = f_{1,d} \in \mathbb{F}_{\text{sm}}[X, Y]$ as defined above satisfies the following: For any $w \in \{-k, k\}^d$ with $w_{[d]} = 0$, $\mathcal{M}_w(F_{n,d})$ is full-rank when viewed as a matrix over the field (Y) , the field of rational functions over the Y variables.*

3.4 A Hard Set-Multilinear Polynomial in VBP

3.4.1 Arc-partition Measure Description

This subsection is adapted from Section 2 of [10]. Let $n = 2^k$, $d \leq n$ be an even integer, and let $X = (X_1, X_2, \dots, X_d)$ be a collection of disjoint sets of n variables each. An *arc-partition* will be a special kind of *symmetric* word $w \in \{-k, k\}^d$ (i.e., a one-to-one map Π from X to $\{-k, k\}^d$). For the purpose of this subsection, the reader can even choose to think of the alphabet of w as $\{-1, 1\}$ (i.e., one “positive” and one “negative” value) – we use $k, -k$ only to remain consistent with Definition 2.2.

Identify X with the set $\{1, 2, \dots, d\}$ in the natural way. Consider the d -cycle graph, i.e., the graph with nodes $\{1, 2, \dots, d\}$ and edges between i and $i + 1$ modulo d . For two nodes $i \neq j$ in the d -cycle, denote by $[i, j]$ the arc between i, j , that is, the set of nodes on the path $\{i, i + 1, \dots, j - 1, j\}$ from i to j in d -cycle. First, define a distribution \mathcal{D}_P on a family of pairings (a list of disjoint pairs of nodes in the cycle) as follows. A random pairing is constructed in $d/2$ steps. At the end of step $t \in [d/2]$, we shall have a pairing (P_1, \dots, P_t) of the arc $[L_t, R_t]$. The size of $[L_t, R_t]$ is always $2t$. The first pairing contains only $P_1 = \{L_1, R_1\}$ with $L_1 = 1$ and $R_1 = 2$. Given (P_1, \dots, P_t) and $[L_t, R_t]$, define the random pair P_{t+1} (independently of previous choices) by

$$P_{t+1} = \begin{cases} \{L_t - 2, L_t - 1\} & \text{with probability } 1/3 \\ \{L_t - 1, R_t + 1\} & \text{with probability } 1/3 \\ \{R_t + 1, R_t + 2\} & \text{with probability } 1/3 \end{cases}$$

Define

$$[L_{t+1}, R_{t+1}] = [L_t, R_t] \cup P_{t+1}.$$

So, L_{t+1} is either $L_t - 2$, $L_t - 1$ or L_t , each value is obtained with probability $1/3$, and similarly (but not independently) for R_{t+1} .

The final pairing is $P = (P_1, P_2, \dots, P_{d/2})$. Denote by $P \sim \mathcal{D}_P$ a pairing distributed according to \mathcal{D}_P .

Once a pairing P has been obtained, a word $w \in \{-k, k\}^d$ is obtained by simply randomly assigning $+k$ and $-k$ to the indices of any pair P_i . More formally, for every $t \in [d/2]$, if $P_t = \{i_t, j_t\}$, let with probability $1/2$, independently of all other choices,

$$w_{i_t} = +k \text{ and } w_{j_t} = -k,$$

and with probability $1/2$,

$$w_{i_t} = -k \text{ and } w_{j_t} = +k.$$

Denote by $w \sim \mathcal{D}$ a word in $\{-1, 1\}^n$ that is sampled using this procedure. We call such a word an *arc-partition*. For a pair $P_t = \{i_t, j_t\}$, we refer to i_t and j_t as *partners*.

► **Definition 3.5** (Arc-full-rank). *We say that a polynomial f that is set-multilinear over $X = (X_1, \dots, X_d)$ is **arc-full-rank** if for every arc-partition $w \in \{-k, k\}^d$, $\text{relrk}_w(f) = 1$.*

3.4.2 Construction of an Arc-full-rank Polynomial

Below, we describe a simple construction of a polynomial sized ABP that computes an arc-full-rank set-multilinear polynomial. The high-level idea is to construct an ABP in which every path between start-node and end-node corresponds to a specific execution of the random process which samples arc-partitions. Each node in the ABP corresponds to an arc $[L, R]$, which sends an edge to each of the nodes $[L - 2, R]$, $[L - 1, R + 1]$ and $[L, R + 2]$. The edges have specially chosen labels that help guarantee full rank with respect to every arc-partition. For simplicity of presentation, we allow the edges of the program to be labeled by degree four set-multilinear polynomial polynomials over the corresponding subset of the variable partition. This assumption can be easily removed by replacing each edge with a polynomial-sized ABP computing the corresponding degree four polynomial.

Formally, the nodes of the program are even-size arcs in the d -cycle, d an even integer. The start-node of the program is the empty arc \emptyset and the end-node is the whole cycle $[d]$ (both are “special” arcs). Let $X = (X_1, \dots, X_d)$ be a collection of sets of variables where each $|X_i| = n$, and similarly, let $Y = (Y_1, \dots, Y_d)$ be a distinct collection of sets of variables where each $|Y_i| = n$ (we shall refer to the Y -variables as *auxiliary* variables). For i and j in $\{1, \dots, d\}$, let $X_i \cdot X_j$ denote the inner-product quadratic form $\sum_{k=1}^n x_{ik}x_{jk}$. Here, we shall assume that $X_i = \{x_{i,1}, \dots, x_{i,n}\}$ and $Y_i = \{y_{i,1}, \dots, y_{i,n}\}$.

Construct the branching program by connecting a node/arc of size $2t$ to three nodes/arcs of size $2t + 2$. For $t = 1$, there is just one node $[1, 2]$, and the edge from start-node to it is labeled $y_{1,2}y_{2,1}(X_0 \cdot X_1)$. For $t > 1$, the node $[L, R] \supset [1, 2]$ of size $2t < d$ is connected to the three nodes: $[L - 2, R]$, $[L - 1, R + 1]$, and $[L, R + 2]$. (It may be the case that the three nodes are the end-node.) The edge labeling is:

- The edge between $[L, R]$ and $[L - 2, R]$ is labeled $y_{L-2,L-1}y_{L-1,L-2}(X_{L-2} \cdot X_{L-1})$.
- The edge between $[L, R]$ and $[L - 1, R + 1]$ is labeled $y_{L-1,R+1}y_{R+1,L-1}(X_{L-1} \cdot X_{R+1})$.
- The edge between $[L, R]$ and $[L, R + 2]$ is labeled $y_{R+1,R+2}y_{R+2,R+1}(X_{R+1} \cdot X_{R+2})$.

Consider the ABP thus described, and the polynomial $G_{n,d}$ it computes. For every path γ from start-node to end-node in the ABP, the list of edges along γ yields a pairing P ; every edge e in γ corresponds to a pair $P_e = \{i_e, j_e\}$ of nodes in d -cycle. Thus,

$$G_{n,d} = \sum_{\gamma} \prod_{e=\{i_e, j_e\} \in \gamma} y_{i_e, j_e} y_{j_e, i_e} \cdot (X_{i_e} \cdot X_{j_e}). \quad (1)$$

where the sum is over all paths γ from start-node to end-node.

► **Remark 3.6.** There is in fact a one-to-one correspondence between pairings P and such paths γ (this follows by induction on t). Note that this is true only because pairings are tuples i.e., they are *ordered* by definition. Otherwise, it is of course still possible to obtain the same *set* of pairs in a given pairing using multiple different orderings. The sum defining $G_{n,d}$ can be thought of, therefore, as over pairings P .

The following statement summarizes the main useful property of $G_{n,d}$.

► **Lemma 3.7** ([26]). *Over any field of characteristic zero, the polynomial $G_{n,d}$ defined above is arc-full-rank as a set-multilinear polynomial in the variables X over the field (Y) of rational functions in Y .*

Proof. Let $w \sim \mathcal{D}$ be an arc-partition. We want to show that $\mathcal{M}_w(G_{n,d})$ has full rank. The arc-partition w is defined from a pairing $P = (P_1, \dots, P_{d/2})$ (though as discussed in Remark 3.6, there could be multiple such P). The pairing P corresponds to a path γ from start-node to end-node. Consider the polynomial f that is obtained by setting every $y_{i,j} = y_{j,i} = 0$ in F such that $\{i, j\}$ is not a pair in P , and setting every $y_{i,j} = y_{j,i} = 1$ for every pair $\{i, j\}$ in P . Then, it is easy to see that the only terms that survive in Equation 1 correspond to paths (and in turn, pairings) which have the same underlying *set* of pairs as P . As a consequence, f is simply some non-zero constant times a polynomial which is full-rank (recall Observation 3.1). $M_w(f)$ being full rank then implies that $M_w(G_{n,d})$ is also full-rank. ◀

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 Satharishi, 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 Vikraman Arvind and S. Raja. Some lower bound results for set-multilinear arithmetic computations. *Chic. J. Theor. Comput. Sci.*, 2016, 2016. URL: <http://cjtc.cs.uchicago.edu/articles/2016/6/contents.html>.
- 4 C. S. Bhargav, Sagnik Dutta, and Nitin Saxena. Improved lower bound, and proof barrier, for constant depth algebraic circuits. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPICs*, pages 18:1–18:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.MFCS.2022.18.
- 5 C.S. Bhargav, Prateek Dwivedi, and Nitin Saxena. Lower bounds for the sum of small-size algebraic branching programs. *To appear in the proceedings of the Annual Conference on Theory and Applications of Models of Computation*, 2024. URL: <https://www.cse.iitk.ac.in/users/nitin/papers/sumRO.pdf>.
- 6 Vishwas Bhargava and Sumanta Ghosh. Improved hitting set for orbit of roabps. *Comput. Complex.*, 31(2):15, 2022. doi:10.1007/S00037-022-00230-9.
- 7 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.
- 8 Peter Bürgisser. Cook’s versus valiant’s hypothesis. *Theor. Comput. Sci.*, 235(1):71–88, 2000. doi:10.1016/S0304-3975(99)00183-8.
- 9 Prerona Chatterjee, Mrinal Kumar, Adrian She, and Ben Lee Volk. Quadratic lower bounds for algebraic branching programs and formulas. *Comput. Complex.*, 31(2):8, 2022. doi:10.1007/S00037-022-00223-8.
- 10 Zeev Dvir, Guillaume Malod, Sylvain Perifel, and Amir Yehudayoff. Separating multilinear branching programs and formulas. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 615–624. ACM, 2012. doi:10.1145/2213977.2214034.
- 11 Michael A. Forbes, Ramprasad Satharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In David B. Shmoys, editor, *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.
- 12 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, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.34.

- 13 Purnata Ghosal and B. V. Raghavendra Rao. Limitations of sums of bounded read formulas and abps. In Rahul Santhanam and Daniil Musatov, editors, *Computer Science - Theory and Applications - 16th International Computer Science Symposium in Russia, CSR 2021, Sochi, Russia, June 28 - July 2, 2021, Proceedings*, volume 12730 of *Lecture Notes in Computer Science*, pages 147–169. Springer, 2021. doi:10.1007/978-3-030-79416-3_9.
- 14 Zeyu Guo and Rohit Gurjar. Improved explicit hitting-sets for roabps. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
- 15 Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Identity testing for constant-width, and any-order, read-once oblivious arithmetic branching programs. *Theory Comput.*, 13(1):1–21, 2017. doi:10.4086/TOC.2017.V013A002.
- 16 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.
- 17 Rohit Gurjar and Ben Lee Volk. Pseudorandom bits for oblivious branching programs. *ACM Trans. Comput. Theory*, 12(2):8:1–8:12, 2020. doi:10.1145/3378663.
- 18 Maurice J. Jansen. Lower bounds for syntactically multilinear algebraic branching programs. In Edward Ochmanski and Jerzy Tyszkiewicz, editors, *Mathematical Foundations of Computer Science 2008, 33rd International Symposium, MFCS 2008, Torun, Poland, August 25-29, 2008, Proceedings*, volume 5162 of *Lecture Notes in Computer Science*, pages 407–418. Springer, 2008. doi:10.1007/978-3-540-85238-4_33.
- 19 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *Electron. Colloquium Comput. Complex.*, TR12-081, 2012. arXiv:TR12-081.
- 20 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. *SIAM J. Comput.*, 46(1):307–335, 2017. doi:10.1137/151002423.
- 21 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.
- 22 Neeraj Kayal, Chandan Saha, and Ramprasad Satharishi. A super-polynomial lower bound for regular arithmetic formulas. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 146–153. ACM, 2014. doi:10.1145/2591796.2591847.
- 23 Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. An almost cubic lower bound for depth three arithmetic circuits. 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 33:1–33:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICs.ICALP.2016.33.
- 24 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. *SIAM J. Comput.*, 46(1):336–387, 2017. doi:10.1137/140999335.
- 25 Deepanshu Kush and Shubhangi Saraf. Improved low-depth set-multilinear circuit lower bounds. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 38:1–38:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.CCC.2022.38.
- 26 Deepanshu Kush and Shubhangi Saraf. Near-optimal set-multilinear formula lower bounds. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*, volume 264 of *LIPICs*, pages 15:1–15:33. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.CCC.2023.15.
- 27 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 804–814. IEEE, 2021. doi:10.1109/FOCS52979.2021.00083.

- 28 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. On the partial derivative method applied to lopsided set-multilinear polynomials. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 32:1–32:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.CCC.2022.32.
- 29 Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *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.
- 30 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Comput. Complex.*, 6(3):217–234, 1997. doi:10.1007/BF01294256.
- 31 C. Ramya and B. V. Raghavendra Rao. Lower bounds for special cases of syntactic multilinear abps. *Theor. Comput. Sci.*, 809:1–20, 2020. doi:10.1016/J.TCS.2019.10.047.
- 32 Ran Raz. Separation of multilinear circuit and formula size. *Theory Comput.*, 2(6):121–135, 2006. doi:10.4086/toc.2006.v002a006.
- 33 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *J. ACM*, 60(6):40:1–40:15, 2013. doi:10.1145/2535928.
- 34 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.
- 35 Ran Raz and Amir Yehudayoff. Balancing syntactically multilinear arithmetic circuits. *Comput. Complex.*, 17(4):515–535, 2008. doi:10.1007/S00037-008-0254-0.
- 36 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Comput. Complex.*, 18(2):171–207, 2009. doi:10.1007/s00037-009-0270-8.
- 37 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. *Github Survey*, 2015. URL: <https://github.com/dasarpmar/lowerbounds-survey>.
- 38 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2010. doi:10.1561/04000000039.
- 39 Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. 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 416–425. ACM, 2022. doi:10.1145/3519935.3520044.
- 40 L. G. Valiant. Completeness classes in algebra. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 249–261, New York, NY, USA, 1979. Association for Computing Machinery. doi:10.1145/800135.804419.

Public-Key Pseudoentanglement and the Hardness of Learning Ground State Entanglement Structure

Adam Bouland ✉

Department of Computer Science, Stanford University, CA, USA

Bill Fefferman ✉

Department of Computer Science, University of Chicago, IL, USA

Soumik Ghosh ✉

Department of Computer Science, University of Chicago, IL, USA

Tony Metger ✉

ETH Zürich, Switzerland

Umesh Vazirani ✉

Department of Electrical Engineering and Computer Sciences,
University of California, Berkeley, CA, USA

Chenyi Zhang ✉

Department of Computer Science, Stanford University, CA, USA

Zixin Zhou ✉

Department of Computer Science, Stanford University, CA, USA

Abstract

Given a local Hamiltonian, how difficult is it to determine the entanglement structure of its ground state? We show that this problem is computationally intractable even if one is only trying to decide if the ground state is volume-law vs near area-law entangled. We prove this by constructing strong forms of pseudoentanglement in a public-key setting, where the circuits used to prepare the states are public knowledge. In particular, we construct two families of quantum circuits which produce volume-law vs near area-law entangled states, but nonetheless the classical descriptions of the circuits are indistinguishable under the Learning with Errors (LWE) assumption. Indistinguishability of the circuits then allows us to translate our construction to Hamiltonians. Our work opens new directions in Hamiltonian complexity, for example whether it is difficult to learn certain phases of matter.

2012 ACM Subject Classification Theory of computation → Quantum computation theory; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Quantum computing, Quantum complexity theory, entanglement

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.21

Related Version *Full Version:* <https://arxiv.org/abs/2311.12017>

Funding B.F. and S.G. acknowledge support from AFOSR (FA9550-21-1-0008). This material is based upon work partially supported by the National Science Foundation under Grant CCF-2044923 (CAREER) and by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers (Q-NEXT). This research was also supported in part by the National Science Foundation under Grant No. NSF PHY-1748958. A.B., B.F., C.Z., and Z.Z. were supported in part by the DOE QuantISED grant DE-SC0020360. A.B. and C.Z. were supported in part by the U.S. DOE Office of Science under Award Number DE-SC0020266. A.B. was supported in part by the AFOSR under grant FA9550-21-1-0392. C.Z. was supported in part by the Shoucheng Zhang graduate fellowship. T.M. acknowledges support from SNSF Grant No. 200021_188541, the ETH Zurich Quantum Center, and an ETH Doc.Mobility Fellowship. U.V. was supported in part by DOE NQISRC QSA grant FP00010905, NSF QLCI Grant No. 2016245, and MURI Grant FA9550-18-1-0161.



© Adam Bouland, Bill Fefferman, Soumik Ghosh, Tony Metger,
Umesh Vazirani, Chenyi Zhang, and Zixin Zhou;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 21; pp. 21:1–21:23



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Acknowledgements We thank Rotem Arnon-Friedman, Jordan Docter, Tudor Giurgica-Tiron, Andru Gheorghiu, Hsin-Yuan Huang, Vinod Vaikuntanathan, and Thomas Vidick for helpful discussions. We thank the Simons Institute for the Theory of Computing, where some of this work was conducted.

1 Introduction

The central problem in Hamiltonian complexity is to understand the structure of ground states of local Hamiltonians and the difficulty of learning properties of them, e.g. [20, 29, 24, 6, 5]. In this work we study the following question: given a local Hamiltonian H , how difficult is it to learn about the entanglement structure of its ground state? For example, given H , can you tell if its ground state is area-law or volume-law entangled? We call such questions about the qualitative features of the entanglement structure a *Learning Ground State Entanglement Structure* (LGSES) problem. This problem is related to both condensed matter physics – where ground state entanglement structure is a central object of study – and may also shed light on questions in quantum gravity regarding how entanglement could possibly be dual to other physical quantities [12, 18].

Whereas most effort in many-body physics has been directed towards the positive side of this question, i.e. finding conditions under which properties of the ground state can be efficiently learnt or computed (see e.g. [15, 24, 21]), here our goal is to prove hardness results for the LGSES problem from cryptographic assumptions. This explores the limitations that any algorithm for such problems must run up against. In other words, hardness results for the LGSES problem point to qualitative features of Hamiltonian ground states that are inherently computationally intractable to compute.

To prove computational hardness results for the LGSES problem, we will relate it to a notion called *pseudoentanglement*, a term recently introduced in [1]. Informally, pseudoentangled state ensembles consist of low-entanglement states that masquerade as high-entanglement states to computationally bounded observers; in other words, a pseudoentangled state only looks like a high-entanglement state to bounded observers, whereas its actual (information-theoretic) entanglement is low. The main result of [1] is that it is possible to create pseudoentangled states which hide vast differences in entanglement. More concretely, they construct two ensembles of quantum states, Ψ^{high} and Ψ^{low} , such that any state $|\psi\rangle \in \Psi^{\text{high}}$ has high entanglement entropy across any bipartition of the qubits and states in Ψ^{low} have low entanglement, but any computationally bounded quantum algorithm that receives a state from $\Psi^{\text{high}} \cup \Psi^{\text{low}}$ cannot tell which kind of state it received.

This notion of pseudoentanglement is interesting in its own right and has been used for various applications in [1], but its relevance to Hamiltonian complexity is unclear. This is because the settings are inherently different: the notion of pseudoentanglement in [1] involves a *quantum* input, namely copies of the relevant quantum states, being given to the distinguisher. If we tried to translate this into a setting involving Hamiltonian ground states, we would end up in a model where we study the properties of Hamiltonian ground states given quantum copies of these ground states, but without knowing the actual Hamiltonian itself. In contrast, in Hamiltonian complexity we assume that we know a classical description of the Hamiltonian under consideration and would like to determine properties of its ground state.

Therefore, if we want to use some notion of pseudoentanglement to prove hardness results for the LGSES problem, we need to consider a model where the distinguisher is not just given quantum copies of the high- or low-entanglement states, but rather an efficient classical description of these states. This leads to a different kind of pseudoentanglement, which we

call *public-key* pseudoentanglement since the description of the states (the “key”) can be made public. This notion is already implicit in earlier work by Gheorghiu and Hoban [18], who gave a construction of public-key pseudoentanglement that we discuss in more detail below.

► **Definition 1** (Public-key pseudoentanglement (implicit in [18])). *Two ensembles of n -qubit $\text{poly}(n)$ -gate quantum circuits $\{C_k\}, \{C'_k\}$ are public-key pseudoentangled with entanglement gap $f(n)$ vs $g(n)$ if they are computationally indistinguishable to poly-time quantum algorithms, and yet with high probability over the ensembles, $C_k|0^n\rangle$ has entanglement $\geq f(n)$ and $C'_k|0^n\rangle$ has entanglement $\leq g(n)$ across one or more cuts of the system.*

The key difference between this definition and the one in [1] is that here a classical description of the circuit used to prepare the states is given as input to the distinguisher, whereas in [1] the distinguisher only receives copies of the output state of the circuit. A distinguisher can therefore not only prepare copies of the output state, but also analyse the classical description of the circuits directly to gain additional information, making it harder to “hide” the entanglement of pseudoentangled states. Hence, public-key pseudoentanglement is a much stronger notion than that in [1], which we will call *private-key* pseudoentanglement as the circuits are hidden.

There are generally two features we care about in a pseudoentanglement construction: the entanglement gap, which we want to be as large as possible in order to hide as much entanglement as possible, and the set of cuts across which this entanglement gap holds. Informally, a more general set of cuts (i.e. bipartitions of the qubits) across which the entanglement gap holds corresponds to hiding more qualitative information about the entanglement structure of the state; this is what we are most interested in for the LGSES problem.

The pioneering work of Gheorghiu and Hoban gave the first pseudoentanglement construction with entanglement gap n vs $n - \Omega(1)$ across a single cut based on the Learning With Errors (LWE) assumption [18]. Using this they showed that it is difficult to learn *fine-grained* properties of the ground state entanglement – namely if the ground state has entanglement n vs $n - \Omega(1)$ across a fixed cut. The basic idea is that if one passes the circuit to prepare pseudoentangled states through a modified Kitaev clock construction [23, 26], the ground state of the resulting Hamiltonian has the same entanglement properties as the output state of the circuit. We emphasize that the Kitaev clock Hamiltonian encodes the circuit used to prepare the state in plaintext. Consequently, this application necessarily requires a *public-key* construction if we aim to construct the hard instance via the Kitaev clock.

However, Gheorghiu and Hoban’s construction only suffices to prove hardness of detecting very small differences in the entanglement of the ground state across a single cut. In contrast, the LGSES problem asks about *qualitative* or *coarse-grained* features of the entanglement structure, as very fine-grained properties are usually not physically relevant. This raises the following question: is it possible to get a public-key pseudoentanglement construction that hides qualitatively different entanglement structures? This would lead to natural hardness statements for the LGSES problem.

1.1 Near area-law public-key pseudoentanglement

Our first result is to construct strong forms of public-key pseudoentanglement from LWE. In particular we show it is possible to hide 1D near area-law vs volume-law entanglement.

► **Theorem 2** (Volume vs area-law public-key pseudoentanglement (informal)). *Assuming subexponential-time hardness of LWE, there exist public-key pseudoentangled ensembles with volume-law vs near area-law entanglement when the qubits are arranged on a 1D line.*

That is, in one case the states have entanglement $\Omega(\min(k, n - k))$ across any division of the qubits into k vs $n - k$ qubits (volume-law), and in the other case the entanglement of any cut is $\leq |A|\text{polylog}(n)$ where $|A|$ is the area of the cut when the qubits are arranged on a 1D line (i.e. the number of times the cut crosses the 1D line). We call this *near area-law entangled*. This is optimal because if the polylog factor were changed to a log, then these states would be efficiently distinguishable from one another by standard Matrix Product State learning algorithms [15, 24]. Hiding such qualitatively different entanglement structures requires a completely different construction from [18]. We note that while in Theorem 2 we assume subexponential-time hardness of LWE, we also show that the standard LWE assumption implies a similar result.¹ We will discuss the application of our result to Hamiltonian complexity shortly, after we present its proof sketch.

Proof sketch for single-cut pseudoentanglement. Let us first consider a single cut partitioning the qubits into sets A and B . In this case, the most natural states to consider are of the form $\sum_{x \in \{0,1\}^n} |x\rangle_A |h(x)\rangle_B$ for some function h . If one chooses h to be injective, then this state has entanglement entropy n across this cut; if one chooses h to be 2^k -to-1, then the entanglement is $n - k$. [18] used exactly these states with trapdoor claw-free functions from [13], which are functions that are either injective or 2-to-1, but the two cases are computationally hard to distinguish (given a description of the function) assuming the hardness of LWE. There are some additional subtleties arising from the fact that the trapdoor claw-free functions from [13] do not output numbers, but probability distributions, and are only approximately 2-to-1. We refer to [18] for a detailed analysis of this construction.

To increase the entanglement gap, we need to make the many-to-1 functions *more compressing*. The functions in [18, 13] additionally have a trapdoor; however, we observe that for pseudoentanglement, we can dispense with the trapdoor and only need so-called *lossy functions*: these are functions that are either injective or 2^k -to-1, but again the two kinds of functions are hard to distinguish. Starting from this observation, it turns out to be possible to combine the construction from [18] with ideas from a recent randomness generation protocol [25] to achieve an entanglement gap of n vs n^δ for any $\delta > 0$.²

The challenge with this approach based on [18] is that it appears fundamentally restricted to a single cut. However, our goal is to obtain pseudoentangled states with near area-law vs volume-law entanglement structure, which requires low entanglement across *exponentially many cuts* simultaneously. This would require controlling the entanglement not just between regions A and B , but also within these regions. With the approach of [18] it seems difficult to appropriately modify the state in register A without jeopardising the entanglement across the cut between A and B .

For this reason, we need a different kind of state that allows us to control the entanglement across all cuts simultaneously. It turns out that a useful class to consider are binary phase states as in [22, 14, 1], i.e., states of the form $\sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle$. Our approach will be

¹ In particular, the polylog correction factor to the area-law scaling is replaced by an n^ϵ correction, where ϵ can be any constant > 0 .

² We note that independently from and simultaneously to our work, Gheorghiu and Hoban updated their results to include a construction of this form, which achieves the aforementioned gap of n^δ vs n across a fixed cut with n qubits in one set and $\text{poly}(n)$ in the other.

as follows: we start from a phase state with high-entanglement across every cut. We will then modify this state (in a computationally undetectable way) to have low entanglement across some particular cut. This achieves essentially the same as the [18]-based construction above.³ However, crucially our “modification procedure” is iterable: this means that we can perform essentially the same entanglement reduction operation across many cuts in sequence and end up with a state with low entanglement across every cut.

We first describe the construction and proof for a single cut. For simplicity, let us first consider the cut between the first and second $n/2$ qubits. One can easily compute that the reduced density matrix of the first half of the state $\sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle$ is $\rho \propto TT^\top$, where $T_{ij} = (-1)^{f(i||j)}$. Here, $||$ denotes the concatenation of two strings, and $i, j \in \{0,1\}^{n/2}$ correspond to the first and second halves of the string x , i.e., T is the truth table of the function f written out in a matrix form. By a direct calculation, one can show an upper bound on the entanglement entropy of our phase state (i.e. the von Neumann entropy of ρ) in terms of the rank of T , and a lower bound on the entanglement entropy in terms of the Frobenius norm of TT^\top . Our strategy will therefore be to start from a “ T -matrix” for a high-entanglement state, and then perform one of two modifications in a computationally indistinguishable way: either modify T to reduce its rank to get low-entanglement states, or modify T in a way that does *not* reduce $\|TT^\top\|_2$ too much so that the entanglement of the states remains high, where $\|\cdot\|_2$ denotes the Frobenius norm, as defined in Section 2.1. These modified T -matrices then correspond to modified phase states, and our goal is to hide which of the two procedures we performed, even when handing out a classical description for preparing the corresponding states.

In [1] the idea is to apply a private key cryptographic hash function to replace rows of the matrix T with copies of other rows to reduce its rank. That is, we pick a phase function $f(x)$ which yields high entanglement [14], and then “whittle down” its entanglement by replacing $f(x)$ with $f(h(i) || j)$ where $i, j \in \{0,1\}^{n/2}$ and $h : \{0,1\}^{n/2} \rightarrow \{0,1\}^{n/2}$ is a 2^k -to-1 function. This reduces the number of distinct rows in T (each of which is now repeated many times) and correspondingly decreases the rank of T . As a result, this procedure lowers the entanglement of ρ by k . On the other hand, if we pick h to be a 1-to-1 function, we simply permute the rows of T and do not change $\|TT^\top\|_2$. To make this public-key, we need a way of applying a 2^k -to-1 or a 1-to-1 hash function to these phase states in such a way that it is hard to tell which one was applied, even when the state preparation circuit (and therefore the source code for h) is public.

At first sight, it may seem that we can use the same idea as above: take the lossy functions from [25] and use them to reduce the rank of T . However, the phase state construction is much less flexible than states of the form $\sum_x |x\rangle |h(x)\rangle$: for the latter, the codomain of h does not matter and we can use the functions h from [25], whose codomain are probability distributions over \mathbb{Z}_q^m for $m \gg n$. In contrast, for phase states we need lossy functions $h : \{0,1\}^{n/2} \rightarrow \{0,1\}^{n/2}$, i.e. the codomain has to be the same as the domain. This is a somewhat unusual requirement from a cryptographic perspective and forces us to use a custom construction of “imperfect” lossy functions based on LWE. Concretely, we first show how to create lossy functions mapping $\{0,1\}^{n/2} \rightarrow \{0,1\}^{\text{poly}(n)}$ which are *exactly* injective vs 2^k -to-1 (for sufficiently large k) with high probability. This is not yet what we need, as the codomain is exponentially larger than the domain. To fix this, we compose these

³ One advantage of this construction even for a single cut is that it allows an entanglement gap of n vs $\text{poly} \log n$ for an $O(n)$ -qubit state, whereas in the [18]-based construction the B -register had to have $\text{poly}(n)$ qubits for a comparable entanglement gap.

functions with pairwise independent hash functions which shrink the codomain size back to $2^{n/2}$. This can only make the 2^k -to-1 functions more compressing, which is to our benefit. However, this also introduces unwanted collisions for the injective functions which might break the high-entanglement case. To deal with this, we show that the repetition pattern this produces in the matrix T is sufficiently well-behaved that the corresponding states still have high entanglement. Intuitively, this holds because even though the “injective” functions or no longer actually injective, they are still pairwise independent, which ensures enough independence in the row repetition pattern of T to give a strong lower bound on the Frobenius norm $\|TT^\top\|_2$.

From single-cut to multi-cut pseudoentanglement. As we mentioned, the advantage of the above construction is that it can be extended to pseudoentanglement across all cuts simultaneously. We now explain how this extension works at a high level. A first observation is that to achieve pseudoentangled states with 1D near area law scaling, reducing the entanglement across all $n - 1$ contiguous cuts of the line to $O(\text{polylog}(n))$ suffices by strong subadditivity. Therefore, a natural approach is to perform a 1D sweep of the line, reducing the entanglement of the contiguous cuts one at a time. For reasons that will become apparent below, we perform this sweep right-to-left. The final phase function is then a complicated composition of n independent lossy functions and hash functions.

To show that this construction indeed achieves pseudoentanglement across all cuts simultaneously, we need to worry about two issues: firstly, for the low entanglement states we need to ensure that performing the entanglement reduction operation for a cut towards the left of the line does not inadvertently increase the entanglement across earlier cuts to the right, so that the low entanglement across those earlier cuts is preserved. Secondly, for the high entanglement states we need to make sure that the fact that we are using “imperfect” injective functions does not decrease the entanglement too much even after applying these functions across every cut.

The first concern is relatively easy to deal with due to the relationships between the “ T -matrices” for different cuts. To see this, consider a phase state $|\psi\rangle = \sum (-1)^{f(x)} |x\rangle$ on $n + m$ qubits on a line and let $T^{n|m} \in \{\pm 1\}^{2^n \times 2^m}$ be the T -matrix for the cut between the first n and last m qubits, i.e. $T_{ij}^{n|m} = (-1)^{f(i||j)}$ for $i \in \{0, 1\}^n, j \in \{0, 1\}^m$. After performing the entanglement reduction operation, this matrix will only have some smaller number R of distinct rows, each repeated many times. In the next step of the sweep (recalling that we move right to left), we consider the cut between the first $n - 1$ and last $m + 1$ qubits. We denote the corresponding T -matrix by $T^{n-1|m+1} \in \{\pm 1\}^{2^{n-1} \times 2^{m+1}}$. From the definition of the T -matrix, one can see that the first row of $T^{n-1|m+1}$ simply consists of the first two rows of $T^{n|m}$ stacked side by side. More generally, the i -th row of $T^{n-1|m+1}$ is simply the horizontal concatenation of rows $2i - 1$ and $2i$ from $T^{n|m}$. Suppose we now reduce the rank of $T^{n-1|m+1}$ by removing some rows and duplicating others. We then need to check that if we go back to the cut $n|m$, the resulting T -matrix (denoted $\tilde{T}^{n|m}$) still has rank at most R . This is the case since the rows of $\tilde{T}^{n|m}$ consist of the first and second parts of the rows of $T^{n-1|m+1}$; since the rank reduction only repeats, but does not modify, rows in $T^{n-1|m+1}$, every row in $\tilde{T}^{n|m}$ must have already appeared in $T^{n|m}$. As a result, the subsequent rank reduction for cut $n - 1|m + 1$ can only decrease, not increase, the rank of the T -matrix across $n|m$. It is not too hard to see that this argument generalises to any future rank reduction operation, not just the immediately subsequent one.

The second concern is more difficult to deal with. When applying this sweep with *approximately* injective functions, the entanglement is reduced slightly each time. The rightmost (first) cut in particular has its entanglement reduced n times, so even a tiny loss

could kill the entire construction. Perhaps surprisingly, we show that this is not the case, and the entanglement losses do not compound too badly. We show that different rows have different probabilities of being hashed together due to the structure of the sweep, and a careful accounting of this process reveals that not much entanglement is lost, even in the first cut. The analysis is somewhat technical and we refer to Section 3.3 for details. Finally, we note that while the construction we described here does not produce pseudorandom states ensembles (i.e. the families of pseudoentangled states, without the public key, are not necessarily pseudorandom states [22]), we can make a simple modification to our construction to ensure that this is the case.

1.2 Hardness of learning ground state entanglement structure

Our second result is to show that this public-key, area vs volume-law pseudoentanglement construction enables new applications in quantum Hamiltonian complexity. Because our public-key pseudoentanglement construction can hide qualitative features of the entanglement structure, we can show natural results for the hardness of the Learning Ground State Entanglement Structure (LGSES) problem for broad differences in entanglement structure. As we mentioned above, the main idea for turning pseudoentanglement constructions into hard instances of LGSES is to use a circuit-to-Hamiltonian construction on the state preparation circuit for the pseudoentangled state. There are a variety of circuit-to-Hamiltonian constructions and using these on our pseudoentangled states yields a variety of hardness statements for LGSES. In this paper, we consider three different constructions: a Kitaev clock construction with a binary clock, a Kitaev clock construction with a unary clock, and a customised version of a geometrically local 2D construction [3]. As we discuss in Section 1.4, an interesting open problem is whether a custom circuit-to-Hamiltonian construction that is focused purely on preserving entanglement structure (rather than QMA-hardness) can produce hard instances of the LGSES problem for more physically natural Hamiltonians.

Using a Kitaev clock construction with a binary clock [23], we get the following result (see Theorem 26 for the formal statement).

► **Theorem 3** (informal). *Assuming subexponential-time hardness of LWE, LGSES is intractable when the input Hamiltonian is $O(\log n)$ -local on n qubits, and the goal is to decide whether the ground state is volume-law or near area-law entangled for the qubits arranged on a 1D line.*

This result follows relatively straightforwardly from the standard Kitaev clock construction. There are only two issues that need to be addressed: firstly, the ground state of the Hamiltonian in the Kitaev clock construction is the history state of the circuit, not the output state. However, our pseudoentanglement construction only provides guarantees on the entanglement structure of the output state. This problem can be addressed using a “padding trick” [26]: we can simply pad the pseudoentanglement circuit with a large (polynomial) number of identity gates at the end. This will ensure that the history state has most weight on the output state. Using continuity properties of the von Neumann entropy, this implies that the history state has the desired entanglement structure, too. The second issue is that we have no control over the entanglement within the clock register of the Hamiltonian. However, this does not matter for the coarse-grained entanglement structure of the state: since the clock register only has logarithmically many qubits, discarding it only changes the entanglement by $O(\log n)$, which is irrelevant for our $O(\text{poly}(\log n))$ vs $\Omega(n)$ entanglement gap.

The Hamiltonian in Theorem 3 does not achieve constant locality because the Hamiltonian terms acting on the binary clock register require locality $\log n$. By using a unary clock instead of a binary clock, we can make the Hamiltonians in Theorem 3 have constant locality [23]. This is also what was used in [18] to study a Hamiltonian version of their entropy difference problem. However, the clock register now has $\Theta(n)$ qubits, and because it has so many qubits, the analysis from Theorem 3 no longer yields the desired entanglement gap. However, if we trace out the clock register and measure entanglement of the remaining mixed state by any operational mixed-state entanglement measure, we show that we still recover a maximal entanglement gap across any cut. Intuitively, this is because due to the padding trick, after tracing out the clock register the remaining mixed state is close in trace distance to the (pure) output state of the pseudoentanglement circuit. We refer to Theorem 28 for the formal statement.

The main downside of the Kitaev clock construction is that the resulting Hamiltonian is not *geometrically* local, i.e. even though we imagine the qubits as arranged on a 1D line in order to talk about area and volume law entanglement, the Hamiltonian itself has no inherent 1D geometrical structure. In contrast, most physical Hamiltonians are geometrically local. To obtain hard instances of the LGSES problem for geometrically local Hamiltonians we can use a more sophisticated 2D clock construction, where we account for time across one of the spatial dimensions instead of needing to add extra clock qubits to the circuit. We first state the resulting hardness statement for LGSES informally and then briefly sketch the proof. We refer to Theorem 31 for the formal statement and Section 4.3 for details of the construction.

► **Theorem 4.** *Assuming subexponential-time hardness of LWE, LGSES is intractable when the input Hamiltonian is geometrically local on a 2D grid of $n \times \text{poly}(n)$ qudits with constant local dimension $d = O(1)$, and the goal is to decide whether the ground state has entanglement scaling $\text{polylog}(n)$ or n across horizontal cuts.*

At a high level, the construction of the 2D geometrically local case is similar to before: we take padded versions of our pseudoentanglement circuits and convert them to local Hamiltonians using a 2D circuit-to-Hamiltonian construction [3]. This circuit-to-Hamiltonian construction produces a geometrically local Hamiltonian by dispensing with an explicit clock register. As a result, the ground state also does not have a clock register and is instead of the form $\sum_t V_t |\psi_t\rangle$ (with normalization), where $|\psi_t\rangle$ is the state of the circuit after time step t and V_t are isometries such that $V_t^\dagger V_{t'} = 0$ for any $t \neq t'$. In other words, similarly to the Kitaev clock construction, the ground state of the Hamiltonian has the form of a history state, with different time steps encoded in mutually orthogonal states. Since we padded the circuit with identities, we can approximate this ground state by $\sum V_t |\psi_{\text{out}}\rangle$ with $|\psi_{\text{out}}\rangle$ the output state of our pseudoentanglement circuit.

The challenge in bounding the entropy of the reduced states of this “history state” is that the different time steps are encoded in different bases, specified by the isometries V_t . This is in contrast to the Kitaev construction, where the intermediate states are all encoded in the same basis. As a result, when we trace out part of the state $\sum V_t |\psi_{\text{out}}\rangle$, we get a state that looks very different from just the reduced state of $|\psi_{\text{out}}\rangle$. With the construction of [3], we do not know how to bound the entanglement of these reduced states.

We therefore need to modify the construction from [3] to gain better control over the entropy of these reduced states. We do this by increasing the local dimension of the qudits in order to better keep track of different steps of the circuit execution. With this modified construction, we can ensure that reduced states of different $V_t |\psi_{\text{out}}\rangle$ corresponding to different phases of the circuit execution are, in a certain sense, “cutwise” orthogonal. The overall reduced

state is now a sum of different “orthogonal” reduced states, each corresponding to a different phase of the circuit execution. We can compute the entropy of each of these individual reduced states relatively easily from the entanglement properties of our pseudoentangled states. Using cutwise orthogonality allows us relate the entropy of the overall state to the entropies of the individual reduced states that we sum over. As a result, we can compute the entropy of the overall reduced state even though all the different time steps are encoded in different bases.

1.3 Related work

We have already given a detailed discussion of the work of Gheorghiu and Hoban [18], which introduced the idea of public key-pseudoentanglement and gave the first construction, and the work of Aaronson et al. [1], which coined the term pseudoentanglement and gave a private-key construction with maximal entanglement gap across any cut.

The main motivation in [18] was to provide a hardness result for the so-called (quantum) entropy difference problem: given two (quantum) circuits, decide whose output has more entropy when acting on a uniformly random input. If the circuit depth is polynomial, these problems are known to be complete for the complexity classes QSZK and SZK, respectively [19, 10]. Gheorghiu and Hoban showed that for constant-depth circuits with unbounded fan-out or logarithmic-depth circuits with constant fan-out, both the QED and ED problems are still at least as hard as breaking LWE. In their proof, the entropy difference between the high- and low-entropy circuits was a single bit. Our improved pseudoentanglement construction implies that both ED and QED remain LWE-hard with large entropy gaps.⁴ This is similar in spirit to the classical result that SZK gaps can be amplified [28].

Recently, independent and complementary work of Arnon-Friedman, Brakerski, and Vidick [7] gave a new definition of pseudoentanglement. Their definition is private-key and is natural in the context of operational tasks in quantum Shannon theory. Consequently, they focus on operational mixed-state entanglement measures across a single cut and require their states to be efficiently preparable under LOCC. In contrast in our work we focus on creating *public-key* pseudoentanglement with different large-scale geometrical structures, which is driven by our applications in Hamiltonian complexity.

Finally, we discuss the relationship between the LGSES problem and existing algorithms for properties of ground states. While the physics literature on computing properties of ground states is too vast to survey here, we highlight two results closer to computer science. First, in [24] the authors provide a polynomial-time algorithm that, given a classical description of a one-dimensional geometrically local Hamiltonian with constant spectral gap, outputs an MPS description of the ground state. Therefore 1D constant gapped geometrically local Hamiltonians cannot “hide” anything about their ground state entanglement structure. We note that this algorithm cannot be used on the Hamiltonians we construct in this paper as they are neither 1D geometrically local nor have constant spectral gap. Therefore our results limit potential further improvements to their algorithm. Second, the recent result [21] considers the problem of distinguishing phases of matter given labelled examples of states

⁴ This result only requires our single-cut pseudoentanglement construction, for which the depth can be made logarithmic as in [18]. In fact, as mentioned earlier, an independently updated version of [18] also achieves single-cut pseudoentanglement with a large gap, implying the same hardness result for the (Q)ED problem that we obtain from our construction, although their circuits have $\text{poly}(n)$ output qubits for an entropy gap of n^δ vs n , whereas ours only have $O(n)$ output qubits, i.e. achieve a larger relative gap. We refer to [18] for a more detailed analysis.

in different phases. The authors show that if there is a constant spectral gap and the separation between the phases is sufficiently well-conditioned⁵, then a classical algorithm can efficiently learn to distinguish between the phases using information from only few-body measurements. In condensed matter physics, different qualitative entanglement structures are often associated with different quantum phases of matter; therefore our result also limits potential further improvements to such algorithms, i.e. it is not possible to relax some of their assumptions e.g. to gapless phases. An interesting direction for future work is to make our pseudoentanglement Hamiltonians “more physical” to be closer to the assumptions of these algorithmic settings. This would help to better delineate the boundary between tractability vs intractability of learning properties of ground states of local Hamiltonians.

1.4 Discussion and open questions

In this work, we have introduced and studied the Learning Ground State Entanglement Structure (LGSES) problem: given a classical description of a local Hamiltonian, determine qualitative properties of the entanglement of its ground state, e.g. whether it is area-law or volume-law entangled. To prove hardness results for this problem, we have related it to a notion that we call public-key pseudoentanglement: low-entanglement states that are computationally indistinguishable from high-entanglement states even when given the state preparation circuit. Our main technical contribution is to construct public-key pseudoentanglement with (near) area-law vs volume-law scaling assuming the hardness of LWE.

Pseudoentanglement is a relatively new idea with many avenues for future work. We suggest three main directions: (i) improving pseudoentanglement constructions themselves, (ii) strengthening the link between pseudoentanglement and condensed matter physics, and (iii) applications of pseudoentanglement beyond Hamiltonian complexity. We briefly discuss each in turn.

- (i) Our public-key pseudoentanglement construction achieves essentially optimal parameters, but its construction uses a fairly involved iterated entanglement reduction procedure. In contrast, the private-key construction from [1] is very simple and based on subset states. It would be desirable to have a similarly straightforward construction of public-key pseudoentanglement, too. Furthermore, as we suggested in our discussion of [7], one can extend our definition of public-key pseudoentanglement to include a trapdoor that allows for efficient distillation of the “hidden” entanglement. It is not obvious how to extend our construction to include this feature.
- (ii) Our hardness results for the LGSES problem use Hamiltonians that differ from the Hamiltonians typically studied in condensed matter physics. For example, while we do prove a hardness result for the LGSES problem for 2D geometrically local Hamiltonians, this only holds for a certain set of cuts across the system. We expect that these results can be improved to be closer to the settings studied in condensed matter physics, such as to geometrically local Hamiltonians with more natural entanglement structures and larger spectral gaps,⁶ which would have implications for the hardness of studying

⁵ In particular, there must be a well-behaved function of few-body observables that separates the phases.

⁶ Of course, we cannot hope to construct hard instances of the LGSES problem where both the area and volume law Hamiltonians are geometrically local and have constant spectral gap. This is simply because the area law (proven in 1D [20] and in 2D under extra conditions [4], but widely believed to hold generally) requires *any* such Hamiltonian to have area law entanglement. However, this does not rule out computationally indistinguishable families of Hamiltonians where the area law Hamiltonian has constant gap and the volume law Hamiltonian has inverse polynomial gap, since determining the spectral gap itself is computationally infeasible [16, 9].

quantum phases of matter. This may require developing new sorts of clock constructions where the only goal is to preserve entanglement structure of BQP computations rather than to encode more general QMA-complete problems.

- (iii) While we have focused on applications in Hamiltonian complexity in this work, pseudo-entanglement might be a useful tool for proving hardness results in other domains, too. For example, recent work [11, 7] has analysed the computational resources required to execute certain tasks from quantum Shannon theory, e.g. entanglement distillation. As observed in [7], proving hardness results for such problems is closely related to pseudoentanglement, and we hope that our construction of public-key pseudoentanglement will lead to additional and stronger hardness results in this direction.

Furthermore, public-key pseudoentanglement might also be interesting from a quantum cryptographic point of view, in particular its trapdoor-variant we suggested above. For example, recent work has focused on finding minimal assumptions for quantum cryptography (see e.g. [32] and references therein for an overview), and it would be interesting to explore how pseudoentanglement is related to these assumptions.

Finally, it is natural to ask if public-key pseudoentanglement might have applications in quantum gravity. The AdS/CFT correspondence postulates that gravitational theories are dual to quantum mechanical systems, and that the entanglement structure of the quantum system is related to the geometry of the gravitational system [27]. Our results show that it is difficult to estimate the entanglement of quantum states. In contrast, geometry seems to be easy to determine, which might provide an argument that this duality is exponentially hard to compute, as first suggested in [12]. Indeed this was part of the motivation for prior works of pseudoentanglement [18, 1, 7]. Our public-key extension might allow one to argue about hardness of different versions of the duality, e.g. the duality remains hard to compute even if given a parent Hamiltonian for the quantum state.

2 Preliminaries

2.1 Notation

We write $[n]$ for the set $\{1, \dots, n\}$. For a bitstring $x \in \{0, 1\}^n$, we denote the m most and least significant bits by $\text{MSB}_m(x)$ and $\text{LSB}_m(x)$, respectively. We denote the concatenation of strings by $x \parallel y$. For a set of indices $I \subset [n]$ and bitstrings $x \in \{0, 1\}^{|I|}$, $y \in \{0, 1\}^{n-|I|}$ we denote by $z = x \parallel_I y$ the string z that equals x in indices in I and y on indices in $[n] \setminus I$. We will occasionally think of a bitstring as a \mathbb{Z}_2 -vector, in which case we denote it as \vec{x} .

For a matrix $A \in \mathbb{C}^{m \times n}$, we denote by $\|A\|_p = \text{Tr}[(A^\dagger A)^{p/2}]^{1/p}$ its Schatten p -norm. The 1-norm is also called the trace norm, the 2-norm is the Frobenius norm (or Hilbert-Schmidt norm), and the ∞ -norm the operator norm.

Quantum systems are denoted by capital letters A, B , etc. For a pure state $|\psi\rangle_{AB}$ or a mixed state ρ_{AB} on systems A and B , we denote the reduced states on system A by ψ_A and ρ_A , respectively. We write quantum circuits as $C = U_T \cdot U_{T-1} \cdots U_1$, where U_i are elementary gates. This should be thought of as a list of gates, not simply a large unitary; in particular, inserting identity gates into the circuit does change the circuit (although of course it does not change the unitary implemented by the circuit).

2.2 Independent hash functions

► **Definition 5** (*r*-wise independent function family). A function family $H = \{h_k : [N] \rightarrow [M]\}_{k \in \mathcal{K}}$ indexed by some set of keys \mathcal{K} is *r*-wise independent if for all distinct $x_1, \dots, x_r \in [N]$, the random variables $h_k(x_1), \dots, h_k(x_r)$ (for $k \in \mathcal{K}$ chosen uniformly) are uniform i.i.d.

The following is a standard result (see e.g. [30, Corollary 3.34]):

► **Lemma 6.** For any $n, m, r \in \mathbb{N}$, there exists an *r*-wise independent function family $H_n = \{h_k : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^m\}_{k \in \mathcal{K}}$ such that each $k \in \mathcal{K}$ has length $\text{poly}(n, m, r, \log q)$ and given $k \in \mathcal{K}$, the function h_k can be evaluated in time $\text{poly}(n, m, r, \log q)$.

2.3 Entropies

We recall the basic definitions of quantum entropies. Throughout, we use the convention that $0 \log 0 = 0$.

► **Definition 7** (von Neumann entropy). The von Neumann entropy of a quantum state ρ is defined as

$$S(\rho) = -\text{Tr}[\rho \log \rho].$$

► **Definition 8** (Conditional von Neumann entropy). The conditional von Neumann entropy of a quantum state ρ_{AB} is defined as

$$S(\rho_{A|B}) = S(\rho_{AB}) - S(\rho_B).$$

► **Definition 9** (Binary entropy function). The binary entropy function is defined as

$$h(x) = -x \log x - (1-x) \log(1-x),$$

for $x \in [0, 1]$.

2.3.1 Continuity properties

► **Lemma 10** (Continuity of the von Neumann entropy [17, 8]). Let ρ_{AB} and σ_{AB} be the density matrix of two *n*-qubit quantum states respectively, each partitioned into subsystems *A* and *B*, and let

$$\frac{1}{2} \|\rho_{AB} - \sigma_{AB}\|_1 \leq \epsilon.$$

Then,

$$|S(\rho_{AB}) - S(\sigma_{AB})| \leq \epsilon \cdot n + h(\epsilon),$$

where $h(\cdot)$ is the binary entropy function.

► **Lemma 11** (Continuity of the conditional von Neumann entropy [31]). Let ρ_{AB} and σ_{AB} be the density matrix of two *n*-qubit quantum states respectively, each partitioned into subsystems *A* and *B*, and let

$$\frac{1}{2} \|\rho_{AB} - \sigma_{AB}\|_1 \leq \epsilon.$$

Then,

$$|S(\rho_{A|B}) - S(\sigma_{A|B})| \leq 2\epsilon \cdot \log |A| + (1 + \epsilon) \cdot h\left(\frac{\epsilon}{1 + \epsilon}\right),$$

where $|A|$ is the dimension of the Hilbert space for subsystem *A* and $h(\cdot)$ is the binary entropy function.

2.4 Entanglement measures

2.4.1 Pure state entanglement measure

For pure states on systems AB , the entanglement between A and B is quantified using the so-called entanglement entropy, which is simply the von Neumann entropy of the reduced state on either subsystem.

► **Definition 12** (Entanglement entropy). *For a pure state $|\psi\rangle_{AB}$, the entanglement entropy between systems A and B is defined as $S(\psi_A)$. Note that this is invariant under swapping A and B since for a pure state $|\psi\rangle_{AB}$, $S(\psi_A) = S(\psi_B)$.*

2.4.2 Entanglement entropy for phase states

► **Definition 13** (T -matrix associated with phase states). *Let $s : \{0, 1\}^n \rightarrow \{0, 1\}$. For an n -qubit phase state*

$$|\psi\rangle = \sum_x (-1)^{s(x)} |x\rangle$$

and a subset $X \subseteq [n]$, we define the “ T -matrix” with respect to the cut X as a $\{\pm 1\}^{2^{|X|} \times 2^{n-|X|}}$ -matrix with entries

$$T_{ij} = (-1)^{s(i\|_X j)},$$

where $\|_X$ is the “index string concatenation” defined in Section 2.1.

► **Lemma 14.** *Let $s : \{0, 1\}^n \rightarrow \{0, 1\}$ and $|\psi\rangle = \sum_x (-1)^{s(x)} |x\rangle$. Then for any cut $X \subseteq [n]$, the entanglement entropy of that cut is bounded by*

$$-\log \left(\left\| \frac{1}{2^n} TT^\top \right\|_2 \right) \leq S(\psi_X) \leq \log \text{rank}(T),$$

where T is the T -matrix of $|\psi\rangle$ across cut X .

3 Public-key pseudoentanglement: definition and construction

In this section, we define and construct public-key pseudoentanglement. Our construction uses a similar idea as in [1, Appendix A], which is to consider phase states whose phases have been manipulated in a particular way to create high or low entanglement. The “manipulation” of these phases is by means of applying a one-to-one or many-to-one function (see Section 3.2 for details). We therefore need to construct such *lossy functions* with the appropriate parameters, which we do in Section 3.1 based on the LWE assumption.

In Section 3.2, we then use these lossy functions to construct indistinguishable families of quantum states where states in one family have high entanglement and states in the other family have low entanglement across a *single fixed bipartition* of the qubits. In Section 3.3, we extend this construction to states that have high or low entanglement *for (almost) every cut on a 1Dimensional line*, i.e. we imagine the qubits of the state being arranged on a line and consider all bipartitions into left and right qubits. We show that under the subexponential-time LWE assumption, it is possible to construct pseudoentangled states of this form where either all cuts have a linear or a polylogarithmic amount of entanglement, which is the largest possible separation, as discussed in Remark 19. In this sense our public-key pseudoentanglement construction is optimal. We will use this multi-cut construction in Section 4 to show that learning the ground state entanglement structure of (classically described) local Hamiltonians is computationally hard (under the LWE assumption).

3.1 Construction of lossy functions

We define the following rounding function for \mathbb{Z}_q elements.

► **Definition 15.** For $q = cp$ with $c \in \mathbb{N}$, divide \mathbb{Z}_q into p consecutive bins. We define $\lfloor x \rfloor_p \in \mathbb{Z}_p$ as the index of the bin in which x lies. For a vector $x \in \mathbb{Z}_q^m$, $\lfloor \vec{x} \rfloor_p \in \mathbb{Z}_p^m$ is defined as the element-wise application of $\lfloor \cdot \rfloor_p$.

► **Definition 16 (Lossy function construction).** Choose parameters $\ell(m), r(m) \leq \text{poly}(m)$. Let $p = 2^4$, $q = 2^m$, and $\sigma = q/m^3$. Let $H_m = \{h_k : \mathbb{Z}_p^m \rightarrow \mathbb{Z}_2^m\}_{k \in \mathcal{K}_m^{\text{hash}}}$ be the $r(m)$ -wise independent function family from Lemma 6. We define two families of functions $f : \{0, 1\}^m \rightarrow \{0, 1\}^m$ indexed by key sets $\mathcal{K}_m^{\text{inj}}, \mathcal{K}_m^{\text{lossy}} \subset \mathbb{Z}_q^{m \times m} \times \mathcal{K}_m^{\text{hash}}$ as follows:

- To sample a key from $\mathcal{K}_m^{\text{inj}}$, denoted $k \leftarrow \mathcal{K}_m^{\text{inj}}$, sample $A \leftarrow U_q^{m \times m}$ and $k^{\text{hash}} \in \mathcal{K}_m^{\text{hash}}$ uniformly. Set $k = (A, k^{\text{hash}})$.
- To sample a key from $\mathcal{K}_m^{\text{lossy}}$, denoted $k \leftarrow \mathcal{K}_m^{\text{lossy}}$, sample $A \leftarrow L_{q, \ell, \sigma}^{m \times m}$ and $k^{\text{hash}} \in \mathcal{K}_m^{\text{hash}}$ uniformly. Set $k = (A, k^{\text{hash}})$.
- For a key $k = (A, k^{\text{hash}}) \in \mathcal{K}_m^{\text{inj}} \cup \mathcal{K}_m^{\text{lossy}}$, the function $f_k : \{0, 1\}^m \rightarrow \{0, 1\}^m$ is defined as

$$f_k(\vec{x}) = h_{k^{\text{hash}}}(\lfloor A \cdot \vec{x} \rfloor_p).$$

3.2 Public-key pseudoentanglement across a single cut

We will now use our lossy function construction from Section 3.1 to construct public-key pseudoentangled states for the middle cut that separates the left and right half of qubits. In Section 3.3 we will use the ideas from this section in an iterated way to construct pseudoentangled states for qubits arranged on a line that are pseudoentangled across every cut on the line. Strictly speaking, all the results in this section follow from the more general analysis in Section 3.3. We spell them out nonetheless because it may be easier for readers to first understand the single-cut construction in detail before moving on to Section 3.3.

We begin by defining single-cut public-key pseudoentanglement formally.

► **Definition 17 (Public-key pseudoentanglement across a single cut).** A public-key pseudoentangled state ensemble with entanglement gap $(f(n), g(n))$ across cuts $X_n \subset [n]$ consists of two sequences of families of quantum states $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ indexed by key sets $\mathcal{K}_n^{\text{low}}$ and $\mathcal{K}_n^{\text{high}}$ respectively with the following properties:

- (i) Every $|\psi_k\rangle \in \Psi_n^{\text{low}} \cup \Psi_n^{\text{high}}$ is an n -qubit state.
- (ii) Every key $k \in \mathcal{K}_n^{\text{low}} \cup \mathcal{K}_n^{\text{high}}$ has length $\text{poly}(n)$, and there exists an efficient sampling procedure that, given as input n and a label “high” or “low”, outputs a key $k \in \mathcal{K}_n^{\text{low}}$ or $k \in \mathcal{K}_n^{\text{high}}$, respectively. We write $k \leftarrow \mathcal{K}_n^{\text{low}}$ and $k \leftarrow \mathcal{K}_n^{\text{high}}$ for keys sampled according to this procedure.
- (iii) Given $k \in \mathcal{K}_n^{\text{low}} \cup \mathcal{K}_n^{\text{high}}$, the corresponding state $|\psi_k\rangle$ is efficiently preparable (without knowing whether $k \in \mathcal{K}_n^{\text{low}}$ or $k \in \mathcal{K}_n^{\text{high}}$). Formally, there exists a uniform polynomial-time circuit family $\{C_n\}$ such that C_n takes as input a key $k \in \mathcal{K}_n^{\text{low}} \cup \mathcal{K}_n^{\text{high}}$ and outputs a state negligibly close to $|\psi_k\rangle$.
- (iv) The keys from $\mathcal{K}_n^{\text{low}}$ and $\mathcal{K}_n^{\text{high}}$ are computationally indistinguishable. Formally, for all $\text{poly}(n)$ -time quantum adversaries \mathcal{A} that take as input a key $k \in \mathcal{K}_n^{\text{low}} \cup \mathcal{K}_n^{\text{high}}$ and output a single bit:

$$\left| \Pr_{k \leftarrow \mathcal{K}_n^{\text{low}}}[\mathcal{A}(k) = 0] - \Pr_{k \leftarrow \mathcal{K}_n^{\text{high}}}[\mathcal{A}(k) = 0] \right| = \text{negl}(n).$$

- (v) *With overwhelming probability, across the cut X_n states in Ψ_n^{low} have entanglement entropy $\Theta(f(n))$ and states in Ψ_n^{high} have entanglement entropy $\Theta(g(n))$. Formally, there exist constants $0 < C_1 < C_2$ such that for all sufficiently large n ,*

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{low}}} [S((\psi_k)_{X_n}) \in [C_1 f(n), C_2 f(n)]] \geq 1 - \text{negl}(n),$$

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{high}}} [S((\psi_k)_{X_n}) \in [C_1 g(n), C_2 g(n)]] \geq 1 - \text{negl}(n).$$

Here, $S((\psi_k)_{X_n})$ is the von Neumann entropy of the reduced state of $|\psi_k\rangle$ on the qubits in the set $X_n \subset [n]$.

► **Remark 18.** We will frequently abuse notation and use entanglement gaps of the form $(O(f(n)), \Omega(g(n)))$. By this, we mean that (across a specified cut X_n) Ψ_n^{low} has entanglement at most $O(f(n))$ and Ψ_n^{high} has entanglement at least $\Omega(g(n))$. Formally, this means that for this case Item v of Definition 17 has to be modified as follows:

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{low}}} [S((\psi_k)_{X_n}) \leq O(f(n))] \geq 1 - \text{negl}(n),$$

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{high}}} [S((\psi_k)_{X_n}) \geq \Omega(g(n))] \geq 1 - \text{negl}(n).$$

► **Remark 19.** A natural question is what the optimal entanglement gap for pseudoentangled states is. Clearly, the high-entanglement states can have entanglement at most $g(n) = O(n)$ across any cut, since the entanglement entropy is upper bounded by the number of qubits. For the low-entanglement states, one can show that the entanglement entropy needs to scale faster than $\log n$, i.e. $f(n) = \omega(\log n)$. Otherwise, one could distinguish the low-entanglement states from the high-entanglement states using a variant of the SWAP test. This was proven in [22] and [1, Appendix F] for private-key pseudoentangled states and the same proof applies to the public-key setting, too.

We now give a construction of single-cut pseudoentangled states based on our lossy function construction from Section 3.1. As we will show in Theorem 21, these states do indeed form pseudoentangled ensembles in the sense of Definition 17. Under the standard LWE assumption, we can achieve an entanglement gap of $(O(n^\delta), \Omega(n))$ for any $\delta > 0$, where n is the number of qubits and the cut divides the qubits into two equal halves; under the stronger subexponential-time LWE assumption we can achieve an entanglement gap of $(O(\text{poly log } n), \Omega(n))$, which is essentially optimal as noted in Remark 19.

For simplicity, for the rest of this subsection we always assume that n is even and write $m = n/2$. We will consider the cut that divides the qubits into two sets of size n ; we could also consider more general cuts and treat them with the same technique, which we do in Section 3.3.

► **Definition 20.** *Fix a function $f(n)$. (This will be treated as a parameter of the construction.) Let $H_n = \{h_k : \{0, 1\}^n \rightarrow \{0, 1\}\}_{k \in \mathcal{K}_n^4}$ be a 4-wise independent family as given in Lemma 6. Instantiate the lossy functions from Section 3.1 with parameters $\ell(m) = \sqrt{f(2m)}$ and $r(m) = 2$. (Recall that $m := n/2$.)*

We first describe the sampling procedure for the keys $\mathcal{K}_n^{\text{low}}$ and $\mathcal{K}_n^{\text{high}}$.

(i) *To sample $k \in \mathcal{K}_n^{\text{low}}$, sample $k^{\text{rep}} \leftarrow \mathcal{K}_m^{\text{lossy}}$ and $k^{\text{fin}} \in \mathcal{K}_n^4$ uniformly. Set $k = (k^{\text{rep}}, k^{\text{fin}})$.*

(ii) *To sample $k \in \mathcal{K}_n^{\text{high}}$, sample $k^{\text{rep}} \leftarrow \mathcal{K}_m^{\text{inj}}$ and $k^{\text{fin}} \in \mathcal{K}_n^4$ uniformly. Set $k = (k^{\text{rep}}, k^{\text{fin}})$. For $k = (k^{\text{rep}}, k^{\text{fin}})$, define the labelling function $r_k : \{0, 1\}^n \rightarrow \{0, 1\}^n$ by*

$$r_k(x) = (f_{k^{\text{rep}}}(i) \parallel j) \quad \text{with } i = \text{MSB}_m(x), j = \text{LSB}_m(x).$$

We next define the function $s_k : \{0, 1\}^n \rightarrow \{0, 1\}$ as

$$s_k(x) = h_{k^{\text{fin}}}(r_k(x)).$$

The states $|\psi_k\rangle$ are then given by

$$|\psi_k\rangle = \sum_{x \in \{0,1\}^n} (-1)^{s_k(x)} |x\rangle.$$

► **Theorem 21.**

- (i) Under the standard LWE assumption, for any function $f(n) = n^\delta$ for $\delta > 0$, the state families $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ from Definition 20 form a pseudoentangled state ensemble with entanglement gap $(O(f(n)), \Omega(n))$ across the cuts $X_n = \lfloor n/2 \rfloor$.
- (ii) Under the subexponential-time LWE assumption, there exists a function $f(n) = \text{poly log } n$ such that the state families $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ from Definition 20 form a pseudoentangled state ensemble with entanglement gap $(O(f(n)), \Omega(n))$ across the cuts $X_n = \lfloor n/2 \rfloor$.

3.3 Area-law public-key pseudoentangled states on a 1D line

In this section we will give a (nearly) area-law public-key pseudoentangled states construction on a line based on the row repetition technique we introduced in Section 3.2. This means that we will construct public-key pseudoentangled states such that if we imagine the qubits arranged on a line, the entanglement gap is $\text{poly log } n$ vs n across all cuts that separate the qubits into left and right qubits on the line.

We begin by formally defining this multi-cut version of pseudoentanglement. The definition is almost identical to Definition 17, except that we now need to require an entanglement gap across all cuts on a line simultaneously. One slight subtlety is that if we consider cuts close to the end of the line, the entanglement will be low simply by virtue of the fact that there are only very few qubits on one side of the cut. Therefore, in the high-entanglement case, we need to require the entanglement to be at least $g(\text{distance from end of line})$ rather than simply $g(n)$. Furthermore, very close to the boundary (namely, $O(\log n)$ close), certain properties of our construction break down. Therefore, we only consider cuts that are at least $\omega(\log n)$ far from the boundary. Since we are primarily interested in large entanglement gaps of the form $(O(\text{poly log } n), \Omega(n))$, this small boundary region is of no particular interest to us. Nonetheless, it is possible to modify our construction to work for such small boundary regions too.

As in the single-cut case, we use entanglement gaps of the form $(O(f(n)), \Omega(g(n)))$ for pseudoentangled states where we only have an upper bound on the entanglement in the low-entanglement case and a lower bound in the high-entanglement case (see Remark 18 for details). To simplify the definition slightly, below we state the definition directly for this case; it is straightforward to adapt it to the case where one wants the exact scaling rather than one-sided bounds, but we will not need this for our results.

► **Definition 22** (Public-key pseudoentanglement across geometrically local cuts in 1D). *A public-key pseudoentangled state ensemble with entanglement gap $(O(f(n)), \Omega(g(n)))$ across geometrically local cuts on a 1D line consists of two sequences of families of quantum states $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ indexed by key sets $\mathcal{K}_n^{\text{low}}$ and $\mathcal{K}_n^{\text{high}}$ respectively that satisfy Items i–iv from Definition 17 and the following modified version of Item v from Definition 17:*

- (v') For any function $b(n) = \omega(\log n)$, with overwhelming probability, states in Ψ_n^{low} have entanglement entropy $O(f(n))$ and states in Ψ_n^{high} have entanglement entropy $\Omega(g(\text{distance from end of line}))$ for all geometrically local cuts that are at least $b(n)$ far from the end of the line. Formally,

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{low}}} [\forall c \in \{b(n), \dots, n - b(n)\} : S((\psi_k)_{[c]}) \leq O(f(n))] \geq 1 - \text{negl}(n),$$

$$\Pr_{k \leftarrow \mathcal{K}_n^{\text{high}}} [\forall c \in \{b(n), \dots, n - b(n)\} : S((\psi_k)_{[c]}) \geq \Omega(\min(g(c), g(n - c)))] \geq 1 - \text{negl}(n).$$

Here, $(\psi_k)_{[c]}$ is the reduced states of $|\psi_k\rangle$ on qubits $(1, \dots, c)$.

► **Definition 23.** Fix a function $f(n)$ (this will be treated as a parameter of the construction). Let $H_n = \{h_k : \{0, 1\}^n \rightarrow \{0, 1\}\}_{k \in \mathcal{K}_n^4}$ be a 4-wise independent family as given in Lemma 6. For $m \in \{f(n), f(n) + 1, \dots, n\}$, instantiate the m -bit lossy function from Section 3.1 with parameters $\ell(m) = \sqrt{f(n)}$ and $r(m) = 2$.

We first describe the sampling procedure for the keys $\mathcal{K}_n^{\text{low}}$ and $\mathcal{K}_n^{\text{high}}$.

- (i) To sample $k \in \mathcal{K}_n^{\text{low}}$, for $m \in \{f(n), f(n) + 1, \dots, n\}$, independently sample $k_m^{\text{rep}} \leftarrow \mathcal{K}_m^{\text{lossy}}$ (see Definition 16) and $k^{\text{fin}} \in \mathcal{K}_n^4$ uniformly. Set $k = (k_{f(n)}^{\text{rep}}, k_{f(n)+1}^{\text{rep}}, \dots, k_n^{\text{rep}}, k^{\text{fin}})$.
- (ii) To sample $k \in \mathcal{K}_n^{\text{high}}$, for $m \in \{f(n), f(n) + 1, \dots, n\}$, independently sample $k_m^{\text{rep}} \leftarrow \mathcal{K}_m^{\text{inj}}$ and $k^{\text{fin}} \in \mathcal{K}_n^4$ uniformly. Set $k = (k_{f(n)}^{\text{rep}}, k_{f(n)+1}^{\text{rep}}, \dots, k_n^{\text{rep}}, k^{\text{fin}})$.

For $k = (k_{f(n)}^{\text{rep}}, k_{f(n)+1}^{\text{rep}}, \dots, k_n^{\text{rep}}, k^{\text{fin}})$, define the labelling functions $r_k^{f(n)}, \dots, r_k^n : \{0, 1\}^n \rightarrow \{0, 1\}^n$ recursively by

$$r_k^m(x) = \begin{cases} x & m = n + 1, \\ r_k^{m+1}(f_{k_m^{\text{rep}}}^m(i) \parallel j) & f(n) \leq m \leq n, \text{MSB}_m(x) = i, \text{LSB}_{n-m}(x) = j, \end{cases}$$

where $\text{MSB}_m(x)$ is the first m bits of x , $\text{LSB}_m(x)$ is the last m bits of x , and $i \parallel j$ is the concatenation of bit strings i and j . For simplicity, we define $r_k(x) = r_k^{f(n)}(x)$.

With this notation, for $k = (k_{f(n)}^{\text{rep}}, k_{f(n)+1}^{\text{rep}}, \dots, k_n^{\text{rep}}, k^{\text{fin}})$, we next define the function $s_k : \{0, 1\}^n \rightarrow \{0, 1\}$ as

$$s_k(x) = h_{k^{\text{fin}}}(r_k(x)),$$

The states $|\psi_k\rangle$ are then given by

$$|\psi_k\rangle = \sum_{x \in \{0, 1\}^n} (-1)^{s_k(x)} |x\rangle.$$

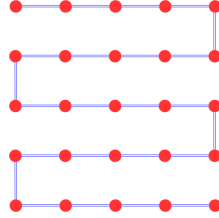
Our main result is that this construction satisfies the requirements from Definition 22 as summarised by the following theorem. In particular, we show that under the subexponential-time LWE assumption, our construction achieves an entanglement gap of $\text{poly log } n$ vs n , which is essentially optimal by Remark 19. On a 1D line, this entanglement scaling corresponds to area-law (up to poly log factors) vs volume law entanglement, which is why we call this result area-law pseudoentangled states.

► **Theorem 24.**

- (i) Under the standard LWE assumption, for any function $f(n) = n^\delta$ for $\delta > 0$, the state families $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ form a pseudoentangled state ensemble with entanglement gap $(O(f(n)), \Omega(n))$ across geometrically local cuts in 1D.
- (ii) Under the subexponential-time LWE assumption, there exists a function $f(n) = \text{poly log } n$ such that the state families $\Psi_n^{\text{low}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{low}}}$ and $\Psi_n^{\text{high}} = \{|\psi_k\rangle\}_{k \in \mathcal{K}_n^{\text{high}}}$ from Definition 23 form a pseudoentangled state ensemble with entanglement gap $(O(f(n)), \Omega(n))$ across geometrically local cuts in 1D.

3.4 Area-law public-key pseudoentangled states on a 2D grid

We can easily generalise the 1D construction from Section 3.3 to a system of qubits arranged on a 2D grid. This is the same construction as in [1, Appendix D.5], so we only provide a short sketch. Let $|\psi_k\rangle$ be an n -qubit state from a pseudoentangled state ensemble. We can arrange the qubits of this state on an $\sqrt{n} \times \sqrt{n}$ grid as shown in Figure 1. Now consider a



■ **Figure 1** Arranging an n -qubit state on a $\sqrt{n} \times \sqrt{n}$ grid.

contiguous 2D subregion R of this $\sqrt{n} \times \sqrt{n}$ grid. Let $|R|$ be the size of R (i.e. the number of qubits in R) and $|\partial R|$ the size of the boundary of R . Unfolding the “snake”, this region R corresponds to a (not necessarily geometrically local) cut in 1D.

For the pseudoentangled state ensembles we constructed in Theorem 24, a high-entanglement state $|\psi_k\rangle$ has entanglement entropy $\Omega(|R|)$ for any (sufficiently large) cut R , even if the cut is not 1D geometrically local. This means that arranged on a 2D grid, the high-entanglement states from Theorem 24 exhibit volume law entanglement scaling.

Conversely, consider a low-entanglement state $|\psi_k\rangle$ from the construction in Theorem 24. From the geometry of Figure 1 it is easy to see that a region R corresponds to a 1D cut that divides the qubits into at most $O(|\partial R|)$ contiguous regions; this is because the boundary of the region R can cut the “snake” at most $O(|\partial R|)$ times. For each of these $O(|\partial R|)$ cuts in 1D, we know from Theorem 24 that $|\psi_k\rangle$ has entanglement entropy at most $O(\text{poly log } n)$ across that cut. Using subadditivity of entanglement entropy, it then follows that the entanglement entropy of the region R is at most $O(|\partial R| \cdot \text{poly log } n)$, which corresponds to area-law scaling in two dimensions (up to polylogarithmic factors).

4 Computational hardness of learning ground state entanglement structure

Our public-key pseudoentanglement constructions can be leveraged to construct Hamiltonians such that it is hard to learn the entanglement structure of their ground state. This is what we will discuss in the next sections. Specifically, we will study variants of the following problem, which we define somewhat informally.

► **Definition 25** (Learning Ground State Entanglement Structure (LGSES) problem). *Given a classical description of a k -local Hamiltonian H on n qubits with spectral gap at least $\frac{1}{\text{poly}(n)}$, decide whether the ground state of H has entanglement structure **A** or **B**? Here, **A** and **B** should be two qualitatively different, pre-specified entanglement structures, e.g. near area-law and volume law entanglement.*

We will see three different variants of this problem for three different types of local Hamiltonians and correspondingly three slightly different entanglement structures. We will progressively make our constructions more local – in some sense, more local corresponds to more physical Hamiltonians – but we will pay a slight price in terms of how straightforwardly the entanglement structures can be described.

- (i) In Section 4.1, we will study the hardness of LGSES for $O(\log n)$ -local Hamiltonians on n qubits arranged in a 1D line. The two entanglement structures to distinguish between will be $\text{poly log } n$ vs $\Omega(n)$ entanglement across geometrically local cuts in 1D. In other words, we are asked to distinguish 1D near area-law vs volume-law entanglement.
- (ii) In Section 4.2, we will improve upon the locality of the Hamiltonian and study the hardness of LGSES for $O(1)$ -local Hamiltonians on n qubits arranged in a 1D line. However, the entanglement structure will be slightly more complicated: we will consider the reduced states of ground states on a specific subsystem and ask whether this has 1D near area-law or volume-law entanglement structure for a mixed state measure of entanglement.
- (iii) In Section 4.3, we will study the hardness of LGSES for 2-local Hamiltonians on a 2D grid of size $n \times \text{poly}(n)$ and with constant local dimension, where all Hamiltonian terms are *geometrically local* (i.e. only nearest neighbors on the grid can interact). The two entanglement structures to distinguish will be entanglement entropy $O(\text{poly log } n)$ vs $\Omega(n)$ across horizontal cuts through the grid.

4.1 1D Hamiltonians with $\log n$ -locality and pure states

In this section, we will show how to obtain two families of $\log n$ -local Hamiltonians, one whose ground state has $\text{poly log } n$ entanglement scaling and the other whose ground state has $\Omega(n)$ entanglement scaling across geometrically local cuts in 1D, such that given the description of one of these Hamiltonians it is computationally hard to decide which family it belongs to.

We will start with the public-key pseudoentangled state constructions in Section 3.3, use the padded circuit-to-Hamiltonian construction, and then use the trace distance closeness property to show that the entanglement structures of the ground states of these Hamiltonians resemble the entanglement structure of the public-key pseudoentangled states.

► **Theorem 26.** *For every $n \in \mathbb{N}$, there exist two families $\mathcal{H}_n^{\text{low}}$ and $\mathcal{H}_n^{\text{high}}$ of $O(\log n)$ -local Hamiltonians on $(n + O(\log n))$ qubits arranged on a 1D line with spectral gap $\Omega(1/\text{poly}(n))$ and efficient procedures that sample (classical descriptions of) Hamiltonians from either family (denoted $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$) with the following properties:*

- (i) *Hamiltonians sampled according to $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$ are computationally indistinguishable under the assumption that LWE is subexponentially hard.*
- (ii) *With overwhelming probability, the ground states of Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{low}}$ have 1D near area-law entanglement and Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{high}}$ have 1D volume-law entanglement. Formally, this means that for geometrically local cuts in 1D of size $r = \omega(\log n)$, the ground states of the Hamiltonians have entanglement entropy $O(\text{poly log } n)$ or $\Omega(\min(r, n - r))$, respectively.*

► **Remark 27.** Under the standard LWE assumption instead of the subexponentially hardness assumption, Theorem 26 still holds, but with the smaller entanglement gap $O(n^\delta)$ vs $\Omega(n)$ for any $\delta > 0$. This mirrors directly the statement in Theorem 24.

4.2 1D Hamiltonians with constant locality and mixed states

In this section, we will modify the construction in Section 4.1 with a unary clock to get constant locality. However, because the clock register will now have $\text{poly}(n)$ qubits, we can no longer simply remove the clock qubits as we did in Theorem 26. Therefore, we will consider the entanglement structure of the reduced density matrices of the ground state with the clock register traced out. Using mixed state entanglement measures, we will show that one such density matrix will have high entanglement, and the other will have low entanglement.

As discussed in Section 2.4, there are many mixed state measures of entanglement. We will show that for any “natural” mixed state entanglement measure, the ground state of the our Hamiltonian (with the clock register traced out) has either high or low entanglement. We achieve this by giving an upper bound on the entanglement of formation of our low entanglement construction and a lower bound on the distillable entanglement of our high entanglement construction. This gives an entanglement gap for any natural entanglement measure. In fact, Hamiltonians constructed from our ensembles of pseudoentangled states achieve a maximally large gap.

► **Theorem 28.** *For every $n \in \mathbb{N}$, there exist two families $\mathcal{H}_n^{\text{low}}$ and $\mathcal{H}_n^{\text{high}}$ of $O(1)$ -local Hamiltonians on $(n + \text{poly}(n))$ qubits arranged on a 1D line with spectral gap $\Omega(1/\text{poly}(n))$ and efficient procedures that sample (classical descriptions of) Hamiltonians from either family (denoted $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$) with the following properties:*

- (i) *Hamiltonians sampled according to $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$ are computationally indistinguishable under the assumption that LWE is subexponentially hard.*
- (ii) *If we trace out $\text{poly}(n)$ many qubits from the ground state of each Hamiltonian, the entanglement gap between the resultant quantum states in the high and low families is $\Omega(\min(r, n - r))$ versus $\mathcal{O}(\text{polylog } n)$, for a cut of size $(r, n - r)$, for any natural entanglement measure. With overwhelming probability, the reduced states on the first n qubits of the ground states of Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{low}}$ have 1D near area-law entanglement and Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{high}}$ have 1D volume-law entanglement with respect to any natural mixed state entanglement measure.*

► **Remark 29.** Just as in Remark 27, under the standard LWE assumption Theorem 28 still holds, but with the smaller entanglement gap $O(n^\delta)$ vs $\Omega(n)$ for any $\delta > 0$.

4.3 2D Hamiltonians with geometric locality and pure states

In this section, we will show how to obtain two families of 2D Hamiltonians on a 2D grid of $\text{poly}(n)$ qudits, one whose ground state has entanglement entropy of order n and the other whose ground state has entanglement entropy of order $\text{polylog } n$, with respect to most horizontal cuts across the 2D grid. Thus, arguably, this gives us a relatively more complicated entanglement structure than the constructions in Theorem 26 and Theorem 28. However, we gain in geometric locality: the Hamiltonian only has nearest neighbor interactions on a 2D grid. Formally, 2D Hamiltonians are defined as follows.

► **Definition 30** (2D (local) Hamiltonian, [2]). *Let H be a Hermitian operator (interpreted as a Hamiltonian, giving the energy of some system). We say that H is an r -state Hamiltonian if it acts on r -state qudits (i.e. $d = r$). When $r = 2$, namely, when the qudits are qubits. We say that H is k -local if it can be written as*

$$H = \sum_i H_i,$$

where each H_i acts non-trivially on at most k qudits. Note that this term does not assume anything about the physical location of the qudits. We say that H is a 2D Hamiltonian if the qudits are arranged on a 2D grid and the terms H_i interact only pairs of nearest neighbor qudits. In particular, a 2D Hamiltonian is 2-local.

► **Theorem 31.** *For every $n \in \mathbb{N}$, there exist two families $\mathcal{H}_n^{\text{low}}$ and $\mathcal{H}_n^{\text{high}}$ of geometrically 2D-local Hamiltonians on $(n \times \text{poly } n)$ qudits arranged in an $n \times \text{poly}(n)$ grid with spectral gap $\Omega(1/\text{poly}(n))$, and there are efficient procedures that sample (classical descriptions of) Hamiltonians from either family (denoted $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$) with the following properties:*

- (i) *Hamiltonians sampled according to $H \leftarrow \mathcal{H}_n^{\text{low}}$ and $H \leftarrow \mathcal{H}_n^{\text{high}}$ are computationally indistinguishable under the assumption that LWE is subexponentially hard.*
- (ii) *With overwhelming probability, the ground states of Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{low}}$ have $\text{poly } \log n$ entanglement across horizontal cuts through the grid that are at least $\omega(\log n)$ far from the boundary, and Hamiltonians $H \leftarrow \mathcal{H}_n^{\text{high}}$ have $\Omega(n)$ entanglement across the same cuts.*

Overview. The main steps of our construction are as follows:

- First, we start with two n -qubit public key pseudoentangled states across multiple cuts, according to the construction in Section 3.3, and consider the circuits for preparing them. Suppose these circuits have $K = \text{poly}(n)$ gates. Without loss of generality, we assume the circuit can be decomposed into $R = \text{poly}(n)$ “rounds”, each made up of exactly n nearest-neighbor interactions on qubits 1, (1,2), (2,3), etc. Any circuit can be transformed into this form by inserting a polynomial number of identity and swap gates. Hence, the circuit contains nR gates in total. As in Section 4.1, we pad the circuits with nM identity gates at the end for a sufficiently large $M = \text{poly}(n)$.
- Then, we use a modified version of the 2D clock construction [3] to construct two families of 2D Hamiltonians such that the padded circuit is embedded into its ground state. That is, if $C = U_{nT} \cdot U_{nT-1} \cdots U_1$ is the circuit with padding, where $T = M + R$ is the total number of rounds after padding, we construct a Hamiltonian H such that the ground state $|\psi_{\text{ground}}\rangle$, on an $n \times T$ grid of qudits, encodes the time evolution of the padded circuit.
- We show how, because of the padding, the entanglement structure of the 2D ground state across any horizontal cut resembles the entanglement structure of the state

$$|\psi_{\text{output}}\rangle = U_{nR} \cdot U_{nR-1} \cdots U_1 |0^n\rangle, \quad (4.1)$$

across the same cut.

- By virtue of our pseudoentanglement construction, the state in Equation (4.1) either has high or low entanglement, whenever the cut has distance $\omega(\log n)$ to the boundary of the grid. Then, by a continuity argument, we show that the ground state $|\psi_{\text{ground}}\rangle$ also inherits the high or low entanglement property.

References

- 1 Scott Aaronson, Adam Bouland, Bill Fefferman, Soumik Ghosh, Umesh Vazirani, Chenyi Zhang, and Zixin Zhou. Quantum pseudoentanglement. *arXiv preprint v2*, 2023. [arXiv:2211.00747v2](https://arxiv.org/abs/2211.00747v2).
- 2 Dorit Aharonov, Daniel Gottesman, Sandy Irani, and Julia Kempe. The power of quantum systems on a line. *Communications in mathematical physics*, 287(1):41–65, 2009.
- 3 Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.
- 4 Anurag Anshu, Itai Arad, and David Gosset. An area law for 2d frustration-free spin systems. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 12–18, 2022.

- 5 Anurag Anshu and Srinivasan Arunachalam. A survey on the complexity of learning quantum states. *arXiv preprint*, 2023. [arXiv:2305.20069](https://arxiv.org/abs/2305.20069).
- 6 Itai Arad, Zeph Landau, Umesh Vazirani, and Thomas Vidick. Rigorous RG algorithms and area laws for low energy eigenstates in 1D. *Communications in Mathematical Physics*, 356:65–105, 2017.
- 7 Rotem Arnon-Friedman, Zvika Brakerski, and Thomas Vidick. Computational entanglement theory, 2023. [arXiv:2310.02783](https://arxiv.org/abs/2310.02783).
- 8 Koenaad M R Audenaert. A sharp continuity estimate for the von Neumann entropy. *Journal of Physics A: Mathematical and Theoretical*, 40(28):8127–8136, June 2007. [doi:10.1088/1751-8113/40/28/s18](https://doi.org/10.1088/1751-8113/40/28/s18).
- 9 Johannes Bausch, Toby S Cubitt, Angelo Lucia, and David Perez-Garcia. Undecidability of the spectral gap in one dimension. *Physical Review X*, 10(3):031038, 2020.
- 10 Avraham Ben-Aroya, Oded Schwartz, and Amnon Ta-Shma. Quantum expanders: Motivation and constructions. In *2008 23rd Annual IEEE Conference on Computational Complexity*, pages 292–303. IEEE, 2008.
- 11 John Bostanci, Yuval Efron, Tony Metger, Alexander Poremba, Luowen Qian, and Henry Yuen. Unitary complexity and the uhlmann transformation problem. *arXiv preprint*, 2023. [arXiv:2306.13073](https://arxiv.org/abs/2306.13073).
- 12 Adam Bouland, Bill Fefferman, and Umesh Vazirani. Computational pseudorandomness, the wormhole growth paradox, and constraints on the ads/cft duality. *arXiv preprint*, 2019. [arXiv:1910.14646](https://arxiv.org/abs/1910.14646).
- 13 Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. *IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 320–331, 2018. [doi:10.1109/FOCS.2018.00038](https://doi.org/10.1109/FOCS.2018.00038).
- 14 Zvika Brakerski and Omri Shmueli. (Pseudo) random quantum states with binary phase. In *Theory of Cryptography Conference*, pages 229–250. Springer, 2019.
- 15 Marcus Cramer, Martin B Plenio, Steven T Flammia, Rolando Somma, David Gross, Stephen D Bartlett, Olivier Landon-Cardinal, David Poulin, and Yi-Kai Liu. Efficient quantum state tomography. *Nature communications*, 1(1):149, 2010.
- 16 Toby S Cubitt, David Perez-Garcia, and Michael M Wolf. Undecidability of the spectral gap. *Nature*, 528(7581):207–211, 2015.
- 17 M. Fannes. A continuity property of the entropy density for spin lattice systems. *Communications in Mathematical Physics*, 31(4):291–294, December 1973. [doi:10.1007/bf01646490](https://doi.org/10.1007/bf01646490).
- 18 Alexandru Gheorghiu and Matty J Hoban. Estimating the entropy of shallow circuit outputs is hard. *arXiv preprint*, 2020. [arXiv:2002.12814](https://arxiv.org/abs/2002.12814).
- 19 Oded Goldreich and Salil Vadhan. Comparing entropies in statistical zero knowledge with applications to the structure of szk. In *Proceedings. Fourteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference)(Cat. No. 99CB36317)*, pages 54–73. IEEE, 1999.
- 20 Matthew B Hastings. An area law for one-dimensional quantum systems. *Journal of statistical mechanics: theory and experiment*, 2007(08):P08024, 2007.
- 21 Hsin-Yuan Huang, Richard Kueng, Giacomo Torlai, Victor V Albert, and John Preskill. Provably efficient machine learning for quantum many-body problems. *Science*, 377(6613):eabk3333, 2022.
- 22 Zhengfeng Ji, Yi-Kai Liu, and Fang Song. Pseudorandom quantum states. In *Advances in Cryptology—CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part III 38*, pages 126–152. Springer, 2018.
- 23 A. Yu. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, USA, 2002.

- 24 Zeph Landau, Umesh Vazirani, and Thomas Vidick. A polynomial time algorithm for the ground state of one-dimensional gapped local Hamiltonians. *Nature Physics*, 11(7):566–569, 2015.
- 25 Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. Efficient certifiable randomness from a single quantum device, 2022. [arXiv:2204.11353](https://arxiv.org/abs/2204.11353).
- 26 Chinmay Nirkhe, Umesh Vazirani, and Henry Yuen. Approximate low-weight check codes and circuit lower bounds for noisy ground states. *arXiv preprint*, 2018. [arXiv:1802.07419](https://arxiv.org/abs/1802.07419).
- 27 Shinsei Ryu and Tadashi Takayanagi. Holographic derivation of entanglement entropy from the anti-de sitter space/conformal field theory correspondence. *Physical review letters*, 96(18):181602, 2006.
- 28 Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM (JACM)*, 50(2):196–249, 2003.
- 29 Norbert Schuch, Ignacio Cirac, and Frank Verstraete. Computational difficulty of finding matrix product ground states. *Physical review letters*, 100(25):250501, 2008.
- 30 Salil Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.
- 31 Andreas Winter. Tight uniform continuity bounds for quantum entropies: Conditional entropy, relative entropy distance and energy constraints. *Communications in Mathematical Physics*, 347(1):291–313, March 2016. doi:10.1007/s00220-016-2609-8.
- 32 Mark Zhandry. Quantum minimalism, 2023. Talk at Simons Institute, <https://www.youtube.com/live/7cqnRASfjco?si=1XLLZpqfaEsBEVp8>.

Depth- d Frege Systems Are Not Automatable Unless $P = NP$

Theodoros Papamakarios  

Department of Computer Science, University of Chicago, IL, USA

Abstract

We show that for any integer $d > 0$, depth- d Frege systems are NP-hard to automate. Namely, given a set S of depth- d formulas, it is NP-hard to find a depth- d Frege refutation of S in time polynomial in the size of the shortest such refutation. This extends the result of Atserias and Müller [JACM, 2020] for the non-automatability of resolution – a depth-1 Frege system – to Frege systems of any depth $d > 0$.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof complexity, Automatability, Bounded-depth Frege

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.22

Acknowledgements I would like to thank Alexander Razborov for numerous remarks and suggestions that greatly improved the presentation of the paper.

1 Introduction

Since its inception as child discipline of mathematical logic, computability, and by extension complexity theory, has had the following two questions at its core: First, broadly asked, how hard is it to prove a theorem, and secondly, knowing that a proof exists, how hard is it to find one. Significantly refining earlier results, most notably [1], Atserias and Müller [2] showed that a version of the latter question, even for a system as weak as resolution, is the same as asking whether $P = NP$.

Namely, a proof system σ is called *automatable* if there is an algorithm that, given a provable formula ϕ , constructs a proof of ϕ in σ , in time polynomial in the size of the smallest proof of ϕ in σ . What Atserias and Müller show is that resolution is not automatable unless $P = NP$.

Now, resolution lies at the bottom of a hierarchy of proof systems, the so called Frege systems of bounded depth, the d -th level of that hierarchy – depth- d Frege – being a system operating with formulas of depth d . It seems plausible that the more complicated the proof systems is, the harder it is to automate it. Following this intuition, as depth- $(d - 1)$ Frege is a subsystem of depth- d Frege, the latter should be harder to automate. We show that for any d , depth- d Frege is as hard to automate as possible. More specifically, we extend the Atserias-Müller result, to show:

► **Theorem 1.1.** *If $P \neq NP$, then for any $d > 0$, depth- d Frege systems are not automatable.*

The Atserias-Müller result has been extended to cutting planes [11], $\text{Res}(k)$ [10], and various algebraic proof systems [7]. Whether it can be extended to bounded-depth Frege systems had remained open. It should be noted that the non-automatability of bounded-depth Frege systems was known under a stronger assumption, namely that the Diffie-Hellman key exchange protocol cannot be broken with circuits of subexponential size [4]. The present paper improves on [4] on three fronts. First, the assumption $P \neq NP$ is much weaker, in particular, it is as weak as possible. Secondly, the result of [4] only works for sufficiently large d , while ours works for all d . Finally, our result requires proving new lower bounds for bounded-depth



© Theodoros Papamakarios;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 22; pp. 22:1–22:17



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Frege, unlike the approach of [4]. However, still, this result and ours are incomparable: [4] rules out even the weak automatability of bounded-depth Frege systems, which is to say that no system polynomially simulating a depth- d Frege system is automatable.

Proof outline

The proof is a reduction from SAT. We want for any positive integer d , given a CNF formula F , to construct a formula G such that if F is satisfiable, then G has small depth- d refutations, whereas if F is unsatisfiable, then G requires large depth- d refutations.

For $d = 1$, [2] considers the formula $G := \text{Ref}(F, z)$ expressing that z encodes a resolution refutation of F . Then a “relativization” construction is applied to $\text{Ref}(F, z)$ to get the formula $\text{RRef}(F, z)$, stating that z is either itself, or contains a resolution refutation of F . It is shown by Pudlák [16] that if F is satisfiable, then the formula $\text{Ref}(F, z)$ has short resolution refutations, and this readily extends to $\text{RRef}(F, z)$ [2]. To show a lower bound for $\text{RRef}(F, z)$ in the case F is unsatisfiable, first it is argued in [2] that there cannot be resolution refutation of $\text{RRef}(F, z)$ having small index-width, where the index-width of a resolution refutation π of $\text{RRef}(F, z)$ is defined as the maximum number of clauses of z contained in a clause of π . Then it is shown, following [6], that if $\text{RRef}(F, z)$ had a small resolution refutation, then $\text{Ref}(F, z')$, where the size of z' is polynomially related to the size of z , would have a resolution refutation of small index-width. Notice that arguing in terms of a variant of width, index-width in this case, is necessary. The same argument could not have worked for width, since resolution is automatable with respect to width, in the sense that a resolution refutation of F having width w can be found in time $n^{O(w)}$, where n is the number of variables of F .

To extend the above for the case $d > 1$, an idea is to employ the construction of [14] (see also [3]), replacing every variable of $\text{RRef}(F, z)$ with Sipser functions, i.e. formulas of the form

$$\bigwedge_{i_1} \bigvee_{i_2} \cdots \bigwedge_{i_k=1} x_{i_1, \dots, i_k}$$

or

$$\bigvee_{i_1} \bigwedge_{i_2} \cdots \bigvee_{i_k=1} x_{i_1, \dots, i_k},$$

for some suitable k . Following [14, 3], one gets a lower bound by repeated applications of Håstad’s switching lemma [12], which reduce a size lower bound to essentially a width lower bound. In our case, we need a reduction in the base case of the argument to a lower bound for index-width, and trying to apply Håstad’s switching lemma for index-width instead of width, one encounters several difficulties, a main one being that the variables encoding the clauses of z induce an exponential factor in the switching probability, making the lemma trivial. We are able to overcome these difficulties by applying the weaker Furst-Saxe-Sipser switching lemma [8], which can use restrictions that fix much less variables on average than Håstad’s switching lemma. This will give a weaker lower bound, only polynomial in our case, which nonetheless is sufficient for the purposes of showing non-automatability.

Let us note that the reduction described above is to a formula that has large depth. In particular, this does not rule out the possibility of bounded-depth Frege systems being automatable when restricted on refuting CNF formulas. To show non-automatability for refuting CNF formulas, one would need to describe a reduction to a CNF formula. This however we expect to be hard to do; see the discussion in the concluding section.

2 Bounded-depth Frege systems and automatability

2.1 Basic definitions

We assume that formulas are built from constants 0 and 1, propositional variables and their negations, unbounded conjunctions and unbounded disjunctions. So negations can only appear next to variables. The *depth* of a formula is the maximum nesting of conjunctions and disjunctions in it. Formally,

$$d(0) = d(1) = d(x) = d(\neg x) = 0,$$

$$d(\circ\{F_1, \dots, F_k\}) = 1 + \max_i d(F_i),$$

where x is a variable and \circ is either a conjunction \wedge or a disjunction \vee .

Depth-0 formulas that are not constants are called *literals*. We often write literals in the form x^ε , where $x^1 := x$ and $x^0 := \neg x$. Depth-1 formulas are called *clauses/terms*, clauses being disjunctions and terms conjunctions of literals. Depth-2 formulas that are disjunctions of terms are called *DNF formulas* and depth-2 formulas that are conjunctions of clauses are called *CNF formulas*. DNF formulas each conjunction of which consists of at most k literals are called *k-DNF* formulas; *k*-CNF formulas are defined similarly. We define $\Sigma_d^{s,k}$ to be the class of all formulas F for which there is a depth- d formula G that is semantically equivalent to F , the outermost connective of G is \vee , and

1. G contains at most s subformulas of depth at least 2;
2. all depth-2 subformulas of G are either *k*-DNFs or *k*-CNFs.

Similarly, $\Pi_d^{s,k}$ is defined as the class of all formulas F for which there is a depth- d formula G semantically equivalent to F , the outermost connective of which is \wedge , satisfying the above two conditions.

A *restriction* is an assignment $\rho : V \rightarrow \{0, 1\}$ of truth values to a set V of variables. For a restriction ρ and a formula F , we denote by $F|_\rho$ the formula resulting by replacing every variable x of F which is in the domain of ρ by $\rho(x)$, and then eliminating constants from $F|_\rho$ using the identities

$$A \vee 0 = A, \quad A \vee 1 = 1, \quad A \wedge 0 = 0, \quad A \wedge 1 = A.$$

We call a restriction that gives a value to all variables a *total assignment*, or simply assignment. For a set S of formulas, we write $S \models F$ if for any total assignment α , $G|_\alpha = 1$ for every $G \in S$ implies $F|_\alpha = 1$. For formulas F and G , we write $F \equiv G$ if F and G are semantically equivalent, i.e. it holds that $F \models G$ and $G \models F$.

2.2 LK proofs

Bounded-depth Frege systems are commonly presented as subsystems of sequent calculus (*LK* for short) for propositional logic. We give a Tait-style formulation of LK, where we write cedents as disjunctions. The inference rules of the system are shown in Table 1. There, x stands for a propositional variable, A and B stand for arbitrary formulas whose top-most connective is \vee , ϕ stands for an arbitrary propositional formula and Φ stands for a set of propositional formulas. $\bar{\phi}$ is the formula that results from ϕ by exchanging every occurrence of \vee with \wedge and vice versa, and replacing each literal x^ε with $x^{1-\varepsilon}$.

An LK proof from a set of premises S is a sequence of formulas, called the *lines* of the proof, such that each line either belongs to S or results from earlier lines by one of rules of Table 1. If the last line in a proof is the empty disjunction, then the proof is called a *refutation*. A depth- d LK proof is an LK proof each line of which is a formula of depth at most d . The *size* of a proof is the total number of symbols occurring in it.

■ **Table 1** The rules of LK.

$\text{Axioms: } \frac{}{x \vee \neg x}$
$\text{Weakening: } \frac{A}{A \vee B}$
$\vee\text{-introduction: } \frac{A \vee \phi}{A \vee \bigvee \Phi}, \text{ where } \phi \in \Phi$
$\wedge\text{-introduction: } \frac{A \vee \phi_1 \quad \dots \quad A \vee \phi_k}{A \vee \bigwedge \{\phi_1, \dots, \phi_k\}}$
$\text{Cut: } \frac{A \vee \phi \quad B \vee \bar{\phi}}{A \vee B}$

Of particular importance among depth- d LK proofs is the case of depth-1 proofs, called *resolution* proofs. In resolution proofs, lines are clauses, and the only applicable LK rules are the weakening and cut rule, which take the form

$$\frac{C}{C \vee D}, \quad \frac{C \vee x \quad D \vee \neg x}{C \vee D}$$

for clauses C and D . In the rightmost rule, also called the resolution rule, we say that $C \vee D$ is the result of *resolving* $C \vee x$ on $D \vee \neg x$ on x .

We may view a proof as a DAG, by drawing for every line A , edges from the lines A is derived to A . In case a proof DAG is a tree, we refer to the proof as being *tree-like*. The next propositions, due to [14], state that depth- d LK proofs and tree-like depth- $(d+1)$ LK proofs can be turned into one another with only a polynomial increase in size.

► **Proposition 2.1** [14]. *A depth- d LK proof of a formula F from S of size s can be turned into a depth- $(d+1)$ tree-like LK proof of F from S of size polynomial in s .*

► **Proposition 2.2** [14, 3]. *Let S be a set of formulas of depth at most d and F a formula of depth at most d . A depth- $(d+1)$ tree-like LK proof of a formula F from S of size s can be turned into a depth- d LK proof of F from S of size $O(s^2)$.*

2.3 Semantic proofs, variable width and decision trees

A semantic depth- d (Frege) proof from a set of formulas S is a sequence of depth- d formulas F_1, \dots, F_t such that for every i , either $F_i \in S$ or there are $j, k < i$ such that $F_j, F_k \models F_i$. Notice that if S consists of depth- $(d-1)$ formulas, then there is a trivial depth- d proof of any valid consequence of S , as $\bigwedge S$ can be derived in $|S| - 1$ steps. Thus, under this formulation, depth- d proofs from S are interesting only if S contains depth- d formulas not in $\Pi_d^{s,k}$ for any s and k , and indeed, our results pertain to such proofs.

The definitions of lines, size of a proof, refutation, tree-like proofs, apply to semantic proofs as well. The *variable width* of a proof is the maximum number of variables among the lines of the proof.

Unlike size, variable width is an inherently semantic notion. In particular, it is independent of depth: any depth- d proof of variable width w can be transformed into a depth-1 proof of (variable) width $O(w)$. In fact, something stronger can be said. A *decision tree* is a binary tree the internal nodes of which are labelled by variables, and the edges by values 0 or 1. Nodes query variables and the edges going from a node to its children are labelled, one by the value 0 and the other by 1, giving an answer to that query. No variable is repeated in a branch so that branches correspond to restrictions, and each branch has a value, 0 or 1, associated with it, so that the decision tree represents a Boolean function. We denote the set of branches of \mathbf{T} having the value v by $\text{Br}_v(\mathbf{T})$. Specifically, we say that a decision tree \mathbf{T} represents a formula F if for every branch π of \mathbf{T} with value v , $F|_\pi \equiv v$. The *height* of a decision tree is the length of its longest branch. Notice that if a formula F is represented by a decision tree of height h , then $F \in \Sigma_2^{1,h} \cap \Pi_2^{1,h}$. We write $h(F)$ for the minimum height of a decision tree representing F . The following lemma is shown in [18] for a specific type of depth-2 proofs, but holds for proofs of arbitrary depth, or for that matter, arbitrary sound proofs.

► **Lemma 2.3.** *Let S be a set of clauses each containing at most h literals. If there is a semantic refutation of S each line of which is represented by a decision tree of height at most h , then there is a resolution refutation of S of width at most $3h$.*

Proof. Let F_1, \dots, F_t be a semantic refutation of S and let \mathbf{T}_i be a decision tree of height at most h representing F_i . We assume that \mathbf{T}_t has a single node having the value 0. For a restriction π , let C_π be the minimal clause falsified by π . We will show that for every i , for every branch $\pi \in \text{Br}_0(\mathbf{T}_i)$, we can derive C_π via a resolution proof of width at most $3h$. Notice that C_π for $\pi \in \text{Br}_0(\mathbf{T}_t)$ is the empty clause, so this construction will give a refutation.

If F_i is a clause C in S , then every $\pi \in \text{Br}_0(\mathbf{T}_i)$ must make every literal in C false, hence C_π is a weakening of C . Assume now that F_i is derived from F_j and F_k and we have derived all clauses C_π for $\pi \in \text{Br}_0(\mathbf{T}_j) \cup \text{Br}_0(\mathbf{T}_k)$. Let $\sigma \in \text{Br}_0(\mathbf{T}_i)$, and let \mathbf{T} be the tree resulting by appending a copy of \mathbf{T}_k at the end of every branch $\pi \in \text{Br}_1(\mathbf{T}_j)$ of \mathbf{T}_j . We will use \mathbf{T} to extract a resolution proof of C_σ . More specifically, for every node u of \mathbf{T} such that the path π_u from the root of \mathbf{T} to u corresponds to a restriction that is consistent with σ , we will derive $C_\sigma \vee C_{\pi_u}$. When we reach the root of \mathbf{T} we will have derived C_σ . If u is a leaf of \mathbf{T} , then we claim that C_{π_u} is a weakening of some clause C_π for $\pi \in \text{Br}_0(\mathbf{T}_j) \cup \text{Br}_0(\mathbf{T}_k)$. To see this, let $\pi_u = \pi_j \cup \pi_k$, where π_j is the part of π_u that belongs to \mathbf{T}_j and π_k the part that belongs to \mathbf{T}_k . Since $F_j, F_k \models F_i$ and π_u is consistent with σ , it cannot be the case that both $\pi_j \in \text{Br}_1(\mathbf{T}_j)$ and $\pi_k \in \text{Br}_1(\mathbf{T}_k)$, otherwise a total assignment extending both π_u and σ would make F_j and F_k true, but F_i false. Suppose now that u is not a leaf of \mathbf{T} and suppose that v and w are its children. Then either π_v and π_w are both consistent with σ , in which case $C_\sigma \vee C_{\pi_u}$ can be derived by resolving $C_\sigma \vee C_{\pi_v}$ and $C_\sigma \vee C_{\pi_w}$ on the variable labelling u , or one of the children, say v , will be consistent with σ and thus $C_\sigma \vee C_{\pi_u}$ will be identical to $C_\sigma \vee C_{\pi_v}$. ◀

2.4 Automatability and the main result

A proof system σ is called *automatable* [5] if there is an algorithm that given a set of formulas S and a formula ϕ provable from S , outputs a σ -proof of ϕ from S in time polynomial $r + s$, where r is the total size of S and s the size of the shortest σ -proof of ϕ from S .

The main theorem of this paper is the fact that approximating the minimum size of a depth- d Frege refutation within a polynomial factor is NP hard:

► **Theorem 2.4.** *For every integer $d > 0$, there is a polynomial-time computable function, which takes as input a CNF formula F with n variables and m clauses and integers $s, N > 0$ represented in unary, and returns a formula $G_d(F; s, N)$ of depth d such that*

1. *if F is satisfiable, then there is a depth- d LK refutation of $G_d(F; s, N)$ of size*

$$O\left(\left(N^{d+3}s^2n(m+s^2n^3)\right)^2\right);$$

2. *if F is not satisfiable, N is an increasing function of n and s is a polynomial in n , every semantic depth- d refutation of $G_d(F; s, N)$ must have size at least*

$$N^{\frac{1}{3}\left(\frac{\log s}{\log n}-2\right)^{\frac{1}{d-1}}}$$

for large enough n .

The NP hardness of automating depth- d Frege systems follows from Theorem 2.4 by setting $s := n^{(3h)^{d-1}+2}$ and $N := s$ for a large enough constant h (see Theorem 6.1).

We describe the reduction, constructing the formula $G_d(F; s, N)$ from F in Section 3. In Section 4, we show the upper bound of Theorem 2.4, and in Section 5 we show the lower bound. It is important to note that both bounds hold for semantic depth- d refutations. The reason we formulate the upper bound in terms of LK refutations is twofold. First, we are able to apply Proposition 2.2; we contend it is much cleaner to first give a depth- $(d+1)$ tree-like LK refutation of our formulas and then convert it to a depth- d refutation, rather than directly giving a depth- d refutation. Secondly, the notion of automatability is neither monotone nor anti-monotone. Hence it is clear from Theorem 2.4 that the non automatability result applies to any version intermediate between depth- d LK and depth- d semantic systems.

3 The formulas Ref

Let F be a CNF formula with n variables and m clauses. The key ingredient in the non-automatability result of [2] is expressing by a set of clauses $\text{Ref}(F, s)$ the statement that there is a resolution refutation D_1, \dots, D_s of length s from the clauses of F .

The variables of $\text{Ref}(F, s)$ are $D[u, i, b]$, $V[u, i]$, $I[u, j]$, $L[u, v]$ and $R[u, v]$, where $u, v \in [s]$, $i \in [n]$, $j \in [m]$ and $b \in \{0, 1\}$. The meaning of $D[u, i, b]$ is that x_i^b appears in D_u . The meaning of $V[u, i]$ is that D_u is derived as a weakening of the resolvent of two previous clauses on x_i , and the meaning of $I[u, j]$ is that D_u is a weakening of the j -th clause of F . The meaning of $L[u, v]$ is that the left clause (i.e. that which contains $\neg x_i$) from which D_u was derived is D_v , and the meaning of $R[u, w]$ is that the right clause (i.e. that which contains x_i) from which D_u was derived is D_w . We will also use the variables $V[u, 0]$ and $I[u, 0]$ to indicate whether D_u is derived from previous clauses or from an initial clause of F : in the former case, $I[u, 0]$ will be true and $V[u, 0]$ false, and in the latter $V[u, 0]$ will be true and $I[u, 0]$ false. The clauses of $\text{Ref}(F, s)$ encode the following conditions: For each $u, v \in [s]$, $i, i' \in [n]$, $j \in [m]$ and $b \in \{0, 1\}$,

$$\exists!k V[u, k] \ \& \ \exists!k I[u, k] \ \& \ \exists!k L[u, k] \ \& \ \exists!k R[u, k]; \quad (3.1)$$

$$V[u, 0] \iff \neg I[u, 0]; \quad (3.2)$$

$$\neg L[u, v] \text{ for } v \geq u \ \& \ \neg R[u, v] \text{ for } v \geq u; \quad (3.3)$$

$$V[u, i] \ \& \ L[u, v] \implies D[v, i, 0]; \quad (3.4)$$

$$V[u, i] \ \& \ R[u, v] \implies D[v, i, 1]; \quad (3.5)$$

$$V[u, i] \ \& \ L[u, v] \ \& \ D[v, i', b] \ \& \ i \neq i' \implies D[u, i', b]; \quad (3.6)$$

$$V[u, i] \ \& \ R[u, v] \ \& \ D[v, i', b] \ \& \ i \neq i' \implies D[u, i', b]; \quad (3.7)$$

$$I[u, j] \ \& \ x_i^b \text{ appears in } C_j \implies D[u, i, b]; \quad (3.8)$$

$$\neg D[u, i, 0] \vee \neg D[u, i, 1]; \quad (3.9)$$

$$\neg D[s, i, b]. \quad (3.10)$$

It was shown, subsequent to [2], that $\text{Ref}(F, s)$ is hard for resolution whenever F is unsatisfiable [9]. In [2], a variation, $\text{RRef}(F, s)$, is used. $\text{RRef}(F, s)$ expresses the fact that there is a resolution refutation D_1, \dots, D_s or one contained in D_1, \dots, D_s , from the clauses of F . $\text{RRef}(F, s)$ has the same variables as $\text{Ref}(F, s)$ plus a new variable $P[u]$ indicating which of the indices $1, \dots, s$ are active, i.e. are part of the refutation. The clauses of $\text{RRef}(F, s)$ express the following conditions, which are those of $\text{Ref}(F, s)$ conditioned on the fact that $P[u]$ is true, in addition to three new ones requiring $P[s]$ to be true, and $P[v]$ to be true whenever $P[u]$ and $L[u, v]$ or $R[u, v]$ are true:

$$P[u] \implies \exists!k V[u, k] \ \& \ \exists!k I[u, k] \ \& \ \exists!k L[u, k] \ \& \ \exists!k R[u, k]; \quad (3.11)$$

$$P[u] \implies (V[u, 0] \iff \neg I[u, 0]); \quad (3.12)$$

$$P[u] \implies \neg L[u, v] \text{ for } v \geq u \ \& \ \neg R[u, v] \text{ for } v \geq u; \quad (3.13)$$

$$P[u] \implies (V[u, i] \ \& \ L[u, v] \implies D[v, i, 0]); \quad (3.14)$$

$$P[u] \implies (V[u, i] \ \& \ R[u, v] \implies D[v, i, 1]); \quad (3.15)$$

$$P[u] \implies (V[u, i] \ \& \ L[u, v] \ \& \ D[v, i', b] \ \& \ i \neq i' \implies D[u, i', b]); \quad (3.16)$$

$$P[u] \implies (V[u, i] \ \& \ R[u, v] \ \& \ D[v, i', b] \ \& \ i \neq i' \implies D[u, i', b]); \quad (3.17)$$

$$P[u] \implies (I[u, j] \ \& \ x_i^b \text{ appears in } C_j \implies D[u, i, b]); \quad (3.18)$$

$$P[u] \implies (\neg D[u, i, 0] \vee \neg D[u, i, 1]); \quad (3.19)$$

$$P[s] \ \& \ \neg D[s, i, b]; \quad (3.20)$$

$$(P[u] \ \& \ L[u, v] \implies P[v]) \ \& \ (P[u] \ \& \ R[u, v] \implies P[v]). \quad (3.21)$$

Notice that giving truth values to the $P[u]$ variables (where $P[s] = 1$) reduces $\text{RRef}(F, s)$ to $\text{Ref}(F, s')$ where s' is the number of indices u for which $P[u] = 1$.

For an integer $k \geq 1$, we define $\text{R}^k\text{Ref}(F, s)$ as the formula resulting from substituting each variable $P[u]$ in $\text{RRef}(F, s)$ with the conjunction $\bigwedge_{i=1}^k P_i[u]$ for new variables $P_1[u], \dots, P_k[u]$. Note that $\text{RRef}(F, s) = \text{R}^1\text{Ref}(F, s)$.

Now, let $d, N \geq 1$ be integers, and let x be a propositional variable. We associate with x $N^{d-1} \lceil \sqrt{N}/2 \rceil$ new variables x_{i_1, \dots, i_d} , where $i_1, \dots, i_{d-1} \in [N]$ and $i_d \in [\lceil \sqrt{N}/2 \rceil]$. The fact that we make i_d range over $[\lceil \sqrt{N}/2 \rceil]$ instead of $[N]$ will be important later (specifically in Lemma 5.2). The depth- d Sipser functions for x are defined by

$$S_{d,N}^{\wedge}(x) \stackrel{\text{def}}{=} \bigwedge_{i_1=1}^N \bigvee_{i_2=1}^N \cdots \bigwedge_{i_d=1}^{\lceil \sqrt{N}/2 \rceil} x_{i_1, \dots, i_d},$$

$$S_{d,N}^{\vee}(x) \stackrel{\text{def}}{=} \bigvee_{i_1=1}^N \bigwedge_{i_2=1}^N \cdots \bigvee_{i_d=1}^{\lceil \sqrt{N}/2 \rceil} x_{i_1, \dots, i_d}$$

if d is odd, and

$$S_{d,N}^{\wedge}(x) \stackrel{\text{def}}{=} \bigwedge_{i_1=1}^N \bigvee_{i_2=1}^N \cdots \bigvee_{i_d=1}^{\lceil \sqrt{N}/2 \rceil} x_{i_1, \dots, i_d},$$

$$S_{d,N}^{\vee}(x) \stackrel{\text{def}}{=} \bigvee_{i_1=1}^N \bigwedge_{i_2=1}^N \cdots \bigwedge_{i_d=1}^{\lceil \sqrt{N}/2 \rceil} x_{i_1, \dots, i_d}$$

if d is even.

We define $\text{RRef}_{d,N}(F, s)$ to be the result of substituting every variable of the form $P[u]$ in $\text{RRef}(F, s)$ with $S_{d,N}^{\wedge}(P[u])$ and every other variable x with $S_{d,N}^{\vee}(x)$. Notice that $\text{RRef}_{d,N}(F, s)$ is a set of depth- $(d+1)$ formulas. But, as we want to prove statements about whether $\text{RRef}_{d,N}(F, s)$ has or does not have small depth- d refutations, we must write it as a set of depth- d formulas. We may do that with only a polynomial increase in size, as the only clauses of non constant size of $\text{RRef}(F, s)$ are those of the form $\neg P[u] \vee \bigvee_i X[u, i]$ corresponding to conditions (3.11), and these clauses will have depth- d after the substitution taking us from $\text{RRef}(F, s)$ to $\text{RRef}_{d,N}(F, s)$. Note that the conversion from $\text{RRef}_{d,N}(F, s)$ written as a set of depth- d formulas to its equivalent set of depth- $(d+1)$ formulas can be carried in tree-like depth- $(d+1)$ LK in linear time. In particular, a tree-like depth- $(d+1)$ LK refutation of the latter set can be turned into a tree-like depth- $(d+1)$ LK refutation of the former set, increasing the size by at most a factor of N^3 .

4 Upper bounds

We show in this section that if F is satisfiable, then $\text{RRef}_{d,N}(F, s)$ has small depth- d refutations:

► **Proposition 4.1.** *If F is a satisfiable CNF formula with n variables and m clauses, then there is a depth- d LK refutation of $\text{RRef}_{d,N}(F, s)$ of size*

$$S = O\left(\left(N^{d+3} s^2 n (m + s^2 n^3)\right)^2\right).$$

In particular, if $m = O(s^2 n^3)$, then $S = O(N^{2(d+3)} (sn)^8)$.

Proof. We start with a small depth-2 LK tree-like refutation of $\text{RRef}(F, s)$. This refutation will be such that after the substitution with Sipser functions, we get a depth- $(d+1)$ tree-like refutation of $\text{RRef}_{d,N}(F, s)$, which in turn we can convert to a depth- d DAG-like refutation of $\text{RRef}_{d,N}(F, s)$ by Proposition 2.2.

We write, for better readability, $A_1, \dots, A_k \rightarrow B_1, \dots, B_\ell$ instead of $\overline{A_1} \vee \cdots \vee \overline{A_k} \vee B_1 \vee \cdots \vee B_\ell$.

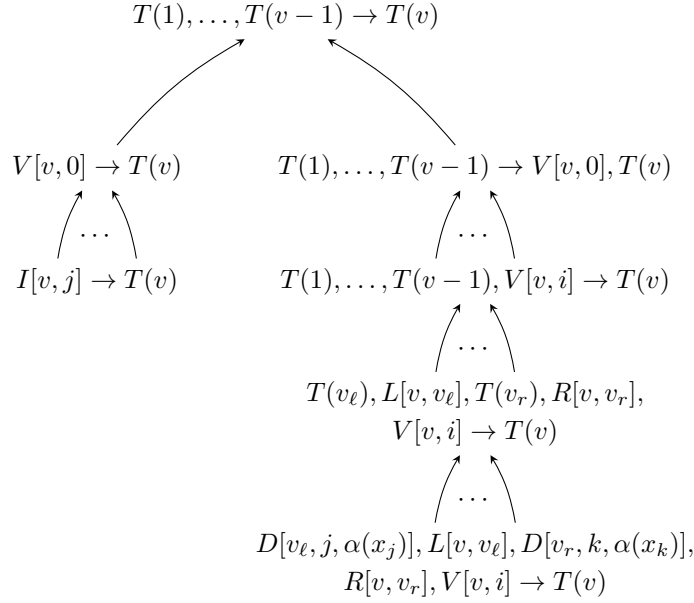
Let α be an assignment that satisfies every clause of F . We set

$$T(u) := P[u] \rightarrow \bigvee_{i=1}^n D[u, i, \alpha(x_i)].$$

What $T(u)$ says is that if $P[u]$ is true, then α satisfies the u -th clause in the refutation $\text{Ref}(F, s)$ describes.

Our refutation of $\text{RRef}(F, s)$ consists of $s - 1$ stages, starting with stage 0. In the u -th stage, $T(1), \dots, T(s - u) \rightarrow 0$ will have been derived. Then we can use this formula, along with a derivation of $T(1), \dots, T(s - u - 1) \rightarrow T(s - u)$, to derive $T(1), \dots, T(s - u - 1) \rightarrow 0$. In the $s - 1$ -th stage, $T(1) \rightarrow 0$ will have been derived, at which point we can reach a contradiction by deriving $T(1)$.

A derivation of $T(1), \dots, T(v - 1) \rightarrow T(v)$ is sketched in Figure 1. The formulas $I[v, j] \rightarrow$



■ **Figure 1** A sketch of a derivation of $T(1), \dots, T(v - 1) \rightarrow T(v)$.

$T(v)$ for $j \in [m]$ can be immediately derived from the clauses $P[u] \wedge I[v, j] \rightarrow D[v, i, \alpha(x_i)]$, which are clauses corresponding to condition (3.18), as the fact that α satisfies the clause C_j means that $x_i^{\alpha(x_i)}$ must belong to C_j for some i . These formulas can be in turn used along with the clauses (3.11) for $I[v, k]$ and (3.12) to derive $V[v, 0] \rightarrow T(v)$. Now deriving

$$T(1), \dots, T(v - 1) \rightarrow V[v, 0], T(v) \quad (4.1)$$

will allow us to derive $T(1), \dots, T(v - 1) \rightarrow T(v)$ by cutting on $V[v, 0]$. We can derive (4.1) from the formulas

$$T(1), \dots, T(v - 1), V[v, i] \rightarrow T(v) \quad (4.2)$$

for $i \in [n]$ using the clauses (3.11) for $V[v, i]$. The formulas (4.2) can be in turn derived from the formulas

$$T(v_\ell), L[v, v_\ell], T(v_r), R[v, v_r], V[v, i] \rightarrow T(v) \quad (4.3)$$

for $v_\ell, v_r \in [s]$ using the clauses (3.11) for $L[v, k]$ and $R[v, k]$, (3.12) and (3.13). Finally, (4.3) can be derived from the formulas

$$D[v_\ell, j, \alpha(x_j)], L[v, v_\ell], D[v_r, k, \alpha(x_k)], R[v, v_r], V[v, i] \rightarrow T(v), \quad (4.4)$$

for $j, k \in [n]$, which can be derived directly from the clauses (3.21) and either (3.14), (3.15) and (3.19) or (3.16) and (3.17) depending on whether $i = j = k$ or not.

22:10 Depth- d Frege Systems Are Not Automatable Unless $P = NP$

We can see that the derivations of $T(1)$, $\overline{T(s)}$ and $T(1), \dots, T(v-1) \rightarrow T(v)$ take at most $O(m + s^2n^3)$ steps, hence the overall refutation has size $O(s^2n(m + s^2n^3))$.

Now, notice that after substituting every variable $P[u]$ in it with $S_{d,N}^\wedge(P[u])$ and every other variable x with $S_{d,N}^\vee(x)$, $T(v)$ becomes a depth- d formula. Hence we see that after the substitution, the refutation described above becomes a depth- $(d+1)$ tree-like LK refutation of $\text{RRef}_{d,N}(F, s)$. We can then get a depth- d refutation of $\text{RRef}_{d,N}(F, s)$ of the required size by applying Proposition 2.2. \blacktriangleleft

5 Lower bounds

Lower bounds for depth- d Frege systems for $d > 1$, typically follow the following strategy:

1. We first show that the formulas we are trying to refute are robust; namely, after applying a restriction selected at random to them, then with high probability they cannot be refuted with proofs whose lines are, in a certain sense, simple.
2. Then we show, through the use of a switching lemma, that applying such a restriction to a short proof will result with high probability in a proof with simple lines.

Here we start with $\text{RRef}_{d,N}(F, s)$, which after applying the restrictions will collapse to $\text{Ref}(F, s')$, where s' is polynomially related to s . For the part of the overall strategy showing that there cannot be refutations with simple lines, we take, as in [2], simple to mean of small *index-width*. We say that a variable of the form $D[u, i, b]$, $V[u, i]$, $I[u, j]$, $L[u, v]$ or $R[u, v]$ *mentions* the index u . The index-width of a clause in the variables of $\text{Ref}(F, s)$ is defined as the number of indices mentioned by its variables, and the index-width of a resolution refutation of $\text{Ref}(F, s)$ is the maximum index-width over its clauses. We have:

► **Theorem 5.1** [2]. *For all integers $n, s > 0$ with $s \leq 2^n$, and every unsatisfiable CNF F with n variables, every resolution refutation of $\text{Ref}(F, s)$ has index-width at least $s/6n$.*

5.1 The robustness of $\text{RRef}_{d,N}$

We create a distribution on restrictions to the variables of $\text{RRef}_{d,N}(F, s)$ as follows. Suppose d is odd (if d were even, we would exchange the roles of 0 and 1 in the following construction). For each $S_{d,N}^\wedge(x)$ formula in $\text{RRef}_{d,N}(F, s)$, look at its bottom-most N^{d-1} \wedge connectives. For each such connective, we decide to “preserve” it with probability $1/\sqrt{N}$, and not to preserve it with probability $1 - 1/\sqrt{N}$. For each of the preserved connectives, we leave its first variable unset and set the rest to 1. For each variable in the unpreserved connectives, we set it to 0 or 1 with probability $1/2$ for each choice. The variables of $S_{d,N}^\vee(x)$ are set in the same way, except that the set variables of the preserved \vee connectives are set to 0 instead of 1.

Under such restrictions, Sipser functions do not simplify much. For formulas F and G , in which each variable appears only once, we say that F *contains* G if we can get G from F by deleting some of its literals and/or renaming some of its variables.

► **Lemma 5.2.** *For any $d \geq 2$, the probability that $S_{d,N}^\vee(x)|_\rho$, where $\nu \in \{\wedge, \vee\}$, does not contain $S_{d-1,N}^\nu(x)$ is at most $2^{-\Omega(\sqrt{N})}$.*

Proof. We show the lemma for $S_{d,N}^\wedge(x)$ and d odd. If $S_{d,N}^\wedge(x)|_\rho$ does not contain $S_{d-1,N}^\wedge(x)$, then either one of its bottom-most \wedge connectives takes the value 1, or in one of its depth-2 subformulas, less than $\sqrt{N}/2$ \wedge connectives are preserved. The probability that a bottom-most \wedge connective takes the value 1 is at most $2^{-\sqrt{N}/2}$ and the probability that this happens for at least one of the N^{d-1} bottom-most \wedge connectives is at most

$$N^{d-1}2^{-\sqrt{N}/2} \leq 2^{-\Omega(\sqrt{N})}.$$

Now fix a depth-2 subformula A of $S_{d,N}^\wedge(x)$. The expected number of preserved \wedge connectives in $A|_\rho$ is $N/\sqrt{N} = \sqrt{N}$, and by the Chernoff bound, the probability that there are less than $\sqrt{N}/2$ preserved \wedge connectives is at most $2^{-\Omega(\sqrt{N})}$. The probability that at least one of the N^{d-2} depth-2 subformulas of $S_{d,N}^\wedge(x)$ has less than $\sqrt{N}/2$ preserved connectives is thus at most

$$N^{d-2}2^{-\Omega(\sqrt{N})} \leq 2^{-\Omega(\sqrt{N})}.$$

We conclude that the probability that $S_{d,N}^\wedge|_\rho$ does not contain $S_{d-1,N}^\wedge$ is at most $2^{-\Omega(\sqrt{N})}$. ◀

5.2 The Furst-Saxe-Sipser switching lemma

Switching lemmas provide conditions under which a k -DNF formula “switches” to a ℓ -CNF formula after applying a restriction created at random. We will use the switching lemma of [8] and a variation tailored for $\text{R}^k\text{Ref}(F, s)$ due to [10].

Let G be a k -DNF formula over the set of variables X . Let X_1, \dots, X_r be a partition of X into r blocks, and let $\nu \in \{0, 1\}$. Consider the following distribution over restrictions on X : For each block X_i , we decide to “preserve” X_i with probability p , and not to preserve it with probability $1 - p$. For each preserved block, we leave one of its variables, say the first in the block, unset, and set all others to ν . For each unpreserved block, we set each of its variables to 0 or 1 with probability $1/2$ for each value.

We can extract the following lemma from [8, 19]. The lemma is implicit in [8, 19] with parameters obscured under a big O notation. We present it here in a more general, improved form, with explicit parameters, using decision trees along the lines of [18]. In what follows, \ln denotes the natural logarithm; we preserve the notation \log for the base 2 logarithm.

► **Lemma 5.3** (see [8, 19]). *If $phk2^k \ln N = o(N^{-\varepsilon})$ for some $\varepsilon \in (0, 1)$, then*

$$P[h(G|_\rho) > kh] \leq o(N^{-\varepsilon h}) \frac{2^{kh} - 1}{2^h - 1}.$$

Proof. The proof is by induction on k . If $k = 0$, G is a constant and can be represented by a decision tree of height 0. Suppose $k > 0$. We distinguish between two cases, G being wide and G being narrow. We call G wide if there are at least $h2^k \ln N$ terms in it such that no two of them contain variables from the same block. G is narrow if and only if it is not wide. If G is wide, then

$$\begin{aligned} P[h(G|_\rho) > kh] &\leq P[G|_\rho \neq 1] \leq \left(1 - \left(\frac{1-p}{2}\right)^k\right)^{h2^k \ln N} \\ &\leq e^{-(1-p)^k h \ln N} = N^{-(1-p)^k h} = o(N^{-\varepsilon h}). \end{aligned}$$

If G is narrow, then take a maximal set of terms such that no two of them contain variables from the same block, and let H be the set of blocks that contain a variable occurring in some term of this set. H contains at most $hk2^k \ln N$ blocks and every term of G contains some variable (or its negation) from some block in H . The probability of the event A that ρ preserves more than h blocks in H is

$$P[A] \leq \binom{hk2^k \ln N}{h} p^h \leq (hk2^k \ln N)^h p^h = o(N^{-\varepsilon h}).$$

22:12 Depth- d Frege Systems Are Not Automatable Unless $\mathbf{P} = \mathbf{NP}$

Now, let π be a restriction that sets the variables of all blocks in H , and let A_π be the event that π is consistent with ρ and $h((G|_\rho)|_\pi) > (k-1)h$. Notice that $G|_\pi$ is a $(k-1)$ -DNF, so by the induction hypothesis,

$$P[A_\pi] \leq P[h((G|_\pi)|_\rho) > (k-1)h] \leq o(N^{-\varepsilon h}) \frac{2^{(k-1)h} - 1}{2^h - 1}.$$

Notice that a restriction ρ that preserves at most h blocks is consistent with at most 2^h restrictions π , so we get

$$\begin{aligned} P \left[A \cup \bigcup_{\pi} A_\pi \right] &\leq o(N^{-\varepsilon h}) + o(N^{-\varepsilon h}) 2^h \frac{2^{(k-1)h} - 1}{2^h - 1} \\ &= o(N^{-\varepsilon h}) \frac{2^{kh} - 1}{2^h - 1}. \end{aligned}$$

In the event

$$\left(A \cup \bigcup_{\pi} A_\pi \right)^c,$$

i.e. the event that ρ preserves at most h blocks in H and for all restrictions π consistent with ρ , $h((G|_\rho)|_\pi) \leq (k-1)h$, we can construct a decision tree of height at most kh representing $G|_\rho$ as follows: We query all variables belonging to some block in H left unset by ρ (since ρ preserves at most h blocks in H , there are at most h of them), and at each branch π of the resulting tree, we append a decision tree of minimum height representing $(G|_\rho)|_\pi$. ◀

We create a distribution on restrictions on the variables of $\mathbf{R}^\ell \text{Ref}(F, s)$ as follows: For every index u and every $i \in [\ell]$, we set $P_i[u]$ to 0 or 1, with probability $1/2$ for each value. Let U be the set of indices such that $P_i[u] = 1$ for all $i \in [\ell]$. For each variable x of $\mathbf{R}^\ell \text{Ref}(F, s)$ not of the form $P_i[u]$ mentioning an index in U , we set x to 0 or 1, with probability $1/2$ for each value.

For a decision tree \mathbf{T} querying variables of $\text{Ref}(F, s)$, we define the *index-height* of \mathbf{T} as the maximum number of indices mentioned by variables over all branches that do not falsify axioms of $\text{Ref}(F, s)$. For a formula G , We denote by $\tilde{h}(G)$ the minimum index-height of a decision tree representing G .

The following lemma is from [10]. We give a proof because in [10] the lemma is stated not for $\mathbf{R}^\ell \text{Ref}(F, s)$ but a variation, plus we view the following proof to be simpler.

► **Lemma 5.4** [10]. *Let F be a CNF formula in n variables, k and ℓ integers with $0 < k \leq \ell$, and G a k -DNF formula over the variables of $\mathbf{R}^\ell \text{Ref}(F, s)$, where $s \leq 2^{\delta n}$ for some $\delta < 1$. Then for large enough n ,*

$$P[\tilde{h}(G|_\rho) > h] \leq 2^{-\frac{h}{n^{k-1}} \gamma(k)},$$

where $\gamma(0) = 1$, $\gamma(i) = (\log e)(i4^{i+1})^{-1} \gamma(i-1)$.

Proof. Let $h_i := h\gamma(i-1)/(4n^{i-1})$. We will show, by induction on k , that for every k and ℓ with $k \leq \ell$, for every k -DNF formula G over the variables of $\mathbf{R}^\ell \text{Ref}(F, s)$,

$$P \left[\tilde{h}(G|_\rho) > \sum_{i=1}^k h_i \right] \leq 2^{-\frac{h}{n^{k-1}} \gamma(k)}$$

for large enough n .

If $k = 0$, F is a constant and can be represented by a decision tree of height 0. Suppose $k > 0$. We call G wide if there are at least h_k/k terms in G over disjoint sets of indices, and call G narrow otherwise. Suppose G is wide. A literal in a term t of G is satisfied with probability at least $1/4$: Literals on a variable $P_i[u]$ are satisfied with probability $1/2$. For any other literal x^ϵ of t mentioning the index u , since $k \leq \ell$, there must be a variable $P_i[u]$ not in t , which is made 0 with probability $1/2$, in which case x^ϵ will be satisfied with probability $1/2$. Hence

$$\begin{aligned} P[\tilde{h}(G|_\rho) > h] &\leq P[G|_\rho \neq 1] \leq (1 - 4^{-k})^{\frac{h\gamma(k-1)}{4kn^{k-1}}} \\ &\leq 2^{-\frac{h}{n^{k-1}}(\log e)(k4^{k+1})^{-1}\gamma(k-1)} \\ &= 2^{-\frac{h}{n^{k-1}}\gamma(k)}. \end{aligned}$$

Suppose now that G is narrow. Take a maximal set of terms over disjoint sets of indices, and let H be the set of indices that are mentioned by the terms of this set. Notice that $|H| \leq h_k$ and that every term of G contains some variable (or its negation) that mentions an index in H . Let π be a restriction that

1. sets all variables mentioning an index in H and leaves all other variables unset, and
2. does not falsify any axioms of $\text{R}^\ell \text{Ref}(F, s)$.

The second condition means in particular that if U is the set of indices u for which π sets $P_i[u]$ to 1 for all i , then for all $u \in U$, there will be exactly one v such that $L[u, v]$ is true, exactly one v such that $R[u, v]$ is true, exactly one i such that $V[u, i]$ is true, and exactly one j such that $I[u, j]$ is true, making the total number of such π 's to be at most

$$S^{|U|} 2^{(|H|-|U|)n_0}$$

where $S := s^2(n+1)(m+1)2^{2n}$ and n_0 is the number of variables of $\text{R}^\ell \text{Ref}(F, s)$ mentioning a fixed index u .

Let A_π be the event that π is consistent with ρ and $\tilde{h}((G|_\rho)|_\pi) > \sum_{i=i}^{k-1} h_i$. We have that

$$\begin{aligned} P[A_\pi] &= P\left[\tilde{h}((G|_\rho)|_\pi) > \sum_{i=i}^{k-1} h_i \mid \rho \text{ con. with } \pi\right] P[\rho \text{ con. with } \pi] \\ &= P\left[\tilde{h}((G|_\pi)|_\rho) > \sum_{i=i}^{k-1} h_i\right] P[\rho \text{ con. with } \pi] \\ &\leq 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell|U|} 2^{-(|H|-|U|)n_0}. \end{aligned}$$

Hence, we get

$$\begin{aligned} P\left[\bigcup_{\pi} A_\pi\right] &\leq \sum_{\pi} P[A_\pi] \\ &\leq \sum_{U \subseteq H} S^{|U|} 2^{(|H|-|U|)n_0} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell|U|} 2^{-(|H|-|U|)n_0} \\ &= \sum_{r=0}^{|H|} \binom{|H|}{r} S^r 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell r} \\ &= (S/2^\ell + 1)^{|H|} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} \\ &\leq S^{|H|} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)}. \end{aligned}$$

22:14 Depth- d Frege Systems Are Not Automatable Unless $P = NP$

Since $s \leq 2^{\delta n}$ for some $\delta < 1$, the quantity

$$S^{|H|} = (s^2(n+1)(m+1)2^{2n})^{\frac{h}{4n^{k-1}}\gamma(k-1)}$$

will be at most $2^{\frac{\varepsilon h}{n^{k-2}}\gamma(k-1)}$ for some $\varepsilon < 1$ for large enough n , therefore

$$P \left[\bigcup_{\pi} A_{\pi} \right] \leq 2^{-\frac{h}{n^{k-1}}\gamma(k)}$$

for large enough n .

In the event $(\bigcup_{\pi} A_{\pi})^c$, that is the event that for every π consistent with ρ , $\tilde{h}((G|_{\rho})|_{\pi}) \leq \sum_{i=1}^{k-1} h_i$, we can construct a decision tree for $G|_{\rho}$ of index-height at most $\sum_{i=1}^k h_i$ as follows: We first query all variables mentioning an index in H left unset by ρ . Then, at each branch π of the resulting tree, we append a decision tree of minimum index-height representing $(G|_{\rho})|_{\pi}$. \blacktriangleleft

5.3 The lower bound for $\text{RRef}_{d,N}$

► **Theorem 5.5.** *For every integer $d > 0$, if F is an unsatisfiable CNF in n variables, N is an increasing function of n and s is a polynomial in n , every semantic depth- d refutation of $\text{RRef}_{d,N}(F, s)$ has size at least*

$$N^{\frac{1}{3}(\frac{\log s}{\log n} - 2)^{\frac{1}{d-1}}}$$

for large enough n .

Proof. Let $h := (1/3)(\log s / \log n - 2)^{1/(d-1)}$ and let G_1, \dots, G_t be a semantic depth- d refutation of $\text{RRef}_{d,N}(F, s)$ of size at most N^h . We assume that each G_i is either a literal or a disjunction of its immediate subformulas. Let A be a depth-1 subformula of some G_i . A is a 1-DNF or a 1-CNF formula, so applying Lemma 5.3 to it (or its negation respectively) with $k = 1$ and $p = N^{-1/2}$ and using as blocks X_1, \dots, X_r the variables in the depth-1 subformulas of $\text{RRef}_{d,N}(F, s)$, we get, since $N^{-1/2}3h \ln N = o(N^{-1/3})$,

$$P[h(A|_{\rho}) > 3h] = o(N^{-h}).$$

Now, there are at most N^h depth-1 subformulas A in the refutation, hence, by Lemma 5.2 and the union bound, the probability that either there is a depth-1 subformula A with $h(A|_{\rho}) > 3h$ or $\text{RRef}_{d,N}(F, s)|_{\rho}$ does not contain $\text{RRef}_{d-1,N}(F, s)$ is $o(1)$. Therefore, for large n , there must be a restriction ρ'_1 such that $\text{RRef}_{d,N}(F, s)|_{\rho'_1}$ contains $\text{RRef}_{d-1,N}(F, s)$ and all depth-1 subformulas of all $G_i|_{\rho'_1}$ are disjunctions or conjunctions of at most $3h$ literals. Let ρ_1 be a restriction extending ρ'_1 such that $\text{RRef}_{d,N}(F, s)|_{\rho_1}$ is exactly $\text{RRef}_{d-1,N}(F, s)$. We continue by applying Lemma 5.3 with $k = 3h$ and $p = N^{-1/2}$ to a $3h$ -CNF or $3h$ -DNF depth-2 subformula B of $G_i|_{\rho_1}$ to get

$$P[h(B|_{\rho}) > (3h)^2] = o(N^{-h}).$$

Since $G_i|_{\rho_1}$ has at most N^h depth-2 subformulas, there is a restriction ρ_2 such that $\text{RRef}_{d,N}(F, s)|_{\rho_1\rho_2}$ becomes $\text{RRef}_{d-2,N}(F, s)$ and all depth-2 subformulas of all $G_i|_{\rho_1}$ can be represented by decision trees of height at most $(3h)^2$. A formula representable by a decision tree of height at most $(3h)^2$ can be written as both a $(3h)^2$ -CNF and a $(3h)^2$ -DNF, so for all $i \in [t]$, $G_i|_{\rho_1\rho_2} \in \Sigma_{d-1}^{N^h, (3h)^2}$.

Repeating the same argument $d - 1$ times, applying Lemma 5.3 at the j -th time to depth-2 subformulas of $\Sigma_{d-j+1}^{N^h, (3h)^j}$ -formulas equivalent to $G_i|_{\rho_1 \dots \rho_{d-1}}$, we get restrictions $\rho_1, \dots, \rho_{d-1}$ such that $\text{RRef}_{d,N}(F, s)|_{\rho_1 \dots \rho_{d-1}}$ becomes $\text{RRef}_{1,N}(F, s)$ and for all $i \in [t]$, $G_i|_{\rho_1 \dots \rho_{d-1}} \in \Sigma_2^{N^h, (3h)^{d-1}}$.

We are now ready to apply Lemma 5.4. First notice that $\text{RRef}_{1,N}(F, s)$ contains $\text{R}^\ell \text{Ref}(F, s)$ for large n , where $\ell := (3h)^{d-1}$. For ρ selected randomly as specified in Lemma 5.4 for this ℓ , we get that the expected number of active indices is $s/2^\ell$, hence $\text{RRef}_{1,N}(F, s)|_\rho$ contains $\text{Ref}(F, s')$, where $s' := s/2^{\ell+1}$, with high probability. Furthermore, Lemma 5.4 gives

$$P \left[\tilde{h}(C|_\rho) > n^{(3h)^{d-1}} \right] \leq 2^{-\Omega(n)},$$

where C is a $(3h)^{d-1}$ -DNF formula equivalent to some $G_i|_{\rho_1 \dots \rho_{d-1}}$. Therefore there must be a restriction ρ_d such that $\text{RRef}_{d,N}|_{\rho_1 \dots \rho_d}$ becomes $\text{Ref}(F, s')$ and for every $i \in [t]$, $\tilde{h}(G_i|_{\rho_1 \dots \rho_{d-1}}) \leq n^{(3h)^{d-1}}$. Applying now the construction of Lemma 2.3¹ to $G_1|_{\rho_1 \dots \rho_{d-1}}, \dots, G_t|_{\rho_1 \dots \rho_{d-1}}$ gives a resolution refutation of $\text{Ref}(F, s')$ of index-width at most $3n^{(3h)^{d-1}} = 3s/n^2$, contradicting Theorem 5.1 for large n . \blacktriangleleft

6 Non-automatability of bounded-depth Frege systems

► **Theorem 6.1.** *If $P \neq NP$, then depth- d Frege systems are not automatable.*

Proof. Suppose there is an algorithm **A** which, given an unsatisfiable CNF formula G , returns a depth- d refutation of G in time polynomial in $S(G) + S$, where $S(G)$ is the size of G and S the size of the smallest depth- d refutation of G . Let $c, n_0 \geq 1$ be integers such that for every G with $|G| \geq n_0$, **A** runs in time at most $(S(G) + S)^c$. We will use **A** to decide in polynomial time whether 3-SAT is satisfiable. Given a 3-CNF formula F with n variables (and thus of size $O(n^3)$), we construct the formula $G := \text{RRef}_{d,N}(F, s)$, where $s := n^{(3h)^{d-1}+2}$, $N := s$ and h is an integer such that

$$((3h)^{d-1} + 2)h > c(((3h)^{d-1} + 2)(2(d+3)) + 8((3h)^{d-1} + 3) + 1).$$

Notice that the left hand side of the above inequality is a polynomial of degree d in h and the right hand side a polynomial of degree $d - 1$, hence such an h must exist. Since N and s are polynomials in n , the size of G is polynomial in n , hence its construction takes polynomial time. Let S be the size of the smallest depth- d refutation of G and let $n_1 \geq n_0$ be an integer such that for all $n \geq n_1$,

$$F \text{ satisfiable} \implies S + S(G) \leq n^{((3h)^{d-1}+2)(2(d+3))+8((3h)^{d-1}+3)+1},$$

$$F \text{ not satisfiable} \implies S \geq n^{((3h)^{d-1}+2)h}.$$

Here we use the bounds given by Proposition 4.1 and Theorem 5.5. To decide whether F is satisfiable, if $n < n_1$, then we check all possible assignments to its variables to see if there is a satisfying one. Otherwise, we run **A** on G for

$$n^{c(((3h)^{d-1}+2)(2(d+3))+8((3h)^{d-1}+3)+1)}$$

steps. If **A** stops, then we can assert that F is satisfiable; otherwise we can assert that F is unsatisfiable. \blacktriangleleft

¹ Lemma 2.3 is stated for height and width, but it is not hard to see that the same construction yields the lemma with index-height and index-width instead of height and width respectively.

7 Conclusion

This paper shows the non-automatability of bounded-depth Frege system assuming $P \neq NP$. We do this, following [2], by constructing, given a CNF formula F , a formula $\text{RRef}_{d,N}(F, s)$, and exhibiting a gap between the size of the shortest depth- d Frege refutations of $\text{RRef}_{d,N}(F, s)$ when F is satisfiable and the size of the shortest depth- d Frege refutations of $\text{RRef}_{d,N}(F, s)$ when F is not satisfiable.

To show the lower bound for depth- d Frege refutations of $\text{RRef}_{d,N}(F, s)$ in the case F is not satisfiable, we employ the Furst-Saxe-Sipser switching lemma [8]. While sufficient for the purpose of showing non-automatability assuming $P \neq NP$, this can only give lower bounds of the form n^h , where h is a barely superconstant function of n . It would be nice to have an exponential lower bound. In particular, as in [2], an exponential lower bound would rule out the automatability of bounded-depth Frege systems in quasipolynomial time unless NP problems can be solved in quasipolynomial time, and their automatability in subexponential time unless NP problems can be solved in subexponential time.

$\text{RRef}_{d,N}(F, s)$ consists of formulas of depth d . In particular, this does not preclude the possibility of bounded-depth Frege systems being automatable on refuting, say CNF formulas. A natural question is whether we could use CNFs, or at least formulas of constant depth, not depending on d , instead. Let us mention here that whether there is a constant depth formula exponentially separating depth- d from depth- $(d + 1)$ Frege is open as well; currently, only a super-polynomial separation is known [13] (see also [15, Section 14.5]). Moreover, the formulas $\text{RRef}_{d,N}(F, s)$ are ad hoc and rather artificial. It would be nice if one could establish a lower bound for formulas $\text{Ref}_d(F, s)$ for an unsatisfiable formula F , encoding the fact that there are depth- d refutations of F of size s (see Problem 2 in [17]), showing that proving lower bounds for a depth- d Frege system is hard within the system. The latter problem for a proof system is considered by Pudlák [17] to be a more important question than the question of whether the system is automatable. Note that a CNF encoding of $\text{Ref}_d(F, s)$ is a candidate formula for the question of whether bounded-depth Frege systems for refuting CNFs are automatable, and a CNF encoding of the reflection principle $\text{Sat}(F, v) \wedge \text{Ref}_d(F, s)$, where $\text{Sat}(F, v)$ encodes that v is an assignment satisfying F , is a candidate formula for the depth- d vs depth- $(d + 1)$ Frege problem (see [17]).

Finally, the non-automatability result of [2] has been shown for cutting planes [11], $\text{Res}(k)$ [10], and various algebraic proof systems [7]. As far as we know, two remaining open cases are the sum of squares and Sherali-Adams proof systems.

References

- 1 Michael Alekhovich and Alexander Razborov. Resolution is not automatizable unless $W[P]$ is tractable. *SIAM Journal of Computing*, 38:1347–1363, 2008.
- 2 Albert Atserias and Moritz Müller. Automating resolution is NP-hard. *Journal of the ACM*, 67:31:1–31:17, 2020.
- 3 Arnold Beckmann and Samuel Buss. Separation results for the size of constant-depth propositional proofs. *Annals of Pure and Applied Logic*, 136:30–55, 2005.
- 4 Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth frege proofs. *Computational Complexity*, 13:47–68, 2004.
- 5 Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for frege systems. *SIAM Journal of Computing*, 29:1939–1967, 2000.
- 6 Stefan Dantchev and Søren Riis. On relativisation and complexity gap. In *Proceedings of the 12th Annual Conference of the EACSL*, pages 142–154, 2003.

- 7 Susanna de Rezende, Mika Göös, Jakob Nordström, Toniann Pitassi, Robert Robere, and Dmitry Sokolov. Automating algebraic proof systems is NP-hard. In *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing*, pages 209–222, 2021.
- 8 Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.
- 9 Michal Garlík. Resolution lower bounds for refutation statements. In *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science*, volume 138, pages 37:1–37:13, 2019.
- 10 Michal Garlík. Failure of feasible disjunction property for k-DNF resolution and NP-hardness of automating it. *Electronic Colloquium on Computational Complexity*, 2020.
- 11 Mika Göös, Sajin Korothe, Ian Mertz, and Toniann Pitassi. Automating cutting planes is NP-hard. In *Proceedings of the 52nd Annual ACM Symposium on Theory of Computing*, pages 68–77, 2020.
- 12 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.
- 13 Russell Impagliazzo and Jan Krajíček. A note on conservativity relations among bounded arithmetic theories. *Mathematical Logic Quarterly*, 48:375–377, 2002.
- 14 Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *Journal of Symbolic Logic*, 59:73–86, 1994.
- 15 Jan Krajíček. *Proof Complexity*. Cambridge University Press, 2019.
- 16 Pavel Pudlák. On reducibility and symmetry of disjoint NP pairs. *Theoretical Computer Science*, 295:323–339, 2003.
- 17 Pavel Pudlák. Reflection principles, propositional proof systems, and theories, 2020. arXiv:2007.14835. [arXiv:2007.14835](https://arxiv.org/abs/2007.14835).
- 18 Nathan Segerlind, Samuel Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for k-DNF resolution. *SIAM Journal of Computing*, 33:1171–1200, 2004.
- 19 Michael Sipser. Borel sets and circuit complexity. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 61–69, 1983.

Exponential Separation Between Powers of Regular and General Resolution over Parities

Sreejata Kishor Bhattacharya ✉

Tata Institute of Fundamental Research, Mumbai, India

Arkadev Chattopadhyay ✉

Tata Institute of Fundamental Research, Mumbai, India

Pavel Dvořák ✉

Tata Institute of Fundamental Research, Mumbai, India

Charles University, Prague, Czech Republic

Abstract

Proving super-polynomial lower bounds on the size of proofs of unsatisfiability of Boolean formulas using resolution over parities is an outstanding problem that has received a lot of attention after its introduction by Itsykson and Sokolov [11]. Very recently, Efremenko, Garlík and Itsykson [7] proved the first exponential lower bounds on the size of ResLin proofs that were additionally restricted to be *bottom-regular*. We show that there are formulas for which such regular ResLin proofs of unsatisfiability continue to have exponential size even though there exist short proofs of their unsatisfiability in ordinary, non-regular resolution. This is the first super-polynomial separation between the power of general ResLin and that of regular ResLin for any natural notion of regularity.

Our argument, while building upon the work of Efremenko et al. [7], uses additional ideas from the literature on *lifting theorems*.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof Complexity, Regular Reslin, Branching Programs, Lifting

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.23

Related Version *Full Version*: <https://arxiv.org/abs/2402.04364>

Funding *Arkadev Chattopadhyay*: Partially funded by the Department of Atomic Energy and a Google India Research Award.

Pavel Dvořák: Work done entirely at TIFR. Supported by Czech Science Foundation GAČR grant #22-14872O.

1 Introduction

One of the most basic and well studied proof systems in propositional proof complexity is resolution. Its weakness by now is reasonably well understood after years of research. Yet, this understanding is quite fragile as natural and simple strengthenings of resolution quickly pose challenges that remain outstanding. One such system is resolution over linear equations, introduced by Raz and Tzameret [16], which has been abbreviated as ResLin. In this paper, we study ResLin over \mathbb{F}_2 that was introduced by Itsykson and Sokolov [11] (for the brevity we use only ResLin for ResLin over \mathbb{F}_2 as we do not consider any other field). More precisely, a *linear* clause is a disjunction of affine equations, generalizing the notion of ordinary clauses. If A and B are two such linear disjunctions and ℓ is a linear form, then the inference rule of ResLin derives the linear clause $A \vee B$ from clauses $A \vee (\ell = 1)$ and $B \vee (\ell = 0)$. To appreciate the power of this system, let us recall that a linear clause, unlike an ordinary clause, can be expressed using many different bases. Indeed, no super-polynomial lower bounds on the size of general proofs in this system is currently known for any explicit unsatisfiable Boolean formula.



© Sreejata Kishor Bhattacharya, Arkadev Chattopadhyay, and Pavel Dvořák;

licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 23; pp. 23:1–23:32



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Progress was first made in the work of Itsykson and Sokolov [11] when they proved exponential lower bounds on the size of tree-like ResLin proofs for central tautologies including the Pigeonhole Principle and Tseitin formulas over expanding graphs lifted with the AND gadget. Further, Itsykson and Sokolov [12] showed that tree-like ResLin proofs are exponentially weaker than general ResLin proofs. Proving such lower bounds and separations was systematized, only recently, in the independent works of Chattopadhyay, Mande, Sanyal and Sherif [4] and that of Beame and Koroth [2].

In the world of ordinary resolution, it is known that there is an intermediate proof system, known as *regular* resolution, whose power strictly lies in between tree-like and general proofs. In the graph of a regular resolution proof, no derivation path from an axiom clause to the final empty clause resolves a variable of the formula more than once. In the dual view of searching for a falsified clause, this corresponds precisely to read-once branching programs, where no source (corresponding to the empty clause) to sink (corresponding to a falsified clause of the formula) path queries a variable more than once. Taking cue from this, Gryaznov, Pudlák and Talebanfard [10] introduced models of read-once linear branching programs (ROLBP), to capture notions of regularity in ResLin. They identified two such notions that extend the notion of regularity in ordinary resolution, or the read-once property of branching programs. Consider a node v of an ROLBP. Let $Pre(v)$ denote the vector space spanned by all the linear queries that appear in some path from the source node to v . Similarly, let $Post(v)$ denote the space spanned by all linear queries that lie in some path from v to a sink node. In the most restrictive notion, called strongly regular proofs or strongly read-once linear branching programs, $Pre(v)$ and $Post(v)$ have no non-trivial intersection for every v . In a more relaxed notion, called weak regularity or weakly ROLBP, the linear query made at node v is not contained in $Pre(v)$. Gryaznov et al. [10] were able to prove an exponential lower bound on the size of strongly ROLBP for computing a *function*, using the notion of directional affine dispersers. However, their argument is not known to work for search problems for even strongly ROLBP.

There is another natural notion of weakly read-once linear branching programs (dually, weakly regular ResLin) that complements the notion defined in [10]. In this notion, we forbid the linear query made at a node v of the branching program to lie in an affine space $Post(u)$, for each u that is a child of v . We will call this notion bottom-read-once (bottom-regular proofs) and the former notion of Gryaznov et al. as top-read-once (top-regular proofs). Both are generalizations of strongly read-once linear branching programs (strongly regular ResLin proofs). Very recently, Efremenko, Garlik and Itsykson [7] proved the first exponential lower bounds on the size of bottom-regular ResLin proofs. The tautology they used was the Binary Pigeonhole Principle (BPHP). It is plausible that the BPHP remains hard for general ResLin proofs. One way to prove such a bound would be to show that every general ResLin proof could be converted to a bottom-regular ResLin proof with a (quasi-)polynomial blow up.

Our main result is a strong refutation of that possibility. We show that bottom-regular ResLin proofs require exponential size blow-up to simulate non-regular proofs even in the ordinary resolution system, that uses only ordinary clauses and only variables (instead of arbitrary linear forms) are resolved. The formulas we use are twists of certain formulas used by Alekhovich, Johannsen, Pitassi and Urquhart [1] for providing separation between regular and general resolution. Alekhovich et al. provided two formulas for proving such a separation. In the second one, the starting point is a pebbling formula on pyramid graphs. It turns out that they are easy for regular resolution. To get around that, they consider *stone formula* for pyramid graphs, that they prove is hard for regular resolution while remaining easy for general resolution. We further obfuscate such stone formulas using another idea of Alekhovich et al. that appears in the construction of their first formula. Finally, we *lift* these formulas by logarithmic size Inner-product (IP) gadgets.

This lifted formula presents fresh difficulties to be overcome to carry out the implicit technique of Efremenko et al. [7]. We overcome them by exploiting two properties of the Inner-product. First, we exploit the property that IP has low discrepancy, invoking a result of Chattopadhyay, Filmus, Korothe, Meir and Pitassi [3]. Second, we use the fact that IP has the stifling property, inspired by the recent work of Chattopadhyay, Mande, Sanyal and Sherif [4].

Our unsatisfiable formula that yields a separation of resolution and bottom-regular ResLin is a stone formula for a pyramid graph G_n of n levels lifted by an inner-product gadget $\text{IP} : \{0, 1\}^b \rightarrow \{0, 1\}$. We denote this formula as $\text{SP}_n \circ \text{IP}$ and it is defined over $M := 2N^2 \cdot b$ variables, where $b = \Theta(\log n)$ and $N = n(n+1)/2$ is the number of vertices of the pyramid graph G_n . For exact definition of $\text{SP}_n \circ \text{IP}$, see Section 2.

► **Theorem 1.** *The formula $\text{SP}_n \circ \text{IP}$ admits a resolution refutation of length that is polynomial in M but any bottom-regular ResLin refutation of it must have length at least $2^{\Omega(M^{1/12}/\log^{1+\varepsilon} M)}$ for $\varepsilon = 1/12$.*

Comparison with Efremenko et al. [7]

Our work builds upon the very recent work of Efremenko, Garlík and Itsykson, who proved an exponential lower bound for the regular linear resolution complexity of the formula Binary-PHP $_n^{n+1}$. A crucial property of this formula that they use is that if we sample an assignment to the variables from the uniform distribution, with high probability one needs to make at least $n^{\Omega(1)}$ bit-queries to locate a falsified clause. Later, they use the following simple property of the uniform distribution: let A be an affine subspace of co-dimension r . Then, the probability mass of A under uniform distribution is very small (inverse-exponential in r). Call this property (*).

Our goal is to show an exponential lower bound on the regular linear resolution complexity of a formula that has a small resolution refutation. A candidate formula would be a CNF which exhibits exponential separation between resolution and regular resolution. Some such formulas are MGT $_{n,\rho}$ and *stone formulas* with auxiliary variables to keep width of clauses short (both defined in Alekhovich et al. [1]). However, all such formulas have constant width – and therefore, a uniformly random assignment falsifies a constant fraction of clauses. It follows that for both these formulas there is a query algorithm making only constantly many queries which finds a falsified clause with high probability under the uniform distribution. Thus, directly adapting the argument of Efremenko et al. would not work for these formulas.

Our main observation is that property (*) continues to hold for a much larger class of distributions than the uniform distribution when the base formula is *lifted* with an appropriate gadget. More precisely, if we take any distribution μ on the assignments of the base formula and let μ' be its *uniform lift*, property (*) holds for μ' . We are now free to choose *any* distribution on the assignments of the base formula for which locating a falsifying axiom requires many queries on average (this is just a sketch; we actually need something slightly stronger). This gives us enough freedom to construct appropriate distributions. Some more ingredients are required to make this idea work; we explain them in the subsequent sections.

Some Other Related Work

Following up on the work by Alekhovich et al. [1], Urquhart [20] proved a stronger separation between the length of regular and general resolution proofs. Much more recently, Vinyals, Elffer, Johannsen and Nordström [21], designed a different formula for showing an even stronger separation between regular and general resolutions. The constructions

of Urquhart [20] as well as that of Vinyals et al. [21] are somewhat related to the hard formulas that we construct in this paper. We talk about them more at the end of Section 2 after describing our construction in detail. In another direction, the model of read-once linear branching programs, introduced by Gryaznov, Pudlák and Talebanfard [10], spawned research in directional affine extractors and pseudo-randomness first by Chattopadhyay and Liao [6], and then further by Li and Zhong [14], and by Li [13]. These work on extractors, while independently interesting, are not known to have consequences for ResLin.

Organization of the Paper

We present our hard formula in the next section and briefly compare it with constructions done in earlier work. In Section 3, we define ResLin refutation system and its regular and tree-like restrictions. Further, we present the connection between resolution proof systems and branching programs. In Section 4, we present some results from linear algebra which we use in our proofs. In Section 5, we sketch the outline and main ideas of the proof of our main result, Theorem 1. Then in Section 6, we prove the upper bound part of Theorem 1, i.e., our hard formula has short resolution refutation. We finish our proof in Section 7 where we prove the lower bound part of Theorem 1, i.e., any bottom-regular ResLin refutation of our hard formula must have exponential length. Finally, in Section 8, we conclude with some of the many open problems that our work raises.

2 A Formula Hard For Just Regular ResLin

Let us first recall the stone formula that was used by Alekhovich et al. [1] for separating the powers of regular and general resolution. The formula that we shall use is a *lift* of this formula by an appropriate gadget. Let $G = (V, E)$ be a directed acyclic graph such that it has exactly one root (vertex with indegree 0), r , and every vertex of G has outdegree either 0 or 2. Call the vertices with outdegree 0 the sinks of G . Let $|V| = N$. We describe the *stone formula on G twisted with ρ* , $\text{Stone}(G, \rho)$ below. In words, the contradiction we are about to describe states the following:

- There are $|V|$ stones. Each stone has a color: red or blue.
- At least one stone must be placed on each vertex.
- All stones placed on sinks must be red.
- All stones placed on the root must be blue.
- Let v be a node with out-neighbors u, w . If a red stone j is placed on u and a red stone k is placed on w , then all stones placed on v must be red.

We shall twist this formula with an obfuscation map ρ to make it hard for regular resolution. We formally define the formula below. We introduce the following set of variables.

Vertex variables: For all $v \in V, 1 \leq j \leq N : P_{v,j}$.

Semantic interpretation: $P_{v,j}$ is set to 1 iff stone j is placed on vertex v .

Stone variables: For all $1 \leq j \leq N : R_j$.

Semantic interpretation: R_j is set to 1 if stone j is colored red, otherwise it is set to 0.

Auxiliary variables: For all $v \in V, 1 \leq j \leq N - 1 : Z_{v,j}$.

Semantic interpretation: These are auxiliary variables used to encode the fact that at least one stone is placed on vertex v , with a bunch of constant-width clauses.

Let \mathcal{V} denote the set of all variables mentioned above and $\rho : [N]^3 \rightarrow \mathcal{V}$ be an arbitrary mapping that we call an *obfuscation map*. Let S be the set of sinks of G . We define $\text{Stone}(G, \rho)$ to be the formula comprising the following set of clauses:

Root clauses: For all $1 \leq j \leq N$: $\neg P_{r,j} \vee \neg R_j$

Semantic interpretation: All stones placed on the root r of G must be coloured blue.

Sink clauses: For all $1 \leq j \leq N, s \in S$: $\neg P_{s,j} \vee R_j$

Semantic interpretation: Each stone placed on a sink of G must be coloured red.

Induction clauses: For all $v \in V(G)$ with out-neighbors u, w and for each $i, j, k \in [N]$:

$$\neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i, j, k)$$

$$\neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg \rho(i, j, k)$$

Semantic interpretation: after resolving out the variable $\rho(i, j, k)$, the clause says that if the stones placed on u and w are colored red, the stone placed at v must also be colored red, i.e., the implication

$$(P_{u,i} \wedge R_i \wedge P_{w,j} \wedge R_j \wedge P_{v,k}) \implies R_k.$$

Stone-placement clauses: For all $v \in V(G)$:

$$P_{v,1} \vee \neg Z_{v,1}$$

$$Z_{v,1} \vee P_{v,2} \vee \neg Z_{v,2}$$

...

$$Z_{v,N-2} \vee P_{v,N-1} \vee \neg Z_{v,N-1}$$

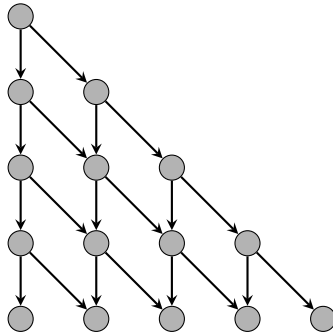
$$Z_{v,N-1} \vee P_{v,N}$$

Semantic interpretation: Together, the clauses are equivalent to

$$P_{v,1} \vee P_{v,2} \vee \dots \vee P_{v,N}$$

i.e., at least one stone is placed on the vertex v .

Let G_n be the pyramid graph on n levels. The vertex set is $V = \{(i, j) | 1 \leq i \leq n, 1 \leq j \leq i\}$. The *level* of a vertex (i, j) is defined to be its first coordinate i . The edge set is $E = \{(i, j) \rightarrow (i+1, j) | 1 \leq i < n, 1 \leq j \leq i\} \cup \{(i, j) \rightarrow (i+1, j+1) | 1 \leq i < n, 1 \leq j \leq i\}$. See Figure 1, for an example of the pyramid graph. The sinks of G_n are the vertices at layer n , i.e., (n, i) for $1 \leq i \leq n$. The root is $(1, 1)$. We have $|V| = N = \frac{1}{2}n(n+1)$.



■ **Figure 1** An example of the pyramid graph with $n = 5$ levels.

We instantiate the stone formula with $G = G_n$. Let $\text{SP}_{n,\rho} = \text{Stone}(G_n, \rho)$. We denote the number of variables of $\text{SP}_{n,\rho}$ by m , i.e. $|\mathcal{V}| = m = N^2 + N + N(N-1) = 2N^2$. In order to prove our lower bound against regular ResLin, it turns out to be convenient working

with a formula that is obtained by *lifting* $\text{SP}_{n,\rho}$. Let $g : \{0, 1\}^b \rightarrow \{0, 1\}$ be a Boolean function, called gadget. For $a \in \{0, 1\}$, we denote the set of all pre-images of a by $g^{-1}(a)$, i.e., $g^{-1}(a) := \{x \in \{0, 1\}^b \mid g(x) = a\}$.

Let $c_1, c_2, \dots, c_k \in \{0, 1\}$. Let $C = [X_1 = c_1] \vee \dots \vee [X_k = c_k]$ be a clause over variables X_1, \dots, X_k . Here $[X_i = c_i]$ denotes a literal, i.e. X_i if $c_i = 1$, and $\neg X_i$ otherwise. To lift Clause C we introduce b variables Y_1^i, \dots, Y_b^i for each variable X_i of C . The lift of C , $C \circ g$, is a set of clauses which, in conjunction, are semantically equivalent to $[g(Y_1^1, \dots, Y_b^1) = c_1] \vee \dots \vee [g(Y_1^k, \dots, Y_b^k) = c_k]$, i.e., the following:

$$C \circ g := \left\{ \bigvee_{1 \leq i \leq k, 1 \leq j \leq b} [Y_j^i = 1 - d_j^i] \mid d^1 \in g^{-1}(1 - c_1), \dots, d^k \in g^{-1}(1 - c_k) \right\},$$

where each d^i is a b -bit string and d_j^i is its j -th bit.

► **Observation 2.** *An assignment $(Y_j^i)_{1 \leq i \leq k, 1 \leq j \leq b}$ satisfies every clause in $C \circ g$ if and only if the lifted assignment (x_1, x_2, \dots, x_k) given by*

$$x_1 = g(Y_1^1, \dots, Y_b^1), \dots, x_k = g(Y_1^k, \dots, Y_b^k)$$

satisfies the clause C .

For a technical reason, we shall also need the following simple observation:

► **Observation 3.** *Let ψ be any clause of $C \circ g$. Suppose, one of the variables Y_j^i appears in ψ . Then, for all $k \in [b]$, the variable Y_k^i also appears in ψ .*

For a set of clauses Φ (a CNF formula), we define its lift as $\Phi \circ g := \cup_{C \in \Phi} (C \circ g)$. We have the following corollary of Observation 2.

► **Corollary 4.** *The set of clauses Φ is unsatisfiable if and only if the set of clauses $\Phi \circ g$ is unsatisfiable.*

We remark that if the base set Φ contains only clauses of width at most k , then $\Phi \circ g$ contains clauses of width at most kb and $|\Phi \circ g| \leq 2^{bk} \cdot |\Phi|$. In particular, a constant-width, poly-size unsatisfiable formula, when lifted by an $O(\log n)$ size gadget, yields an $O(\log n)$ -width, poly-size unsatisfiable formula. Our hard formula will be the stone formula lifted by an inner product gadget $\text{SP}_{n,\rho} \circ \text{IP}$, where $\text{IP} : \{0, 1\}^b \rightarrow \{0, 1\}$ is the inner-product function, i.e. for each $x, y \in \{0, 1\}^{b/2}$, set $\text{IP}(x, y) = x_1 y_1 + \dots + x_{b/2} y_{b/2} \pmod{2}$ where $b = O(\log n)$ is an even integer.

Comparison with related formulas from previous work

We briefly mention some formulas that were used in the past, that are related to $\text{Stone}(G_n, \rho)$. First, the formula sans obfuscation $\text{Stone}(G_n)$ was one of the formulas used by Alekhovich et al. [1] to yield the first exponential separation between regular and general resolution. This separation was further strengthened by Urquhart [20] in a follow-up work, using the following more involved formula. Fix a bijective placement of stones to the vertices of G_n . This reduces $\text{Stone}(G_n)$ to a plain pebbling formula on pyramid graph, denoted by $\text{Peb}(G_n)$. This reduces the number of clauses to linear in N . Urquhart considers the 2-bit XOR lift of such a formula, i.e. $\text{Peb}(G_n) \circ \oplus$. This formula blows up the number of clauses, but still keeps the number of variables to $O(N)$. Finally, he shows that for a suitable ρ , the obfuscation of $\text{Peb}(G_n) \circ \oplus$ by ρ , just as we obfuscate $\text{Stone}(G_n)$ by $\text{Stone}(G_n, \rho)$, yields a separation between regular and general resolution that is stronger than the one by Alekhovich et al. [1].

More recently, Vinyals et al. [21] worked with a different *sparsification* of $\text{Stone}(G_n)$. This comes about naturally by considering $\text{Stone}(G_n)$ as a densification of $\text{Peb}(G_n)$ by using a complete bi-partite graph with N vertices on each side. Roughly speaking, Vinyals et al. used a constant degree bi-partite expander gadget with $\text{Peb}(G_n)$, inspired by the earlier work of Razborov [18]. This results in more modular and optimal arguments. However, the lift by a stifled gadget of a base stone formula, like we do as in $\text{Stone}(G_n, \rho) \circ \text{IP}$, seems not to have been considered earlier.

3 Resolution Proof Systems and Branching Programs

A proof in a propositional proof system starts from a set of clauses Φ , called axioms, that is purportedly unsatisfiable. It generates a proof by deriving the empty clause from the axioms, using inference rules. The main inference rule in the standard resolution, called the resolution rule, derives a clause $A \vee B$ from clauses $A \vee x$ and $B \vee \neg x$ (i.e., we resolve the variable x). If we can derive the empty clause from the original set Φ then it proves the set Φ is unsatisfiable. We will need the following basic and well known fact that states resolution is complete without being too inefficient.

► **Lemma 5.** *Let C be any clause, and Φ be any CNF formula over n Boolean variables and of polynomial size, that semantically implies C . Then, C can be derived from Φ by a resolution proof of size at most $2^{O(n)}$.*

Linear resolution, aka ResLin and introduced by Raz and Tzameret [16], is a generalization of standard resolution, using linear clauses (disjunction of linear equations over \mathbb{F}_2) to express lines of a proof. It consists of two rules:

Resolution Rule: From linear clauses $A \vee (\ell = 0)$ and $B \vee (\ell = 1)$ derive a linear clause $A \vee B$.

Weakening Rule: From a linear clause A derive a linear clause B that is semantically implied by A (i.e., any assignment satisfying A also satisfies B).

The length of a resolution (or ResLin) refutation of a formula Φ is the number of applications of the rules above in order to refute the formula Φ . The width of a resolution (or ResLin) refutation is the maximum width of any (linear) clause that is used in the resolution proof.

It is known that a resolution proof and a linear resolution proof, for an unsatisfiable set of clauses Φ , correspond to a branching program and a linear branching program, respectively, for a search problem $\text{Search}(\Phi)$ (see for example Garg et al [8], who credit it to earlier work of Razborov [17] that was simplified by Pudlák [15] and Sokolov[19]) that is defined as follows. For a given assignment α of the n variables of Φ , one needs to find a clause in Φ that is unsatisfied by α (at least one exists as the set Φ is unsatisfiable). A linear branching program computing a search problem $\text{P} \subseteq \mathbb{F}_2^n \times O$ is defined as follows.

- There is a directed acyclic graph \mathcal{P} of one source and some sinks. Each non-sink node has out-degree at most two. For an inner node v the two out-neighbors u and w (i.e., there are edges (v, u) and (v, w) in \mathcal{P}) are called children of v .
- Each node v of \mathcal{P} is labeled by an affine space $A_v \subseteq \mathbb{F}_2^n$.
- The source is labeled by \mathbb{F}_2^n .
- Let v be a node of out-degree 2 and u and w be children of v . Then, $A_u = A_v^0$ and $A_w = A_v^1$, where $A_v^c = \{x \in A_v \mid \langle f_v, x \rangle = c\}$ for a linear query $f_v = \mathbb{F}_2^n$ and $c \in \{0, 1\}$. We call such v a *query node*.
- Let v be a node of out-degree 1 and u be the child of v . Then, $A_v \subseteq A_u$. We call such v a *forget node*.
- Each sink v of \mathcal{P} has an assigned output $o_v \in O$ such that A_v is o_v -monochromatic according to P , i.e., $\alpha \in A_v \implies (\alpha, o_v) \in \text{P}$.

A standard/ordinary branching program is defined analogously but its nodes are labeled by cubes instead of affine spaces. Consequently, variables instead of arbitrary linear functions are queried at its query nodes.

The correspondence between a branching program computing $Search(\Phi)$ and a (linear) resolution proof refuting Φ is roughly the following. We can represent the resolution proof as a directed acyclic graph where nodes are labeled by (linear) clauses. The sources are labeled by clauses of Φ and there is exactly one sink that is labeled by an empty clause. Each node that is not a source has at most two parents and it corresponds to an application of the (linear) resolution rule (if the node has 2 parents), or the weakening rule (if the node has 1 parent). To get a (linear) branching program for $Search(\Phi)$ we just flip the direction of the edges in the resolution graph and negate the clauses that are used for node labeling. Thus, each node is labeled by a cube or an affine space, the query nodes correspond to applications of the resolution rule, and the forget nodes correspond to applications of the weakening rule. It is clear the size of a branching program \mathcal{P} (number of nodes of \mathcal{P}) is exactly the same as length of the corresponding resolution refutation.

Regular resolution is a subsystem of the resolution system, such that in any path of the resolution proof graph each variable can be resolved at most once. A read-once branching program corresponds to a regular resolution proof, i.e., on each directed path from the source to a sink each variable is queried at most once. There is interest in two generalizations of regular resolution to linear regular resolution – top-regular linear resolution [10] and bottom-regular linear resolution [7] (in both papers called as regular linear resolution). We will define both of them by their corresponding linear branching programs.

► **Definition 6** ([10]). *Let v be a node of a linear branching program \mathcal{P} . Let $Pre(v)$ be the space spanned by all linear functions queried on any path from the source of \mathcal{P} to v . Let $Post(v)$ be the space spanned by all linear functions queried on any path from v to any sink of \mathcal{P} .*

A linear branching program is *top-read-once*¹ [10] if for each query node v , we have $f_v \notin Pre(v)$. A linear branching program is *bottom-read-once* [7] if for each edge (v, u) such that v is a query node holds that $f_v \notin Post(u)$. A linear resolution proof is *top-regular*, or *bottom-regular* if the corresponding branching program is top-read-once, or bottom-read-once, respectively. We use both notion of branching program and resolution to state and prove our result, whichever is more suitable for the presentation at hand. Our separation is only for bottom-regular ResLin, i.e., for bottom-read-once linear branching program, which we abbreviate to BROLBP.

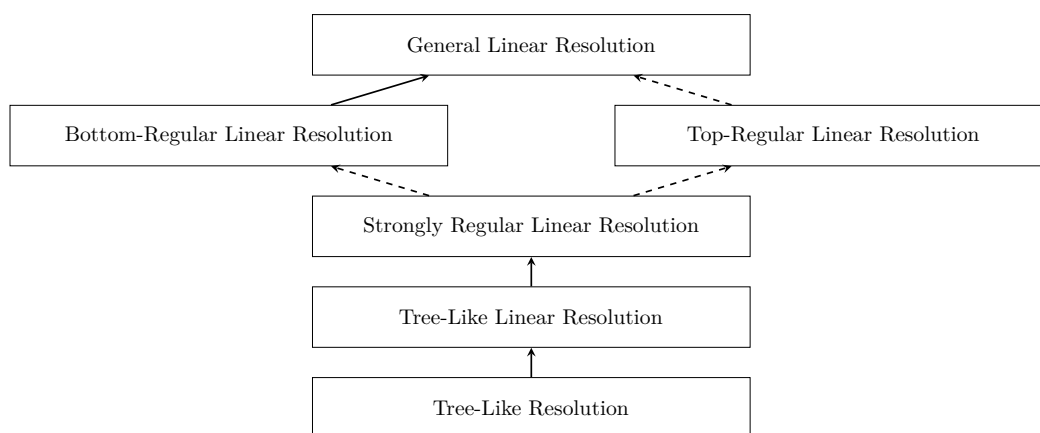
► **Lemma 7** (Lemma 2.6 [7] stated for branching programs). *Let \mathcal{P} be a BROLBP computing a search problem $P \subseteq \{0, 1\}^n \times O$. Let v be a node of \mathcal{P} such that there is a path of length t from the source of \mathcal{P} to v . Then, $\dim(Post(v)) \leq n - t$.*

► **Lemma 8** (Lemma 2.3 [7] stated for branching programs). *Let \mathcal{P} be a linear branching program computing a search problem $P \subseteq \{0, 1\}^n \times O$. Let u and v be nodes of \mathcal{P} such that there is a directed path p from u to v . Let $(M|c)$ be the system of linear equations given by queries along the path p and A be the affine space of solution of $(M|c)$, i.e., $A = \{\alpha \in \{0, 1\}^n \mid M\alpha = c\}$. Then, for A_u and A_v the affine spaces associated with u , and v , respectively, holds that $A_u \cap A \subseteq A_v$.*

¹ Gryaznov et al. [10] used just the name weakly read once for such programs.

Even more restrictive subsystems are tree-like resolution, and tree-like ResLin. These subsystems correspond to decision trees and parity decision trees. A parity decision tree (PDT) is a linear branching program such that its underlying graph is a tree, and a decision tree (DT) is a restriction where we query only bits of the input, instead of linear functions. It is clear that tree-like resolution is a subsystem of regular resolution. Analogously, tree-like ResLin is a subsystem of both bottom-regular and top-regular ResLin.

It is easy to see that strongly read-once linear branching programs are both top-read-once and bottom-read-once. Gryaznov et al. [10] observed that strongly read-once linear branching programs can simulate parity decision trees (see the appendix of Chattopadhyay and Liao [5] for the argument). A super-polynomial separation between tree-like ResLin and bottom-regular ResLin follows from the work of Itsykson and Sokolov [12]. We have the following containments:



■ **Figure 2** Relationships between various notions of linear resolution. Solid arrow indicates the separation is strict, dashed arrow indicates it's not known whether the separation is strict.

In this paper we show the existence of a CNF formula with a polynomial sized resolution refutation for which any bottom-regular linear resolution refutation requires exponential size. Thus, we show that the containment *bottom-regular linear resolution* \subseteq *general ResLin* is strict.

3.1 ROLBP Computing Boolean Function

In the context of computing Boolean functions, branching programs are defined, usually, in a more relaxed fashion in a certain sense. For instance, ordinary branching programs are defined without placing the restriction that the set of inputs reaching a node can be contained in a non-trivial sub-cube. This is something we insist when we define BPs here as our focus is on capturing the limitations of those BPs that are derived from a resolution proof DAG by reversing the direction of its edges. It, possibly, would have been more meaningful to call these latter objects affine DAGs, but we chose to call it linear branching programs for the sake of continuity wrt the earlier works by Efremenko et al. [7] and Gryaznov et al. [10]. In this section, we take the liberty of indeed calling them DAGs to compare them with BPs. Affine DAGs severely restrict the power of computing Boolean functions. This is because the set of sink nodes of such a DAG of small size computing a Boolean function f simply provides an efficient affine cover of $f^{-1}(0)$ as well as $f^{-1}(1)$. Thus, immediately, one concludes that any affine DAG, without any restriction on the number of reads of a

variable, computing the Inner-product on n bits requires $2^{\Omega(n)}$ size² as IP has large affine cover number. On the other hand, IP can be easily seen to be computed by a linear-size read-once and bit-querying branching program.

On the other hand, the situation for problems of searching a falsified clause, is quite different. As Lemma 2.4 in the work of Efremenko et al. [7] proves, for any unsatisfiable CNF Ψ , any top-read-once linear branching program solving $\text{Search}(\Psi)$ gives rise to a top-read-once affine DAG for $\text{Search}(\Psi)$ with hardly any blow-up in its size. The proof can be easily verified to additionally yield that a strongly read-once linear BP for $\text{Search}(\Psi)$, completely analogously, yields a strongly read-once affine DAG for $\text{Search}(\Psi)$ with no essential blow-up to its size. Thus, our main result, Theorem 1 holds equally for strongly read-once linear branching programs that are not restricted by definition to be affine DAGs.

4 Linear Algebraic Facts

In this section, we will describe the notions of linear algebra that we will need in our arguments. Let us introduce some notation first. Let $M \in \mathbb{F}_2^{t \times m}$ be a matrix. We denote the row space of M by $\mathcal{R}(M)$. For a vector $c \in \{0, 1\}^t$, $\mathcal{S}(M, c)$ is the affine space of solutions to the linear system $(M|c)$, i.e., $\mathcal{S}(M, c) = \{\alpha \in \{0, 1\}^m \mid M \cdot \alpha = c\}$.

The entries of vectors of \mathbb{F}_2^{mb} are naturally divided into m blocks, each having b coordinates/bits, i.e., for $j \in [m]$, the j -th block contains the coordinates $(j-1)b+1, \dots, jb$. For $j \in [m]$, $\text{BLOCK}(j) = \{(j-1)b+1, \dots, jb\}$. Also for $T \subseteq [m]$ define $\text{BLOCK}(T) = \cup_{t \in T} \text{BLOCK}(t)$. For a vector $u \in \mathbb{F}_2^{mb}$ and a block $j \in [m]$, $u^j \in \mathbb{F}_2^b$ is the vector corresponding to the block j of u , i.e., $u^j = (u_{(j-1)b+1}, \dots, u_{jb})$. We say a vector $u \in \mathbb{F}_2^{mb}$ touches a block $j \in [m]$ if the vector u^j is non-zero. A set of vectors $R \subseteq \mathbb{F}_2^{mb}$ touches a block j if at least one of the vectors in R touches j . Let U be a subspace of \mathbb{F}_2^{mb} and $T \subseteq [m]$ be a set of blocks. The subspace U_T of U is the linear space of all vectors u that do not touch any block outside T , i.e., $U_T = \{u \in U \mid \forall j \notin T : u^j = (0, \dots, 0)\}$. For $S = \text{BLOCK}(T)$, the subspace $U_{\downarrow T}$ of \mathbb{F}_2^S is the projection of U onto T , i.e., $U_{\downarrow T} = \{x \in \mathbb{F}_2^S \mid \exists y \in \mathbb{F}_2^{[mb] \setminus S} : (x, y) \in U\}$. We call a tuple of vectors $R = (u_1, \dots, u_t)$, $u_i \in \mathbb{F}_2^{mb}$ to be *safe* if the following condition holds:

- The vectors (u_1, \dots, u_t) form a matrix $M \in \mathbb{F}_2^{t \times mb}$ in echelon form, i.e., there are t distinct coordinates $a_1, \dots, a_t \in [mb]$ such that for all $i, j \in [t]$:

$$(u_i)_{a_j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

In other words, the matrix M restricted to the columns a_1, \dots, a_t is the identity matrix $I_t \in \mathbb{F}_2^{t \times t}$.

- Moreover, each pivot a_i lies in a distinct block.

The a_i 's are called the pivot variables of R . There might be multiple possible choices for the tuple of pivot variables (a_1, \dots, a_t) . In that case we pick any valid choice, say the lexicographically smallest valid choice, and call it the set of pivots.

A subspace U of \mathbb{F}_2^{mb} is *spread* if any set of k linearly independent vectors of U touches at least k blocks, for each $1 \leq k \leq m$. We say a set of blocks $T \subseteq [m]$ is an *obstruction* of a space U if $U_{\downarrow \bar{T}}$ is spread, where \bar{T} is complement of T , i.e., $\bar{T} = [m] \setminus T$. An obstruction $T \subseteq [m]$ of a space U is minimal if any proper subset $T' \subset T$ is not an obstruction of U , i.e., $U_{\downarrow \bar{T}'}$ is not spread. Efremenko et al. [7] showed the following result.

² In fact, the case for cube DAGs is known to be more dramatic. If a Boolean function f can be computed by a cube DAG of size s , then it can be also computed by a decision tree of size $s^{O(\log(s) \cdot \log n)}$.

► **Theorem 9** (Theorem 3.1, Efremenko et al. [7]). *Let U be a spread subspace of \mathbb{F}_2^{mb} with $\dim(U) \leq m$. Then, U has a safe basis.*

The following is a basic fact.

► **Observation 10.** *Let U be any subspace of \mathbb{F}_2^{mb} and $T \subseteq [m]$ be a set of blocks. Then,*

$$\dim(U) = \dim(U_T) + \dim(U_{\downarrow \bar{T}}).$$

Proof. Let U' be a suitable subspace of U such that $U = U_T \oplus U'$. It is simple to see that $U_{\downarrow \bar{T}} = (U')_{\downarrow \bar{T}}$. This is because every vector $u \in U$ can be written as $x + u'$, with $x \in U_T$ and $u' \in U'$. But, as $x_{\downarrow \bar{T}} = 0$, we have $u_{\downarrow \bar{T}} = u'_{\downarrow \bar{T}}$. Hence, we conclude that $\dim(U_{\downarrow \bar{T}}) = \dim((U')_{\downarrow \bar{T}}) \leq \dim(U')$. To establish our result, we will simply show that $\dim(U') \leq \dim((U')_{\downarrow \bar{T}})$. This follows if we show that whenever $u'_1, \dots, u'_r \in U'$ are linearly independent vectors, so are $(u'_1)_{\downarrow \bar{T}}, \dots, (u'_r)_{\downarrow \bar{T}}$. If that is not the case then there exists a vector $x \in U_T$ such that x, u'_1, \dots, u'_r are not linearly independent, contradicting our assumption. ◀

Efremenko et al. [7] showed the following properties of minimal obstructions.

► **Lemma 11.** *Let U be a subspace of \mathbb{F}_2^{mb} . Then, a minimal obstruction $T \subseteq [m]$ of U is unique and $|T| \leq \dim(U)$.*

► **Definition 12.** *For an affine space $A = \mathcal{S}(M, c) \subseteq \mathbb{F}_2^{mb}$ we define its closure $Cl(A) \subseteq [m]$ to be the unique minimal obstruction of $\mathcal{R}(M)$. Also, define $VarCl(A) \subseteq [mb]$ to be the set of variables that appear in the blocks of $Cl(A)$, i.e.,*

$$VarCl(A) = BLOCK(Cl(A)).$$

Efremenko et al. [7] proved the following relationship between the closures of two affine spaces when one contains the other.

► **Lemma 13.** *Let $A \subseteq A'$ be two affine spaces of \mathbb{F}_2^{mb} . Then, $Cl(A') \subseteq Cl(A)$.*

A *partial assignment* α' of m variables is a string in $\{0, 1, *\}^m$. A variable $X \in [m]$ is *assigned* if $\alpha_X \in \{0, 1\}$. For a total assignment $\alpha \in \{0, 1\}^m$ and $T \subseteq [m]$ we define the restriction $\alpha|_T$ of α to T to be the partial assignment arising from α by unassigning the variables that are not in T , i.e., for each $i \in [m]$

$$(\alpha|_T)_i = \begin{cases} \alpha_i & \text{if } i \in T, \\ * & \text{otherwise.} \end{cases}$$

We describe the notion of *stifling* introduced by Chattopadhyay et al. [4].

► **Definition 14.** *A Boolean function $g : \{0, 1\}^b \rightarrow \{0, 1\}$ is stifled³ if the following holds*

$$\forall i \in [b] \text{ and } a \in \{0, 1\} \exists \delta \in \{0, 1\}^b \\ \text{such that for all } \gamma \in \{0, 1\}^b \text{ with } \gamma_{|[b] \setminus \{i\}} = \delta_{|[b] \setminus \{i\}} \text{ holds that } g(\gamma) = a.$$

³ 1-stifled called by Chattopadhyay et al. [4]

23:12 Separation Regular and General Resolution over Parities

We call δ from the previous definition a *stifling assignment* for i and a . The utility of stifling is the following. An adversary can pick any variable $i \in [b]$ of g . For any $a \in \{0, 1\}$, we can pick a partial assignment $\delta_a \in \{0, 1, *\}^m$ that assigns a value to all variables except the i -th variable. Now, no matter how the adversary chooses the value for the i -th variable to get a total assignment $\gamma_a \in \{0, 1\}^b$ from δ_a , the value $g(\gamma_a)$ will be always a .

► **Definition 15.** A partial assignment $\beta \in \{0, 1, *\}^{mb}$ is called *block-respecting* if for each block $j \in [m]$, either all variables or no variables are assigned, i.e.,

$$(\beta^j)_i \in \{0, 1\} \text{ for all } i \in [b] \text{ or } (\beta^j)_i = * \text{ for all } i \in [b].$$

A block-respecting assignment $\beta \in \{0, 1, *\}^{mb}$ naturally gives a partial assignment $\vec{g}(\beta) \in \{0, 1, *\}^m$ by applying the gadget g to the assigned blocks. Formally, for each $j \in [m]$ we have

$$\vec{g}(\beta)_j = \begin{cases} g(\beta_1^j, \dots, \beta_b^j) & \text{if for all } i \in [b] : \beta_i^j \text{ are assigned,} \\ * & \text{otherwise.} \end{cases}$$

► **Definition 16.** Let $A \subseteq \mathbb{F}^{mb}$ be an affine space and $\beta \in A$. The *closure-assignment* of β , $\beta|_{\text{VarCl}(A)}$ is the partial assignment which fixes all coordinates in blocks of $\text{Cl}(A)$ according to β and keeps other coordinates free. In other words,

$$(\beta|_{\text{VarCl}(A)})^j = \begin{cases} \beta^j & \text{if } j \in \text{Cl}(A), \\ (*, \dots, *) & \text{otherwise.} \end{cases}$$

► **Lemma 17.** Let $A = \mathcal{S}(M, c) \subseteq \mathbb{F}^{mb}$ be an affine space and let $g : \{0, 1\}^b \rightarrow \{0, 1\}$ be a stifled gadget. Let $\beta \in A$ be a vector and $\beta' \in \{0, 1, *\}^{mb}$ be its closure assignment. Let $\alpha' := \vec{g}(\beta') \in \{0, 1, *\}^m$. Then, for any extension of α' to a total assignment $\alpha \in \{0, 1\}^m$, there exists $\gamma \in A$ such that $\vec{g}(\gamma) = \alpha$.

Proof. WLOG assume the rows of M are linearly independent. Let $U = \mathcal{R}(M)$ be the row-space of M and let $T \subseteq [m]$ be the closure of A . First, we construct a matrix M' which has the same row-space as M .

Construction of M'

1. Let (u_1, \dots, u_d) be an arbitrary basis of U_T and let $M_1 \in \mathbb{F}^{d \times mb}$ be the matrix whose rows are the vectors u_1, \dots, u_d :

$$M_1 = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_d \end{bmatrix}$$

2. Let $(w_1, \dots, w_{d'})$ be a safe basis of $U_{\downarrow \bar{T}}$. Such a basis exists by the definition of closure and Theorem 9. Let $a_1, \dots, a_{d'}$ be pivots of $w_1, \dots, w_{d'}$. Each of these pivots lie in a distinct block. Moreover, none of these pivots are in the blocks of T .

Let $L : U \rightarrow U_{\downarrow \bar{T}}$ be the projection of U to $U_{\downarrow \bar{T}}$. Let w'_i be an arbitrary pre-image of w_i according to L , i.e., $L(w'_i) = w_i$. Since $(w_1, \dots, w_{d'})$ are linearly independent, the vectors $(w'_1, \dots, w'_{d'})$ are linearly independent as well. Let $M_2 \in \mathbb{F}^{d' \times mb}$ be the matrix with the vectors $w'_1, \dots, w'_{d'}$ as its rows.

$$M_2 = \begin{bmatrix} w'_1 \\ w'_2 \\ \vdots \\ w'_{d'} \end{bmatrix}$$

3. Take $M' \in \mathbb{F}^{(d+d') \times mb}$ to be the matrix obtained by stacking M_1 on top of M_2 :

$$M' = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix}$$

▷ **Claim 18.** The matrices M and M' have the same row-space.

Proof. By Observation 10, $\dim(U) = \dim(U_T) + \dim(U_{\downarrow T}) = d + d'$. No row of M_2 can be generated by rows of M_1 as the pivots $a_1, \dots, a_{d'}$ of the matrix M_2 lie in columns where the matrix M_1 has only 0 entries. Thus, $\text{rank}(M') = \text{rank}(M_1) + \text{rank}(M_2) = d + d' = \dim(U) = \text{rank}(M)$. Moreover, any row of M' lies in $U = \mathcal{R}(M)$. It follows that $\mathcal{R}(M) = \mathcal{R}(M')$. ◁

Thus, there is a vector $c' \in \mathbb{F}^{mb}$ such that $A = \mathcal{S}(M', c')$. Now, we prove the lemma. We are given a vector $\beta \in \mathcal{S}(M', c')$ and a target assignment $\alpha \in \{0, 1\}^m$ such that $g(\beta^j) = \alpha_j$ for all $j \in \text{Cl}(A)$. Our goal is to show the existence of a $\gamma \in \mathbb{F}^{mb}$ such that $\vec{g}(\gamma) = \alpha$ and $M'\gamma = c'$. Recall that the tuple of rows of M_2 , $(w'_1, \dots, w'_{d'})$ is a safe tuple with set of pivots $a_1, \dots, a_{d'}$. Suppose $a_j \in \text{BLOCK}(b_j)$. The blocks $b_1, \dots, b_{d'}$ are distinct and all of them lie in $[m] \setminus T$. Let $\text{PIVOTS} = \{b_1, b_2, \dots, b_{d'}\}$ and $\text{FREE} = [m] \setminus (T \cup \text{PIVOTS})$. We construct γ in two steps. In the first step we construct a $\tilde{\beta} \in \mathbb{F}^{mb}$ such that $\vec{g}(\tilde{\beta}) = \alpha$, but it is not necessarily the case that $M'\tilde{\beta} = c'$. In the second step we modify $\tilde{\beta}$ in the coordinates $a_1, \dots, a_{d'}$ to get an assignment $\gamma \in \mathcal{S}(M', c')$.

Constructing $\tilde{\beta}$

- For each $i \in T = \text{Cl}(A)$, $\tilde{\beta}$ agrees with β on $\text{BLOCK}(i)$, i.e., $(\tilde{\beta})^i = \beta^i$.
- For each $i \in \text{FREE}$, choose an arbitrary preimage $u_i \in g^{-1}(\alpha_i)$ and set $(\tilde{\beta})^i = u_i$.
- For each $i = b_j \in \text{PIVOT}$: Suppose the pivot a_j is the ℓ -th coordinate of $\text{BLOCK}(j)$. Pick $u_i \in g^{-1}(\alpha_i)$ to be a stifling assignment for the ℓ -th coordinate, i.e., $g(u_i) = g(u_i^{(\ell)}) = \alpha_i$ (where $s^{(\ell)}$ denotes s with ℓ -th coordinate flipped). Set $(\tilde{\beta})^i = u_i$.

Constructing γ : We modify $\tilde{\beta}$ in the coordinates $a_1, \dots, a_{d'}$ to get an assignment γ in $\mathcal{S}(M', c')$ as follows. For $1 \leq j \leq d$, let $f_j = \langle w'_j, \tilde{\beta} \rangle + (c')_j$. Let $\gamma \in \mathbb{F}^{mb}$ be the following assignment:

$$\gamma_i = \begin{cases} (\tilde{\beta})_i & \text{if } i \notin \{a_1, \dots, a_{d'}\}, \\ (\tilde{\beta})_i + f_j & \text{if } i = a_j. \end{cases}$$

▷ **Claim 19.** $\vec{g}(\gamma) = \alpha$ and $\gamma \in \mathcal{S}(M', c')$.

Proof. We show both points separately.

Showing $\vec{g}(\gamma) = \alpha$: We argue that $g(\gamma^i) = \alpha_i$ for all $i \in [m]$.

Case 1, $i \in T$: We have set $(\tilde{\beta})^i = \beta^i$. Note that γ differs from $\tilde{\beta}$ only in coordinates $a_1, a_2, \dots, a_{d'}$. All these coordinates lie outside $\text{BLOCK}(T)$. Thus, $g(\gamma^i) = g(\beta^i) = \alpha_i$.

Case 2, $i \in \text{FREE}$: We have set $(\tilde{\beta})^i = u_i$ where $u_i \in g^{-1}(\alpha_i)$. Again, note that γ differs from $\tilde{\beta}$ only in the coordinates $a_1, a_2, \dots, a_{d'}$, all of which lie outside $\text{BLOCK}(i)$. It follows that $g(\gamma^i) = \alpha_i$.

23:14 Separation Regular and General Resolution over Parities

Case 3, $i \in \text{PIVOTS}$: Let $i = b_j$ and let $a_j \in \text{BLOCK}(b_j)$ be the corresponding pivot variable. Recall that each pivot variable lies in a distinct block. Let a_j be the ℓ -th coordinate of $\text{BLOCK}(b_j)$. We have set $(\tilde{\beta}^i) = u_i$ where $u_i \in g^{-1}(\alpha_i)$ is a stifling assignment for ℓ and α_i . This means that $g((u_i)^{(\ell)}) = g(u_i) = \alpha_i$ ($s^{(\ell)}$ denotes s with ℓ -th coordinate flipped). Notice that γ and $\tilde{\beta}$ agree everywhere on $\text{BLOCK}(b_j)$ except possibly a_j . This implies $g(\gamma^i) = \alpha_i$.

Showing $\gamma \in \mathcal{S}(M', c')$: Note that all equations corresponding to rows in M_1 are satisfied by γ since they are satisfied by β and hence by $\tilde{\beta}$ too. That the equations corresponding to M_2 are satisfied by γ follows from the row echelon structure of M_2 , i.e., the fact that after an appropriate permutation of the columns, M' looks as follows:

$$M' = \left(\begin{array}{c|c|c} B_1 & 0 & 0 \\ B_2 & I_{d'} & B_3 \end{array} \right) \begin{array}{l} = M_1 \\ = M_2 \end{array} \quad \triangleleft$$

Closure T | Pivots of M_2

Since $\mathcal{S}(M', c') = \mathcal{S}(M, c) = A$, Lemma 17 follows immediately from Claim 19. ◀

5 Proof Outline

In this section, we provide an outline of the proof of our main result, Theorem 1. The proof consists of two parts. The first part shows that the formula $\text{Stone}(G, \rho) \circ g$ has a polynomial length resolution proof for any directed acyclic graph G on N vertices and out-degree 2, any obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$, and any gadget $g : \{0, 1\}^b \rightarrow \{0, 1\}$, where b is logarithmic in N (recall that the number of variables m of the formula $\text{Stone}(G, \rho)$ is $2N^2$). This part of the proof is an adaptation of an analogous proof for the stone formula given by Alekhovich et al. [1].

The second part establishes that there is a graph G and an obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$ such that any bottom-regular ResLin proof of $\text{Stone}(G, \rho) \circ \text{IP}$ has exponential length in m , where IP is the inner product function on $b = \Theta(\log m)$ bits. The proof of this part is involved and non-trivial. We outline the main steps in the figure below, immediately followed by a high-level description of each step depicted.

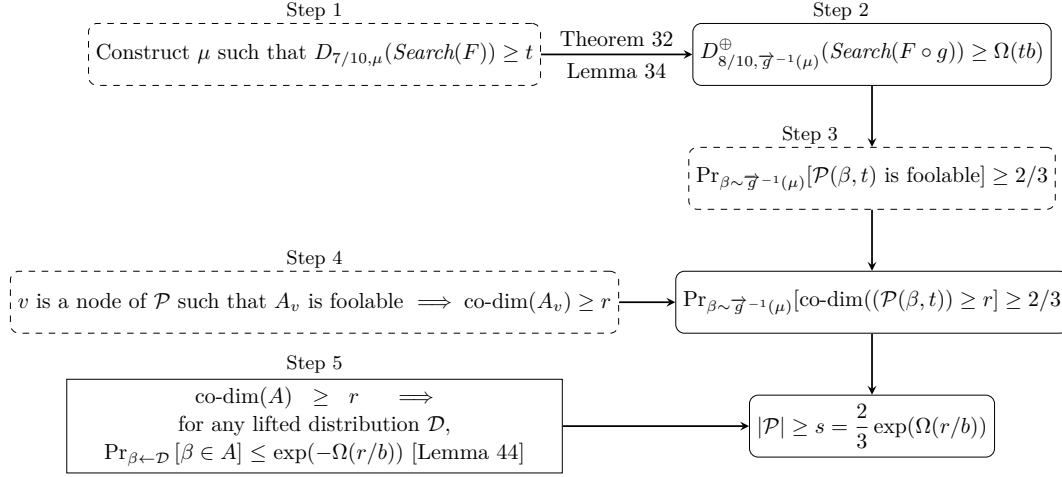
Outline of the Lower Bound Proof

Our argument is an adaptation of the method presented in Efremenko et al [7] with addition of some new ingredients. See Figure 3, for depicting the method. Let $\mathcal{P}(\beta, t)$ be the node that \mathcal{P} arrives at after making t linear queries on β .

Some Details

Let \mathcal{P} be a bottom-read-once branching program computing $\text{Search}(\text{Stone}(G, \rho) \circ \text{IP})$ corresponding to a bottom-regular ResLin proof of $\text{Stone}(G, \rho) \circ \text{IP}$ where G is a pyramid graph of n levels and ρ is a carefully chosen obfuscation map. The proof consists of several steps.

1. We design a distribution μ over the assignment of variables of the base formula F over m variables, typically supported over *critical assignments*, i.e. those which result in the falsification of exactly one clause. This module requires one to show that $\text{Search}(F)$ is average-case hard for deterministic decision trees of small height wrt μ . In particular, our Lemma 31 proves that the problem $\text{Search}(\text{Stone}(G, \rho))$ is average-case hard for deterministic decision trees of height at most $O(n^{1/3})$. As the μ exhibited is formula specific, the box corresponding to this module is dashed.



■ **Figure 3** Outline of the proof. The solid boxes refer to parts that are quite general and not specific to a formula, while the dashed boxes contain modules that are more specific to $\text{SP}_n \circ \text{IP}$ and similar formulas.

2. In this step, we prove that the search problem associated with the lifted formula $F \circ g$ remains average case hard for *parity decision trees* wrt a *lifted distribution* as long as the gadget g has small rectangular discrepancy. More precisely, let $\vec{g}^{-1}(\mu)$ denote the lifted distribution generated by the following sampling: sample an input $z \in \{0, 1\}^m$ according to μ . Then, sample at random an input $\beta \in \{0, 1\}^{mb}$, conditioned on $\vec{g}(\beta) = z$. Using Theorem 32, implicit in the proof of the main result of Chattopadhyay, Filmus, Koroth, Meir and Pitassi [3], we conclude that $\text{Search}(F \circ g)$ is average-case hard for deterministic parity decision trees of small height, under the lifted distribution $\vec{g}^{-1}(\mu)$. This step is generic and works for any gadget of size $c \cdot \log(m)$, that has sufficiently small rectangular discrepancy under the uniform distribution over $\{0, 1\}^b$. The gadget we use here is IP.
3. We then want to define a notion of progress the branching program \mathcal{P} has made on arriving at a node v . To do so, consider the affine space A_v that labels the node. A_v may have nearly fixed/exposed the values of some of the blocks of input. These dangerous blocks are precisely $\text{Cl}(A_v)$ as defined in Section 4. They form the minimum obstruction set. Intuitively, the danger is \mathcal{P} may have nearly found out a falsified clause of $F \circ g$ on reaching v if that clause was made up entirely of variables from blocks in $\text{Cl}(A_v)$. However, in this step we observe that the average-case hardness of the Search problem for PDTs proved in the previous step precludes this from happening with appreciable probability, when the input is sampled according to the lifted distribution $\vec{g}^{-1}(\mu)$. To formalize this idea, we need to concretely say when A_v is (not) dangerous. So far, we have not been able to lay out a general notion of danger, but notions specific to individual formulas have been defined. For $\text{Stone}(G, \rho)$, this notion is captured by Definition 36 of *foolable* spaces, provided in Section 7.3. Theorem 37 shows that w.h.p., \mathcal{P} reaches a foolable space on walking for $n^{1/3}$ steps, querying an input sampled according to $\vec{g}^{-1}(\mu)$.
4. In this step, we show that when the affine space A_v is not dangerous, i.e. it is foolable or consistent, the appropriate notion depending on the formula at hand, A_v has large co-dimension. All steps until now held for general branching programs (or equivalently proof DAGs). This step is the only one where the bottom-read-once property is exploited. For $\text{Stone}(G, \rho)$, this is achieved in Section 7.4, at the end, by Lemma 40.

23:16 Separation Regular and General Resolution over Parities

5. In this step, we prove a general result about lifted distributions. For any affine space A of $\text{co-dim}(A) = r$ and any distribution μ on $\{0, 1\}^m$, we prove that β sampled by $\vec{g}^{-1}(\mu)$ is in A only with probability $2^{-\Omega(r/b)}$, as long as the gadget g is balanced and stifling. In other words, lifted distributions, even though their support is quite sparse in the ambient space, are pseudo-random for the rank measure. This property, though simple to prove, turns out to be extremely useful, especially for formulas like the stone formulas that are barely hard.

At this stage we are ready to put together the above steps in the following way. Let R be a set of nodes w of \mathcal{P} such that there is a path from the root of \mathcal{P} to w of length t , and $\text{co-dim}(A_w) \geq t$. Setting $t := n^{1/3}$ we have the following.

$$\begin{aligned} \frac{7}{10} &\leq \Pr_{\beta \sim \vec{\mathbb{P}}^{-1}(\mu)} [\text{co-dim}(A_v) \geq t \text{ for } v = \mathcal{P}(\beta, t)] && \text{(by Step 3 and 4)} \\ &\leq \sum_{w \in R} \Pr_{\beta \sim \vec{\mathbb{P}}^{-1}(\mu)} [\mathcal{P}(\beta, t) = w] && \text{(by union bound)} \\ &\leq |R| \cdot 2^{\Omega(-t/b)} && \text{(by Step 5)} \end{aligned}$$

By rearranging, we get the lower bound $|R| \geq 2^{\Omega(n^{1/3}/\log n)}$. Recall that the number of variables of $\text{Stone}(G, \rho) \circ \mathbb{P}$ is $M = \Theta(n^4 \log(n))$. In terms of M , the lower bound is $2^{\Omega(M^{1/12}/\log^{13/12} M)} = 2^{M^{\Omega(1)}}$.

6 Upper Bound

In this section, we show the upper bound part of Theorem 1.

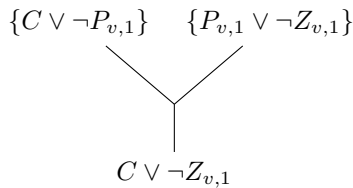
► **Theorem 20.** *Let $G = (V, E)$ be an directed acyclic graph with N vertices such that there is exactly one root r (vertex with indegree 0), and each non-sink vertex has outdegree exactly 2. Let $\rho : [N]^3 \rightarrow \mathcal{V}$ be any obfuscation map, and $g : \{0, 1\}^b \rightarrow \{0, 1\}$ be a Boolean function for $b \leq O(\log N)$. Then, the formula $\text{Stone}(G, \rho) \circ g$ admits a resolution refutation of length polynomial in N .*

The proof of Theorem 20 is an adaptation of the proof given by Alekhnevich et al. [1] for lifted formulas. We remark that Alekhnevich et al. [1] presented a resolution refutation for the stone formulas of constant width. This allow us to adapt the refutation for the lifted formula. For the rest of the section, we fix a graph G , an obfuscation map ρ , and a gadget g satisfying the assumptions of Theorem 20. First, we prove several auxiliary lemmas about the formula $\text{Stone}(G, \rho) \circ g$.

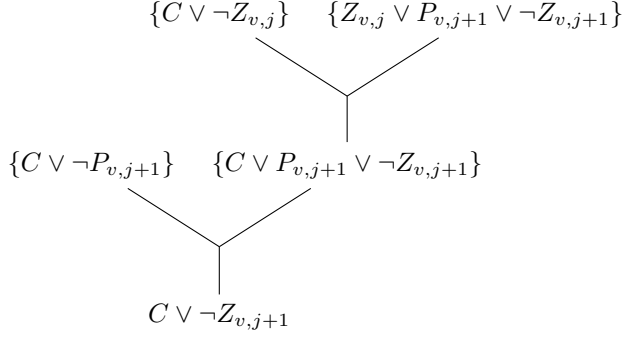
► **Lemma 21.** *Let C be a clause with width w . Suppose we have derived the clauses $C \vee \neg P_{v,j}$ for a fixed $v \in V$ and all $1 \leq j \leq N$. Then, we can derive C in N steps in width $\leq w + 2$.*

Proof. We derive the clause C in N steps. We will subsequently derive $C \vee \neg Z_{v,j+1}$ from $C \vee \neg Z_{v,j}$

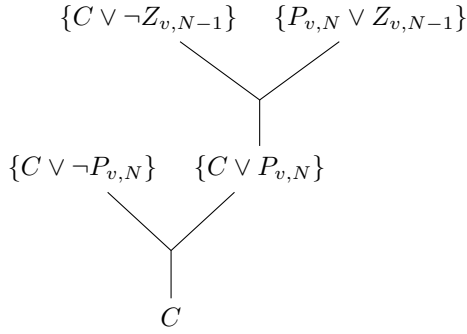
Base step: Deriving $C \vee \neg Z_{v,1}$.



Step j : For $j \in [N - 2]$, deriving $C \vee \neg Z_{v,j+1}$ from $C \vee \neg Z_{v,j}$.



Final step: Deriving C .

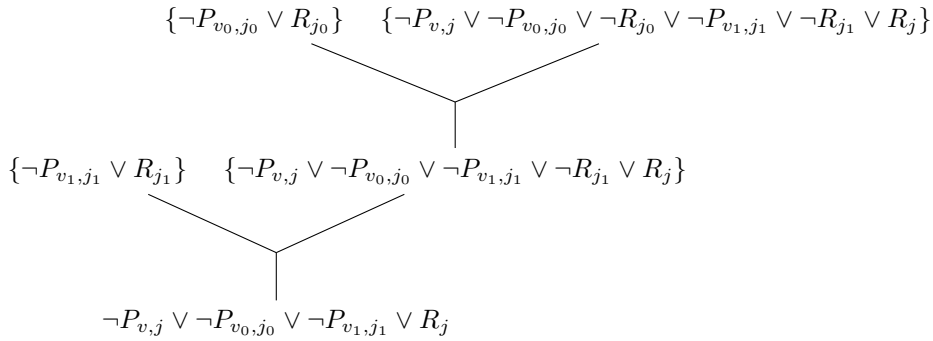


For a vertex v , we define the set of clauses $S(v) = \{\neg P_{v,j} \vee R_j \mid 1 \leq j \leq N\}$.

► **Lemma 22.** *Let v be a vertex in G with children v_0, v_1 . We can derive $S(v)$ from $S(v_0)$, and $S(v_1)$ in constant width and length $O(N^3)$.*

Proof. We derive $S(v)$ in several steps.

1. For every $j, j_0, j_1 \in [N]$, we perform the following sequence of operations:



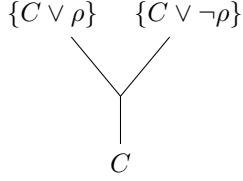
2. For each fixed j_0, j , we apply Lemma 21 to the clause $C := \neg P_{v,j} \vee \neg P_{v_0,j_0} \vee R_j$ and we derive $\neg P_{v,j} \vee \neg P_{v_0,j_0} \vee R_j$.
3. For each fixed j , we apply Lemma 21 to the clause $C := \neg P_{v,j} \vee R_j$ and we derive $\neg P_{v,j} \vee R_j$. ◀

► **Lemma 23.** *The formula $\text{Stone}(G, \rho)$ has a resolution refutation of width $O(1)$ and size polynomial in N .*

23:18 Separation Regular and General Resolution over Parities

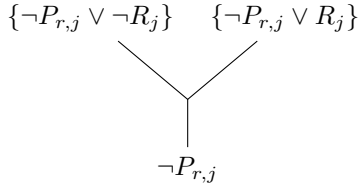
Proof. The refutation proceeds in the following steps.

Elimination of the ρ 's: For every induction clause C , we resolve the appended ρ -variable.



Derivation of $S(r)$: For each sink s of G , the clauses $S(s)$ are present in the axioms of $\text{Stone}(G, \rho)$. By Lemma 22, we subsequently derive the set $S(r)$ for the root r of G .

Empty clause derivation: For each $1 \leq j \leq N$, we derive $\neg P_{r,j}$.



Now by applying Lemma 21 for C being an empty clause \perp , we derive \perp , that concludes the proof. \blacktriangleleft

Now, from constant-width polynomial-length refutation of $\text{Stone}(G, \rho)$ we derive a polynomial-length refutation of the lifted formula $\text{Stone}(G, \rho) \circ g$.

► **Lemma 24.** *Let $g : \{0, 1\}^b \rightarrow \{0, 1\}$ be a Boolean function and Φ be a CNF unsatisfiable formula over n variables containing only constant width clauses. Suppose Φ has a resolution refutation of length ℓ and constant width. Then, $\Phi \circ g$ contains clauses of width $O(b)$ and admits a resolution refutation of size $\ell \cdot 2^{O(b)}$.*

Proof. By construction, if C is a clause of width k , then $|C \circ g| \leq 2^{bk}$. If k is constant, this is $2^{O(b)}$. We show that, for every derivation step $(A \vee x), (B \vee \neg x) \rightarrow (A \vee B)$ in a proof for Φ , we can derive all clauses of $(A \vee B) \circ g$ from the clauses of $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ in polynomial size, assuming each of A, B has constant width. This follows from the fact that $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ semantically imply $(A \vee B) \circ g$: an assignment $(x_{i,1}, \dots, x_{i,b})_{i \in [M]}$ satisfies formula $C \circ g$ if and only if the assignment $(g(x_{i,1}, \dots, x_{i,b}))_{i \in [n]}$ satisfies clause C . And since clauses $A \vee x, B \vee \neg x$ semantically imply $A \vee B$, it follows that the formulas $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ semantically imply the formula $(A \vee B) \circ g$.

As both A and B are constant-width clauses, each of the formulae $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ are defined on at most $O(b)$ variables. By Lemma 5, we can derive each clause in $(A \vee B) \circ g$ from $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ in at most $2^{O(b)}$ resolution steps.

Using this fact, we can mimic the resolution refutation of Φ . For each intermediate clause C derived in the resolution refutation for Φ , we can derive all clauses in $C \circ g$. In the end, we derive $\perp \circ g = \{\perp\}$, i.e., the empty clause. Assuming the width of the resolution refutation for Φ is bounded by some constant, the total length of our simulation is at most $\ell \cdot 2^{O(b)}$. \blacktriangleleft

Now, Theorem 20 is a corollary of Lemma 23, and 24.

7 Lower Bound

In this subsection, we prove the lower bound part of Theorem 1 following the outline given in Section 5.

► **Theorem 25.** *There is an obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$ such that any bottom-regular ResLin refutation of $\text{SP}_{n,\rho} \circ \text{IP}$ must have length at least $2^{\Omega(n^{1/3}/\log n)}$.*

Recall that number of variables of $\text{SP}_{n,\rho} \circ \text{IP}$ is $\Theta(n^4 \log n)$. Thus, the lower bound given by Theorem 25 yields the lower bound claimed in Theorem 1. For the rest of this section, we fix $G = (V, E)$ to be the pyramid graph of n levels, and $N = n(n+1)/2$ vertices.

7.1 The Stone Formula is Average-Case Hard for Decision Trees

We shall construct a distribution μ on $\{0, 1\}^m$ such that for any obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$, the search problem $\text{Search}(\text{SP}_{n,\rho})$ is hard on average w.r.t. μ for deterministic decision trees of sufficiently small height (around $n^{1/3}$).

First, we fix an arbitrary bijection $f : [N] \rightarrow V$ between stones and vertices of the pyramid. All assignments in $\text{Supp}(\mu)$ will place the stone i on vertex $f(i)$. The distribution μ samples the assignments as follows.

1. Assign stone i to vertex $f(i)$. Formally for each $v \in V$, $i \in [N]$, and $j \in [N-1]$, we set:

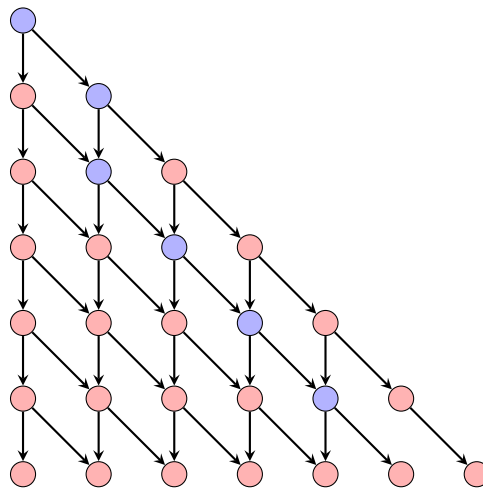
$$P_{v,i} = \begin{cases} 1 & \text{if } f(i) = v \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{v,j} = \begin{cases} 0 & \text{if } j < f^{-1}(v) \\ 1 & \text{otherwise} \end{cases}$$

2. Sample $n-2$ independent uniform bits $B_2, \dots, B_{n-1} \in \{0, 1\}$
3. Let $X_1 = 1$, and for $2 \leq j \leq n-1$, let $X_j = X_{j-1} + B_j$. Color the vertices (j, X_j) blue for $1 \leq j \leq n-1$ and other vertices red, i.e., for each stone $i \in [N]$, we set:

$$R_i = \begin{cases} 0 & \text{if } j \leq n-1 \text{ and } (j, X_j) = f(i) \\ 1 & \text{otherwise} \end{cases}$$

Let $\alpha \in \text{Supp}(\mu)$. The assignment α corresponds to the following stone placement. It places a different stone on each vertex. There is a path P from the root $r = (1, 1)$ to a vertex v in the level $n-1$ given by the random variables X_1, \dots, X_{n-1} , i.e. the vertices of the path are $\{(1, X_1), \dots, (n-1, X_{n-1})\}$. The stones on the vertices of P are colored blue, all other stones are colored red. An example of such a coloring is shown in Figure 4.



■ **Figure 4** An example of a pyramid graph with coloring giving by an assignment sampled by the hard distribution μ .

23:20 Separation Regular and General Resolution over Parities

We call the path P as the *blue path induced by α* and the vertex v as the *end of P* . Note that the only clause falsified by the assignment α is one of the induction clauses for the vertex v and its children u and w . Formally, for the stones $i = f^{-1}(u)$, $j = f^{-1}(w)$, and $k = f^{-1}(v)$, the assignment α falsifies exactly one of the following two induction clauses (depending how α sets the value of $\rho(i, j, k)$):

$$D_1(v) := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i, j, k)$$

$$D_0(v) := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg \rho(i, j, k)$$

Consider a random walk Y_1, \dots, Y_k on a number line starting at $q \in \mathbb{N}$ distributed as follows: $Y_1 = q$, and for $i > 1$

$$Y_i = \begin{cases} Y_{i-1} + 1 & \text{with probability } \frac{1}{2} \\ Y_{i-1} & \text{with probability } \frac{1}{2} \end{cases}$$

Note that the random variables X_1, \dots, X_{n-1} used in the construction of μ are distributed as Y_1, \dots, Y_{n-1} for $Y_1 = 1$.

► **Lemma 26.** *There exists a constant $c_1 \geq 0$ such that for any $p \in \{q, \dots, q + k - 1\}$ and $t \in \{2, \dots, k\}$, we have $\Pr[Y_t = p] \leq c_1/\sqrt{t}$.*

Proof. Note that $Y_t = q + \sum_{i=2}^t B_i$, where each B_i is an independent uniform random bit. Now, $Y_t = p = p' + q$ for $p' \in \{0, \dots, k - 1\}$ if and only if $\sum_{i=2}^t B_i = p'$.

$$\Pr[Y_t = p] = \Pr\left[\sum_{i=2}^t B_i = p'\right] = \binom{t-1}{p'} \cdot 2^{-t+1} \leq \binom{t-1}{\lfloor \frac{t-1}{2} \rfloor} \cdot 2^{-t+1} \leq \frac{c_1}{\sqrt{t}}$$

For an appropriate constant $c_1 > 0$, the last inequality is implied by Stirling's formula. ◀

For a set $S \subseteq [k] \times \mathbb{N}$, we say the random walk W avoids S if for all $(i, j) \in S$ it holds that $Y_i \neq j$.

► **Lemma 27.** *Let $c_2 \geq 1$ be a constant and $S \subseteq [k] \times \mathbb{N}$ be a set of forbidden points with $|S| \leq t$. Suppose there exists an interval $I = [L, R] \subseteq [k]$ with $|I| \geq c \cdot t^2$ for sufficiently large constant c depending on c_2 , such that no point of S has the first coordinate in I , i.e., for all $(i, j) \in S$: $i < L$ or $i > R$. If the random walk $W = Y_1, \dots, Y_k$ avoids S with non-zero probability, then for any $z \in \{q, \dots, q + k - 1\}$ it holds that*

$$\Pr[Y_k = z \mid W \text{ avoids } S] \leq \frac{1}{c_2 t}.$$

Proof. We partition S into two subsets S_1 and S_2 of points before and after the interval I , $S_1 = \{(i, j) \in S \mid i < L\}$, and $S_2 = \{(i, j) \in S \mid i > R\}$. Note that by the assumption, we have $S = S_1 \dot{\cup} S_2$.

We show that for all p such that $\Pr[Y_L = p \mid W \text{ avoids } S] > 0$, we have $\Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] \leq 1/c_2 t$. We set $c := 4c_1^2 c_2^2$ where c_1 is the constant given by Lemma 26. We have by Lemma 26 that

$$\Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S_1] = \Pr[Y_k = z \mid Y_L = p] \leq \frac{1}{2c_2 t}, \quad (1)$$

since $z - L \geq |I| \geq c \cdot t^2$. Again, for all $(i, j) \in S_2$ (i.e., $i > R$) we have by Lemma 26 that

$$\Pr[Y_i = j \mid Y_L = p, W \text{ avoids } S_1] \leq \frac{1}{2c_2 t} \leq \frac{1}{2t}.$$

By union bound,

$$\Pr[\exists(i, j) \in S_2 \text{ such that } Y_i = j \mid Y_L = p, W \text{ avoids } S_1] \leq \frac{1}{2}. \quad (2)$$

Therefore,

$$\begin{aligned} \Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] &= \frac{\Pr[Y_k = z, W \text{ avoids } S_2 \mid Y_L = p, W \text{ avoids } S_1]}{\Pr[W \text{ avoids } S_2 \mid Y_L = p, W \text{ avoids } S_1]} \\ &\leq 2 \cdot \Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S_1] \quad (\text{by (2)}) \\ &\leq \frac{1}{c_2 t} \quad (\text{by (1)}) \end{aligned}$$

Now, we are ready to finish the proof.

$$\begin{aligned} \Pr[Y_k = z \mid W \text{ avoids } S] &= \sum_p \Pr[Y_L = p \mid W \text{ avoids } S] \cdot \Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] \\ &\leq \sum_p \Pr[Y_L = p \mid W \text{ avoids } S] \cdot \frac{1}{c_2 t} = \frac{1}{c_2 t} \quad \blacktriangleleft \end{aligned}$$

Now we show that when inputs are sampled according to μ , any deterministic decision tree for $\text{Search}(\text{SP}_{n,\rho})$ with small height makes an error with high probability. Note that for each $v \in V, i \in [N], j \in [N-1]$, the assignment to the variables $P_{v,i}$ and $Z_{v,j}$ are fixed by any assignment in $\text{Supp}(\mu)$ (in other words, for each vertex the stone placed on it is fixed). Thus, we can assume WLOG the decision tree only queries the variables R_j . We say a decision tree queries the color of a vertex v of G if it queries the variable $R_{f^{-1}(v)}$. (Recall that the stone $f^{-1}(v)$ is placed on the vertex v by assignments in $\text{Supp}(\mu)$.)

Note that there is a simple, even non-adaptive, decision tree of height $O(\sqrt{n})$ that makes few errors. It simply queries the colours of $O(\sqrt{n})$ nodes of the pyramid graph, centered around the $(n-1)/2$ -th node at level $n-1$. With very high probability, there is a blue node among the queried ones which uniquely identifies the falsified induction clause. Nevertheless, we will show that all decision trees of height at most $n^{1/3}$, will make errors with large probability to identify a falsified clause.

Consider a decision tree \mathcal{T} for $\text{Search}(\text{SP}_{n,\rho})$. We transform \mathcal{T} into a decision tree \mathcal{T}' in a canonical form:

- Initially, \mathcal{T}' always queries the color of the root r of G .
- Suppose \mathcal{T} outputs an induction clause $D_0(v)$ or $D_1(v)$ for a vertex v of G . Then, \mathcal{T}' queries the color of the vertex v first. If the color of v is red (i.e., $R_{f^{-1}(v)} = 1$), then \mathcal{T}' outputs an error symbol. Otherwise it outputs the same induction clause that \mathcal{T} outputs.
- If \mathcal{T} outputs any other clause, then \mathcal{T}' outputs an error symbol.

We remark this modification increases the height of the tree by at most two. Given any assignment in $\text{Supp}(\mu)$, a decision tree \mathcal{T}' in a canonical form can output either an induction clause from $\{D_0(v), D_1(v) \mid v \in V(G)\}$ or an error symbol. The probability of \mathcal{T} making an error is precisely the probability of reaching a leaf node of \mathcal{T}' labeled with an error symbol.

Note that for each cube $C \subseteq \{0, 1\}^m$ there is a corresponding partial assignment $\alpha_C \in \{0, 1, *\}^m$ such that the cube C is exactly the set of total extensions of α_C , i.e., $C = \{\alpha \in \{0, 1\}^m \mid \alpha \text{ extends } \alpha_C\}$. We say a cube $C \subseteq \{0, 1\}^m$ fixes a vertex $v \in V(G)$ to red (or blue) if the corresponding partial assignment α_C assigns a value 1 (or 0) to the variable $R_{f^{-1}(v)}$.

Now, fix a decision tree \mathcal{T} for $\text{Search}(\text{SP}_{n,\rho})$ in a canonical form and let $h := \gamma \cdot n^{1/3}$ be the height of \mathcal{T} , where $\gamma > 0$ is sufficiently small constant.

23:22 Separation Regular and General Resolution over Parities

► **Definition 28.** We say a cube $C \subseteq \{0,1\}^m$ is useful if there exist $1 \leq L_1 \leq L_2 \leq L_3 \leq L_4 \leq N$ such that:

1. The cube C fixes the some vertex in level L_1 to blue and some vertex in level L_4 to blue.
2. For all $L_2 \leq \ell \leq L_3$, the cube C does not fix the color of any vertex in level ℓ .
3. $L_3 - L_2 \geq \frac{n}{2h}$

A node p of \mathcal{T} is called useful if the cube C_p associated with it is useful.

Clearly, the root of \mathcal{T} is not useful.

► **Lemma 29.** Let $\alpha \in \text{Supp}(\mu)$ be an assignment on which \mathcal{T} reaches the leaf p . If p does not output an error symbol, then p is useful.

Proof. Let v be the endpoint of the blue path induced by α . Then, p outputs one of the induction clauses $D_0(v)$ or $D_1(v)$ for $v \in V(G)$. Since \mathcal{T} is in the canonical form, the cube C_p fixes the vertex v and the root r of G to blue. Recall that the vertex v is in level $n - 1$. Let $1 = \ell_1 < \dots < \ell_d = n - 1$ be the levels where C_p fixes some vertices. Since $d \leq h$, there must exist an i such that $\ell_{i+1} - \ell_i \geq \frac{n-1}{h+1} > \frac{n}{2h}$. There is no fixed vertex on levels $\ell_i + 1, \dots, \ell_{i+1} - 1$.

We take largest the ℓ_1 such that $\ell_1 \leq \ell_i$ and C_p fixes a vertex on ℓ_1 to blue. Similarly, we take the smallest ℓ_2 such that $\ell_2 \geq \ell_{i+1}$ and C_p fixes a vertex on ℓ_2 to blue. The cube C_p satisfies the conditions of being a useful node in Definition 28 by taking $(L_1, L_2, L_3, L_4) = (\ell_1, \ell_i + 1, \ell_{i+1} - 1, \ell_2)$. ◀

► **Lemma 30.** For any $\varepsilon > 0$ there exists $\gamma > 0$ such that

$$\Pr_{\alpha \sim \mu} [\text{The computation path of } \mathcal{T} \text{ on } \alpha \text{ reaches a useful node}] \leq \varepsilon.$$

Proof. Let $\mathcal{T}(\alpha, k)$ denote the node of \mathcal{T} reached by α after k queries. For each $1 \leq k \leq h$, we upper bound the probability that the computation path of α reaches a useful node for the first time at step k . Then, we shall use union bound on k . Formally, we bound the probability as follows.

$$\begin{aligned} & \Pr_{\alpha \sim \mu} [\text{computation path of } \alpha \text{ reaches a useful node}] \\ &= \Pr_{\alpha \sim \mu} [\exists k \in [h] : \mathcal{T}(\alpha, k) \text{ is useful and } \mathcal{T}(\alpha, k-1) \text{ is not useful}] \\ &\leq \sum_{k=1}^h \Pr_{\alpha \sim \mu} [\mathcal{T}(\alpha, k) \text{ is useful and } \mathcal{T}(\alpha, k-1) \text{ is not useful}] \\ &\leq h \cdot \max_{k \in [h]} \Pr_{\alpha \sim \mu} [\mathcal{T}(\alpha, k) \text{ is useful} \mid \mathcal{T}(\alpha, k-1) \text{ is not useful}] \end{aligned}$$

We bound the last probability for any $k \in [h]$. Let $p = \mathcal{T}(\alpha, k-1)$. We assume the node p is not useful. Let $(i, j) \in V(G)$ be the lowest vertex that is fixed by C_p to blue. Suppose that in the the next step \mathcal{T} queries a color of the vertex (i', j') . If the next node has to be useful, the response to the query has to be blue. Moreover, there have to be $n/2h$ consecutive layers $i < \ell', \dots, \ell' + \frac{n}{2h} - 1 < i'$ such that C_p does not fix any vertex on those layers. The probability that the response to the query is blue is

$$\Pr_{\alpha \sim \mu | C_p} [\text{The blue path induced by } \alpha \text{ visits } (i', j')].$$

Consider the random walk X_1, \dots, X_{n-1} that determines the blue path P induced by $\alpha \sim \mu$. Recall that the vertices of P are $\{(1, X_1), \dots, (n-1, X_{n-1})\}$. The cube C_p fixes colors of some vertices of G . Let B and R be the set of vertices whose colors are fixed by C_p to blue and red respectively.

Conditioning on the cube C_p restricts the random walk X_1, \dots, X_{n-1} that it must visit the points in B and must not visit the points in R . Formally, for any $(q, y) \in B$ it holds that $X_q = y$ and for any $(q', y') \in R$ it holds that $X_{q'} \neq y'$. We know there is at least one walk that avoids R and visits B (the walk corresponding to P). Moreover, we have $|R| \leq h$ and there is a large “gap” in R , i.e., for each $(i_1, j_1) \in R$ it holds that $i_1 < \ell'$ or $i_1 > \ell' + \frac{n}{2h} - 1$. Recall that we set $h = \gamma n^{1/3}$. Set γ to a sufficiently small constant so that $\frac{n}{2h} \geq ch^2$, where c is a sufficiently large constant for which we can apply Lemma 27 with $c_2 = \varepsilon^{-1}$. By applying an appropriate time shift, we have by Lemma 27 that

$$\Pr_{\alpha \sim \mu | C_p} [\text{The blue path induced by } \alpha \text{ visits } (i', j')] = \Pr_{\mu | C_p} [X_{i'} = j'] \leq \frac{1}{c_2 h} \leq \frac{\varepsilon}{h}.$$

Thus, we conclude that for all $k \in [h]$,

$$\Pr_{\mu \sim \alpha} [\mathcal{T}(\alpha, k) \text{ is useful} \mid \mathcal{T}(\alpha, k-1) \text{ is not useful}] \leq \frac{\varepsilon}{h}.$$

Therefore, the probability that the computation path ever reaches a useful node is at most ε . \blacktriangleleft

We end this subsection with by showing that the formula $\text{SP}_{n,\rho}$ is average-case hard for decision trees.

► Lemma 31. *For any $\varepsilon > 0$, there exists $\gamma > 0$ such that every deterministic decision tree of height at most $\gamma \cdot n^{1/3}$ for $\text{Search}(\text{SP}_{n,\rho})$ makes error with probability $\geq 1 - \varepsilon$ w.r.t. the distribution μ .*

Proof. If the decision tree answers correctly, by Lemma 29 it must reach a useful node at some point. By Lemma 30, the probability of this ever happening is at most ε if γ is small enough. \blacktriangleleft

7.2 Lifting the Average-Case Hardness to Parity Decision Trees

We lift the distribution μ to a distribution μ' of variables of $\text{SP}_{n,\rho} \circ \text{IP}$ as follows:

1. Sample an assignment α according to μ .
2. Sample a uniformly random assignment from $\vec{\text{IP}}^{-1}(\alpha)$.

We remark that an assignment β sampled by μ' falsifies exactly one clause of $\text{SP}_{n,\rho} \circ \text{IP}$, in particular one clause that arises by a lifting clause C of $\text{SP}_{n,\rho}$ where C is the unique clause falsified by the assignment $\vec{\text{IP}}(\beta)$.

In this section, we prove $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$ is average-case hard for parity decision trees of small height under the lifted distribution. To do so, we shall use a result of Chattopadhyay et al. [3], that built upon the earlier work of Göös, Pitassi and Watson [9].

We will need to consider randomized decision trees that output Boolean strings in $\{0, 1\}^t$, rather than 0/1. For a given deterministic 2-party communication protocol Π , let $\Pi(x, y)$ denote the transcript generated by Π on input (x, y) .

23:24 Separation Regular and General Resolution over Parities

► **Theorem 32** (Implicit in [3]). Assume $b \geq 50 \log(m)$. Let Π be any deterministic 2-party communication protocol of cost c , where Alice and Bob each get inputs from $\{0, 1\}^{mb}$. For any $z \in \{0, 1\}^m$, let (X_z, Y_z) denote the distribution on pairs obtained by sampling from $\vec{\text{IP}}^{-1}(z)$ uniformly at random. Then, there exists a randomized decision tree \mathcal{T} of cost $O(c/b)$ such that the following holds for every $z \in \{0, 1\}^m$:

$$d_{TV}(\mathcal{T}(z), \Pi(X_z, Y_z)) \leq 1/10.$$

The above theorem says that a randomized decision tree is able to simulate by probing only a few bits of its input z , the transcript of a deterministic communication protocol when it is given a random input pair X_z, Y_z . Its relevance for us is due to the following simple observation.

► **Observation 33.** Every deterministic parity decision tree of height h can be simulated exactly by a deterministic 2-party communication protocol of cost at most $2h$.

Now we can lift our average-case hardness to parity decision trees.

► **Lemma 34.** There exists a constant $c > 0$ such that for every obfuscation map ρ and every parity decision tree \mathcal{T} of height at most $c \cdot n^{1/3} \log n$ purporting to solve $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$, the following is true:

$$\Pr_{\beta \sim \mu'} \left[\mathcal{T}(\beta) \text{ is falsified on } \beta \right] \leq \frac{2}{5}.$$

Proof. Assume \mathcal{T} makes an error with probability $< 3/5$. Then our main idea is that we would be able to construct an ordinary decision tree for $\text{Search}(\text{SP}_{n,\rho})$ of depth $O(n^{1/3})$ which makes error with probability $< 7/10$ under distribution μ . This contradicts Lemma 31.

Using Observation 33, we get a deterministic 2-party protocol of cost at most $2 \cdot \text{depth}(\mathcal{T})$ that makes error less than $3/5$ for solving $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$. Theorem 32 then yields a randomized decision tree \mathcal{T}' with the following properties. On input $\alpha \sim \mu$,

- \mathcal{T}' makes at most $O(\text{cost}(\Pi)/\log n)$ queries to α , i.e., at most $O(n^{1/3})$.
- If \mathcal{D}_1 denotes the actual distribution of the transcript of Π when it is run on input sampled uniformly at random from $\vec{\text{IP}}^{-1}(\alpha)$ and \mathcal{D}_2 denotes the distribution of the transcript of Π simulated by \mathcal{T}' ,

$$\|\mathcal{D}_1 - \mathcal{D}_2\| \leq \frac{1}{10}$$

We now modify \mathcal{T}' to output a clause as follows. A transcript of Π leads it to output a clause of $\text{SP}_{n,\rho} \circ \text{IP}$. The modified \mathcal{T}' outputs the unique corresponding un-lifted clause of $\text{SP}_{n,\rho}$. The probability of error is at most $\Pr[\Pi \text{ errs}] + 1/10 < 7/10$. This gives a randomized decision tree; by fixing the coins we can replace it by a deterministic decision tree, contradicting Lemma 31. ◀

7.3 Foolable Nodes Are Frequent

Let \mathcal{P} be a bottom-read-once linear branching program for $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$ that corresponds to a bottom-regular ResLin proof of the unsatisfiability of $\text{SP}_{n,\rho} \circ \text{IP}$. Our goal is to show size of \mathcal{P} is large. To do so, we establish that the affine spaces associated with many nodes of \mathcal{P} have a certain property that allows to fool them. In particular, let β be an assignment sampled by μ' . We will prove that with high probability after making $t = O(n^{1/3})$ linear queries, according to β , we will end in a node p of \mathcal{P} such that the associated affine space A_p

does not have much information about β . We will show that this implies the affine space A_p contains many useful assignments that allows us to prove the co-dimension of A_p is large. Now, we define the sought property formally.

► **Definition 35.** Let $\alpha \in \text{Supp}(\mu)$ and P be the blue path induced by α that ends at v . Let u and w be the two children of v . We say a subset $T \subseteq [m]$ is α -foolable if T does not contain any variable mentioning v, u or w , i.e. the variables $P_{x,i}, Z_{x,j}$ for $x \in \{u, v, w\}, i \in [N], j \in [N - 1]$.

► **Definition 36.** An affine space $A \subseteq \mathbb{F}_2^{mb}$ is α -foolable if $\text{Cl}(A)$ is α -foolable with $\alpha \in \text{Supp}(\mu)$ and there exists $\beta \in A$ such that $\alpha = \overrightarrow{\text{IP}}(\beta)$.

We call a node p of \mathcal{P} α -foolable if the associated affine space A_p is α -foolable. Recall that $\mathcal{P}(\beta, t)$ is the node that \mathcal{P} arrives at after making t linear queries on β . It turns out the node $\mathcal{P}(\beta, t)$ is α -foolable with high probability if t is sufficiently small. We prove this in the following theorem.

► **Theorem 37.** Let \mathcal{P} be any bottom-read-once linear branching program corresponding to a bottom-regular *ResLin* proof of $\text{SP}_{n,\rho} \circ \text{IP}$. There exists a constant $c > 0$ such that if $t < c \cdot n^{1/3}$, then

$$\Pr_{\alpha \sim \mu, \beta \sim \overrightarrow{\text{IP}}^{-1}(\alpha)} \left[\mathcal{P}(\beta, t) \text{ is } \alpha\text{-foolable} \right] > \frac{3}{5}.$$

Proof. Let p denote the random node $\mathcal{P}(\beta, t)$. Let the blue path induced by α end at v and let the children of v be u and w . Notice that the second condition of being α -foolable (that A_p contains an element of $\overrightarrow{\text{IP}}^{-1}(\alpha)$) is always satisfied by $\mathcal{P}(\beta, t)$ since one such element is β . We just need to lower bound the probability of the first condition of α -foolability being satisfied, i.e., the probability that $\text{Cl}(A_p)$ does not contain any variable mentioning u, w or v .

We construct a PDT \mathcal{T} for $\text{Search}(\text{SP}_n \circ \text{IP})$ from \mathcal{P} in the following manner: on input β , it will simulate the path traced out in \mathcal{P} for t steps by making precisely those linear queries that would have been issued in \mathcal{P} . At the end of it, \mathcal{T} does the following: Let A be the affine space corresponding to the queries issued and responses received so far. For every vertex $k = (i, j)$ in the pyramid graph G_n such that one of its variables ($P_{k,\ell}$ or $Z_{k,\ell}$ for some $\ell \in [N]$) is in $\text{Cl}(A)$, query the b coordinates from the blocks of the following set of variables:

$$S = \{R_{f^{-1}(x)} \mid x \in \{(i-1, j-1), (i-1, j), (i, j-1), (i, j), (i, j+1), (i+1, j), (i+1, j+1)\} \cap V(G)\}$$

If one of the induction clauses mentioning only stones in S is falsified (recall that the placement of stones to vertices is the same for all assignments in μ), output the corresponding clause. Otherwise, output an error symbol.

Clearly, the depth of \mathcal{T} is $O(n^{1/3} \log n)$. Thus, by Lemma 34, the probability that it outputs a falsified clause is at most $2/5$. Let A_p denote the affine space at $\mathcal{P}(\beta, t)$. Note that $A \subseteq A_p \implies \text{Cl}(A_p) \subseteq \text{Cl}(A)$, by Lemma 13. It is straight-forward to verify that if A_p is not α -foolable, then \mathcal{T} successfully outputs a clause falsified by β : if one of the variables belonging to u, w or v is in $\text{Cl}(A)$, in the final step the PDT queries the stones placed on u, w, v and detects that an induction clause at v is falsified. The result now follows from Lemma 34. ◀

7.4 Foolability Implies Large Rank

In this subsection, we prove there is an obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$ such that for a foolable node v of a bottom-read-once linear branching program \mathcal{P} computing $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$, the associated affine space A_v must have large co-dimension.

23:26 Separation Regular and General Resolution over Parities

First, we prove an auxiliary lemma. Let $T \subseteq [m]$ be a subset of the variables of $\text{SP}_{n,\rho}$. We say a stone j is *marked* by T if T contains a variable that mentions the stone j , i.e. $R_j, P_{v,j}$ for any vertex $v \in V$, $P_{f(j),k}$ for any stone $k \in [N]$, or $Z_{f(j),\ell}$ for any $\ell \in [N-1]$. Let $Q(T) \subseteq [N]$ be the set of stones marked by T .

► **Lemma 38.** *Let ρ be any obfuscation map and $\alpha \in \text{Supp}(\mu)$. Let v be the vertex at which the blue path induced by α ends. Let $T \subseteq [m]$ be an α -foolable subset with $|Q(T)| < N/2$. For any $i, j, k \in [N] \setminus Q(T)$, there exists an assignment $\gamma \in \{0, 1\}^m$ extending the restriction $\alpha|_T$ which satisfies all clauses of $\text{SP}_{n,\rho}$ except one of the following two:*

$$\begin{aligned} C_1 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i, j, k), \text{ or} \\ C_2 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg \rho(i, j, k), \end{aligned}$$

where u and w are the out-neighbors of v .

Proof. Let $S \subseteq V(G)$ be the set of vertices in the blue path induced by α . We assign stone k to vertex v and stones i, j to u, w respectively. To all other vertices we assign arbitrary stones as long as they are consistent with $\alpha|_T$. Formally: pick two stones ℓ_1, ℓ_2 in $[N] \setminus (Q(T) \cup f^{-1}(S) \cup \{i, j, k\})$. Consider the following map $\text{STONE} : V \rightarrow [N]$.

$$\text{STONE}(p) = \begin{cases} f^{-1}(p) & \text{if } f^{-1}(p) \in Q(T) \\ i & \text{if } p = v \\ j & \text{if } p = u \\ k & \text{if } p = w \\ \ell_1 & \text{if } f^{-1}(p) \notin Q(T) \cup \{i, j, k\}, f^{-1}(p) \in S \\ \ell_2 & \text{if } f^{-1}(p) \notin Q(T) \cup \{i, j, k\} \cup S \end{cases}$$

Define a coloring map $\text{COLOR} : [N] \rightarrow \{\text{RED}, \text{BLUE}\}$ as follows:

$$\text{COLOR}(s) = \begin{cases} \text{RED} & \text{if } s \in Q(T), f(s) \notin S \\ \text{BLUE} & \text{if } s \in Q(T), f(s) \in S \\ \text{BLUE} & \text{if } s = i \\ \text{RED} & \text{if } s = j \\ \text{RED} & \text{if } s = k \\ \text{BLUE} & \text{if } s = \ell_1 \\ \text{RED} & \text{if } s = \ell_2 \\ \text{BLUE} & \text{otherwise} \end{cases}$$

We remark the color used in the last case does not matter as these stones are not used in the stone placement given by the map STONE . Let $\gamma \in \{0, 1\}^m$ be the assignment which sets the variables according to this placement and coloring map, i.e.,

$$\begin{aligned} P_{v,j} &= \begin{cases} 1 & \text{if } j = \text{STONE}(v) \\ 0 & \text{otherwise} \end{cases} \\ Z_{v,j} &= \begin{cases} 0 & \text{if } j < \text{STONE}(v) \\ 1 & \text{otherwise} \end{cases} \\ R_j &= \begin{cases} 1 & \text{if } \text{COLOR}(j) = \text{RED} \\ 0 & \text{if } \text{COLOR}(j) = \text{BLUE} \end{cases} \end{aligned}$$

Notice that this assignment is consistent with α on T and it falsifies a single induction clause at v : stone i is placed at v , stones j, k are placed at u, w respectively; stones j, k are red while stone i is blue. We remark that there is no set of clause in $\text{SP}_{n,\rho}$ which forces the placement of stones to vertices to be bijective. Thus, the only clause of $\text{SP}_{n,\rho}$ falsified by γ is one of the following:

$$\begin{aligned} C_1 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i, j, k), \\ C_2 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg \rho(i, j, k), \end{aligned} \quad \blacktriangleleft$$

For our hard formula, we use an appropriate obfuscation map ρ given by the following lemma. An analogous lemma was proved by Alekhovich et al. [1] with different constants as their formula is over a slightly smaller set of constants, but still quadratic in N .

► **Lemma 39.** *For N sufficiently large, there exists a mapping $\rho : [N]^3 \rightarrow \mathcal{V}$ such that for every $Q \subseteq [N]$ with $|Q| \leq N/400$ and every $X \in \mathcal{V}$, there exist $i < j < k \in [N] \setminus Q$ such that $\rho(i, j, k) = X$.*

We remark that the proof of the following lemma is the only place where we are using an obfuscation map with a certain property (given by Lemma 39) and also that the branching program computing $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$ is bottom-read-once.

► **Lemma 40.** *There exists an obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$ such that the following holds. Let $\beta \in \mathbb{F}_2^{mb}$, $t > 0$, and $p = \mathcal{P}(\beta, t)$ be a node in a bottom-read-once branching program \mathcal{P} computing $\text{Search}(\text{SP}_{n,\rho} \circ \text{IP})$. If p is α -foolable for $\alpha = \vec{\text{IP}}(\beta)$, then $\text{co-dim}(A_p) \geq \min\{\frac{N}{800}, t\}$.*

Proof. Fix $\rho : [N]^3 \rightarrow \mathcal{V}$ to be a map with the property guaranteed by Lemma 39. Let $A_p = \mathcal{S}(M, c)$. Suppose $\text{rank}(M) \leq \frac{N}{800}$. Let S be the set of sinks of \mathcal{P} reachable from p .

▷ **Claim 41.** For each variable Y of $\text{SP}_{n,\rho} \circ \text{IP}$ there is a sink in S outputting a clause D such that Y or $\neg Y$ is in D .

Proof of Claim 41. We shall show that there exists an assignment $\gamma \in A_p$ such that γ falsifies only one clause of $\text{SP}_{n,\rho} \circ \text{IP}$ and that clause contains Y or $\neg Y$. Let $Y \in \text{BLOCK}(Z)$. Let v be the endpoint of the blue path induced by α , and let its children be u and w .

Let $T = \text{Cl}(A_p)$. Note that $|T| < N/800$, thus $Q(T) < N/400$. By Lemma 39, there exist $i < j < k$ in $[N] \setminus Q(T)$ such that $\rho(i, j, k) = Z$. By assumption, T is α -foolable. By Lemma 38, we can extend the restriction $\alpha|_T$ to a full assignment $\gamma \in \mathbb{F}_2^m$ such that γ does not falsify any clause of $\text{SP}_{n,\rho}$ other than one of the following two clauses:

$$\begin{aligned} C_1 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee Z \\ C_2 &:= \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg Z. \end{aligned}$$

By Lemma 17, there is an assignment $\beta' \in A_p$ such that $\vec{\text{IP}}(\beta') = \gamma$. Note that β' falsifies only one clause of $\text{SP}_{n,\rho} \circ \text{IP}$, and that clause belongs to either $C_1 \circ \text{IP}$ or $C_2 \circ \text{IP}$. By Observation 3, every clause in $C_1 \circ \text{IP}$ and $C_2 \circ \text{IP}$ contains every variable in the block of $\rho(i, j, k) = Z$ (possibly with negations). Thus, the only clause falsified by β' contains either Y or $\neg Y$. Since $\beta' \in A_p$ and β' falsifies only one clause \tilde{C} of $\text{SP}_{n,\rho} \circ \text{IP}$, one of the sinks in S must be labelled \tilde{C} . \triangleleft

23:28 Separation Regular and General Resolution over Parities

We continue the proof of Lemma 40. Let U be the space spanned by rows of matrices defining the spaces of sinks in S . Formally, let $s \in S$ be labelled by the affine space $A_s = \mathcal{S}(M_s, c_s)$ (this corresponds to a clause of $\text{SP}_{n,\rho} \circ \text{IP}$). Then, $U = \text{Span}(\{\mathcal{R}(M_s) \mid s \in S\})$. By Claim 41 each variable of $\text{SP}_{n,\rho} \circ \text{IP}$ is mentioned at some sink in S , so the space U has full dimension, i.e., $\dim(U) = mb$. Let $W = \text{Span}(\mathcal{R}(M) \cup \text{Post}(p))$. By Lemma 7, we have

$$\dim(W) \leq \text{rank}(M) + \dim(\text{Post}(p)) \leq \text{rank}(M) + mb - t.$$

On the other hand by Lemma 8, $U \subseteq W$ and thus, $\dim(W) \geq \dim(U) = mb$. Putting both inequalities together, we get $\text{rank}(M) \geq t$. \blacktriangleleft

7.5 Lifted Distributions Fool Rank

In the earlier two subsections, we have established the following two facts: (i) in any BROLBP \mathcal{P} corresponding to a bottom-regular ResLin proof of $\text{SP}_{n,\rho} \circ \text{IP}$, when inputs β are sampled according to the IP lift of μ , the node $\mathcal{P}(\beta, t)$ is a foolable node with high probability; (ii) in such a BROLBP, the constraint matrix for the affine space associated with a foolable node has large rank.

To prove that \mathcal{P} has large size, it is sufficient to argue that each large rank constraint system is satisfied with small probability under the IP lift of μ . Of course, such a statement is well known to be true if we sample inputs from the uniform distribution in \mathbb{F}_2^{bm} . However, our distribution is not so at all. In particular, it has quite sparse support. Still, it turns out that any lifted distribution is pseudo-random with respect to the rank measure if the gadget satisfies a generalization of the stifling property. Consider a gadget $g : \{0, 1\}^b \rightarrow \{0, 1\}$. For any $i \in [b]$, and $o \in \{0, 1\}$ an assignment α to the bits different from i is called *o-stifling for i* if g gets fixed to o by α , i.e., the induced subfunction $g_{|[b] \setminus \{i\} \leftarrow \alpha}$ gets fixed to the constant function that always evaluates to o , no matter how the i -th bit is set. We say g is ε -balanced, *stifled* if for any $i \in [b]$, and for any $o \in \{0, 1\}$, the following is true: when we sample $x \in \{0, 1\}^b$ uniformly at random from $g^{-1}(o)$, the projection of x on co-ordinates different from i is *o-stifling for i* with probability at least ε .

\triangleright **Claim 42.** Inner-product defined on $2b \geq 8$ bits is a $7/18$ -balanced and stifled gadget.

Proof. By simple manipulation,

$$\Pr_{(x,y) \sim \{0,1\}^{2b}} [\text{IP}(x, y) = 0 \wedge x_1 = 0] = \mathbb{E} \left[\left(\frac{1 + (-1)^{\sum_{i=1}^b x_i y_i}}{2} \right) \left(\frac{1 + (-1)^{x_1}}{2} \right) \right].$$

The RHS becomes,

$$\frac{1}{4} + \frac{1}{4} \mathbb{E}[(-1)^{x_1}] + \frac{1}{2} \mathbb{E}[(-1)^{\sum_{i=1}^b x_i y_i}]$$

The second sum is 0, and the third is at most $\frac{1}{2^{b+1}}$. Overall, this gives that

$$\Pr_{(x,y) \sim \{0,1\}^{2b}} [\text{IP}(x, y) = 0 \wedge x_1 = 0] \geq \frac{1}{4} - \frac{1}{2^{b+1}}.$$

Further, by a similar method,

$$\Pr_{(x,y) \sim \{0,1\}^{2b}} [\text{IP}(x, y) = 0] \leq \frac{1}{2} + \frac{1}{2^b}.$$

Thus,

$$\Pr_{(x,y) \sim \{0,1\}^{2b}} [x_1 = 0 \mid \text{IP}(x,y) = 0] \geq \frac{1/4 - 1/2^{b+1}}{1/2 + 1/2^b}.$$

Note that any assignment that sets $x_1 = 0$ stifles y_1 . Hence, y_1 is stifled with probability at least $7/18$, if $b \geq 4$, by the projection of a random 0 (and similarly 1) assignment to IP. Completely analogously, any bit is stifled with the same probability. \triangleleft

► **Remark 43.** It is worth noting that several gadgets, including Inner-Product, Indexing, Majority, even on sufficiently large but constant number of bits, are stifled and balanced.

Now we state the utility of balanced, stifling gadgets.

► **Lemma 44.** *Let g be any ε -balanced, stifled gadget and $z \in \mathbb{F}_2^n$ be any fixed vector. Then, for every matrix $M \in \mathbb{F}_2^{r \times bm}$ of full rank r , and any vector $\gamma \in \mathbb{F}_2^r$ the following holds:*

$$\Pr_{\beta \sim \vec{g}^{-1}(z)} [M\beta = \gamma] = 2^{-\Omega_\varepsilon(r/b)}.$$

Proof. After Gaussian elimination, turning M into row-echelon form, there are at least r/b different blocks in which pivots of rows appear. Let us call each such block a pivot block. The distribution $\vec{g}^{-1}(z)$ samples independently at random from $g^{-1}(z_i)$ for each of the i -th block. It will be convenient to think that we sample, one after the other, independently from blocks in this way, starting from the rightmost. Consider the situation when we arrive at a pivot block having sampled all blocks to its right. Let the bit of z corresponding to this pivot block be $o \in \{0,1\}$. Consider any one row that has a pivot in that block. Let the equation corresponding to this row be denoted by ℓ . By the property of g , with probability at least ε the random assignment from $g^{-1}(o)$ will be stifling for the pivot of ℓ . Conditioned on that event, the stifled bit will be set to each of 0, 1 with probability exactly $1/2$ as each possible setting gives rise to a distinct assignment in $g^{-1}(o)$. Hence, the probability that ℓ is satisfied by the sampled assignment to this block is at most $(1 - \varepsilon/2)$. Thus, continuing this way, the probability that all the equations are satisfied is at most $(1 - \varepsilon/2)^{r/b}$, yielding the desired result. \blacktriangleleft

7.6 Putting Everything Together

Now, we are ready to finish the proof of our lower bound, i.e., Theorem 25.

Proof of Theorem 25. Let \mathcal{P} be the BROLBP derived from a bottom-regular ResLin proof of $\text{SP}_{n,\rho} \circ \text{IP}$. Let $t = \lfloor c \cdot n^{1/3} \rfloor$ for an appropriately chosen small constant $c > 0$. Combining Theorem 37 and Lemma 40, we get

$$\Pr_{\beta \sim \mu'} [\text{co-dim}(A_{\mathcal{P}(\beta,t)}) \geq t] \geq \Pr_{\alpha \sim \mu, \beta \sim \vec{\text{IP}}^{-1}(\alpha)} [\mathcal{P}(\beta,t) \text{ is } \alpha\text{-foolable}] \geq \frac{3}{5}. \quad (3)$$

On the other hand, for any node v of \mathcal{P} which has $\text{co-dim}(A_v) \geq t$, Lemma 44 yields,

$$\Pr_{\beta \sim \mu'} [\mathcal{P}(\beta,t) = v] \leq 2^{-\Omega\left(\frac{t}{\log n}\right)}. \quad (4)$$

If s is the total number of nodes of \mathcal{P} , combining (3) and (4), we get immediately

$$\begin{aligned} \frac{3}{5} &\leq \Pr_{\beta \sim \mu'} [\text{co-dim}(A_{\mathcal{P}(\beta,t)}) \geq t] = \sum_{v \text{ node of } \mathcal{P}: \text{co-dim}(A_v) \geq t} \Pr_{\beta \sim \mu'} [\mathcal{P}(\beta,t) = v] \\ &\leq s \cdot 2^{-\Omega\left(t/\log(n)\right)} \end{aligned}$$

Substituting the value of t in the above inequality, we immediately get $s \geq 2^{\Omega\left(n^{1/3}/\log(n)\right)}$ \blacktriangleleft

8 Future Directions

We provided the first super-polynomial separation between the power of bottom-regular and power of general ResLin proofs. We believe the general proof strategy that we implemented, modifying and generalizing the recent technique of Efremenko, Garlík and Itsykson [7], should yield exponential lower bounds on the length of bottom-regular ResLin proofs for other formulas as well. For instance, formula MGT_n which is the constant-width version of GT_n , that encodes the contradiction that a finite total order has no minimal element, when obfuscated appropriately and then lifted with inner-product can be proved to be hard for bottom-regular ResLin. Indeed, the first part of our proof strategy is quite general and applies to all ResLin proofs without any assumption on regularity. Here, we need just the fact that the search for a falsified clause in the base formula is hard on average for small height decision trees w.r.t some distribution μ . That is sufficient, thanks to lifting theorems, to yield the fact that after the first few (typically $n^{\Omega(1)}$) linear queries, the branching program corresponding to the ResLin proof is still far from discovering a falsified clause in the lifted formula with high probability, when the input is sampled from the lifted distribution $\vec{g}^{-1}(\mu)$. This is step 2 of our proof outline. In most formulas, one could then define some natural notion of *foolability* and then say the affine space associated with nodes of the branching program are frequently foolable. How do we know this is useful? Unfortunately, the usefulness of these notions seem formula-specific. For the binary pigeonhole principle, Efremenko et al. observed that local consistency was enough to yield high rank of the dual of the affine space. For our formula, we achieved the same exploiting the obfuscation map and stifling nature of our lifting gadget. But this seems not immediately generalizable. An interesting direction here is the following:

► **Problem 1.** *Prove strong lower bounds on the size of bottom-regular proofs for the lift of Tseitin formulas over expander graphs.*

Another direction is to consider the formulas where even implementing the first step of our strategy seems impossible.

► **Problem 2.** *Prove strong lower bounds on size of bottom-regular ResLin proofs for appropriate lifts of random constant-width CNF formulas.*

The above seems challenging as for random formulas of constant-width, for every distribution, there exists a decision tree that finds in $O(1)$ queries a falsified clause with high probability. This, very likely, requires changing our technique substantially. Finally, one of the challenges posed by Gryaznov et al. [10] remains still open.

► **Problem 3.** *Prove super-polynomial lower bounds on the size of top-regular ResLin proofs.*

References




- 1 Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *Theory of Computing*, 3(5):81–102, 2007. Preliminary version in STOC, 2002. doi:10.4086/toc.2007.v003a005.
- 2 Paul Beame and Sajin Korothe. On disperser/lifting properties of the index and inner-product functions. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPICs*, pages 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ITCS.2023.14.

- 3 Arkadev Chattopadhyay, Yuval Filmus, Sajin Koroth, Or Meir, and Toniann Pitassi. Query-to-communication lifting using low-discrepancy gadgets. *SIAM J. Comput.*, 50(1):171–210, 2021. Preliminary version in ICALP, 2019. doi:10.1137/19M1310153.
- 4 Arkadev Chattopadhyay, Nikhil S. Mande, Swagato Sanyal, and Suhail Sherif. Lifting to parity decision trees via stifling. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10–13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPICs*, pages 33:1–33:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ITCS.2023.33.
- 5 Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. *Electron. Colloquium Comput. Complex.*, TR22-153, 2022. arXiv:TR22-153.
- 6 Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17–20, 2023, Warwick, UK*, volume 264 of *LIPICs*, pages 9:1–9:27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.CCC.2023.9.
- 7 Klim Efremenko, Michal Garlík, and Dmitry Itsykson. Lower bounds for regular resolution over parities. *Electron. Colloquium Comput. Complex.*, TR23-187, 2023. arXiv:TR23-187.
- 8 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. *Theory Comput.*, 16:1–30, 2020. Preliminary version in STOC 2018. doi:10.4086/TOC.2020.V016A013.
- 9 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. *SIAM J. Comput.*, 49(4), 2020. Preliminary version in FOCS 2017. doi:10.1137/17M115339X.
- 10 Svyatoslav Gryaznov, Pavel Pudlák, and Navid Talebanfard. Linear branching programs and directional affine extractors. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20–23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 4:1–4:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.CCC.2022.4.
- 11 Dmitry Itsykson and Dmitry Sokolov. Lower bounds for splittings by linear combinations. In Erzsébet Csuhaj-Varjú, Martin Dietzfelbinger, and Zoltán Ésik, editors, *Mathematical Foundations of Computer Science 2014 – 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25–29, 2014. Proceedings, Part II*, volume 8635 of *Lecture Notes in Computer Science*, pages 372–383. Springer, 2014. doi:10.1007/978-3-662-44465-8_32.
- 12 Dmitry Itsykson and Dmitry Sokolov. Resolution over linear equations modulo two. *Annals of Pure and Applied Logic*, 171(1):102722, 2020. doi:10.1016/j.apal.2019.102722.
- 13 Xin Li. Two source extractors for asymptotically optimal entropy, and (many) more. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1271–1281, 2023. doi:10.1109/FOCS57990.2023.00075.
- 14 Xin Li and Yan Zhong. Explicit directional affine extractors and improved hardness for linear branching programs. *CoRR*, abs/2304.11495, 2023. doi:10.48550/arXiv.2304.11495.
- 15 Pavel Pudlák. On extracting computations from propositional proofs (a survey). In Kamal Lodaya and Meena Mahajan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15–18, 2010, Chennai, India*, volume 8 of *LIPICs*, pages 30–41. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPICs.FSTTCS.2010.30.
- 16 Ran Raz and Iddo Zameret. Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Log.*, 155(3):194–224, 2008. doi:10.1016/J.APAL.2008.04.001.
- 17 Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded-arithmetic. *Izvestiya. Math.*, 59(1):205–227, 1995.
- 18 Alexander A. Razborov. A new kind of tradeoffs in propositional proof complexity. *J. ACM*, 63(2):16:1–16:14, 2016. doi:10.1145/2858790.
- 19 Dmitry Sokolov. Dag-like communication and its applications. In Pascal Weil, editor, *Computer Science – Theory and Applications – 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8–12, 2017, Proceedings*, volume 10304 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9_26.

23:32 Separation Regular and General Resolution over Parities

- 20 Alasdair Urquhart. A near-optimal separation of regular and general resolution. *SIAM J. Comput.*, 40(1):107–121, 2011. doi:10.1137/090772897.
- 21 Marc Vinyals, Jan Elffers, Jan Johannsen, and Jakob Nordström. Simplified and improved separations between regular and general resolution by lifting. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020 – 23rd International Conference, Alghero, Italy, July 3-10, 2020, Proceedings*, volume 12178 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2020. doi:10.1007/978-3-030-51825-7_14.

Distribution-Free Proofs of Proximity

Hugo Aaronson   

Department of Computer Science and Technology, University of Cambridge, UK

Tom Gur   

Department of Computer Science and Technology, University of Cambridge, UK

Ninad Rajgopal  

Department of Computer Science and Technology, University of Cambridge, UK

Ron D. Rothblum  

Faculty of Computer Science, Technion, Haifa, Israel

Abstract

Motivated by the fact that input distributions are often unknown in advance, distribution-free property testing considers a setting in which the algorithmic task is to accept functions $f : [n] \rightarrow \{0, 1\}$ having a certain property Π and reject functions that are ε -far from Π , where the distance is measured according to an arbitrary and unknown input distribution $\mathcal{D} \sim [n]$. As usual in property testing, the tester is required to do so while making only a sublinear number of input queries, but as the distribution is unknown, we also allow a sublinear number of samples from the distribution \mathcal{D} .

In this work we initiate the study of *distribution-free interactive proofs of proximity* (df-IPPs) in which the distribution-free testing algorithm is assisted by an all powerful but untrusted prover. Our main result is that for any problem $\Pi \in \text{NC}$, any proximity parameter $\varepsilon > 0$, and any (trade-off) parameter $\tau \leq \sqrt{n}$, we construct a df-IPP for Π with respect to ε , that has query and sample complexities $\tau + O(1/\varepsilon)$, and communication complexity $\tilde{O}(n/\tau + 1/\varepsilon)$. For τ as above and sufficiently large ε (namely, when $\varepsilon > \tau/n$), this result matches the parameters of the best-known general purpose IPPs in the standard uniform setting. Moreover, for such τ , its parameters are optimal up to poly-logarithmic factors under reasonable cryptographic assumptions for the same regime of ε as the uniform setting, i.e., when $\varepsilon \geq 1/\tau$.

For smaller values of ε (i.e., when $\varepsilon < \tau/n$), our protocol has communication complexity $\Omega(1/\varepsilon)$, which is worse than the $\tilde{O}(n/\tau)$ communication complexity of the uniform IPPs (with the same query complexity). With the aim of improving on this gap, we further show that for IPPs over specialised, but large distribution families, such as sufficiently smooth distributions and product distributions, the communication complexity can be reduced to $\tilde{O}(n/\tau^{1-o(1)})$. In addition, we show that for certain natural families of languages, such as symmetric and (relaxed) self-correctable languages, it is possible to further improve the efficiency of distribution-free IPPs.

2012 ACM Subject Classification Theory of computation \rightarrow Interactive proof systems

Keywords and phrases Property Testing, Interactive Proofs, Distribution-Free Property Testing

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.24

Related Version *Full Version*: <https://ecc.weizmann.ac.il/report/2023/118> [1]

Funding Tom Gur and Ninad Rajgopal are supported by the Tom Gur's UKRI Future Leaders Fellowship MR/S031545/1. Tom Gur is also supported in part by EPSRC New Horizons Grant EP/X018180/1 and EPSRC RoaRQ Grant EP/W032635/1. Ron Rothblum is funded by the European Union (ERC, FASTPROOF, 101041208). Part of this work was completed when the first three authors were affiliated with the University of Warwick. 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. Neither the European Union nor the granting authority can be held responsible for them.

Acknowledgements We are grateful to Oded Goldreich for his insightful comments, some of which led to rephrasing Theorem 1 to indicate the trade-off between the query and communication complexities with better clarity. We are also thankful to Marcel Dall'Agnol for many helpful discussions.



© Hugo Aaronson, Tom Gur, Ninad Rajgopal, and Ron D. Rothblum;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 24; pp. 24:1–24:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



1 Introduction

Property Testing, initiated in [45, 25], is a rich and well-studied research field lying at the heart of many advancements in sublinear algorithms and complexity theory; see [21, 7] for a detailed introduction. Loosely speaking, a testing algorithm for a property Π is given oracle access to an input function $f : [n] \rightarrow \{0, 1\}$ and should decide whether $f \in \Pi$ using a small *sublinear* number of queries. As we cannot expect to do so exactly, the tester is required to distinguish between inputs that are in Π from those that are ε -far from every function in Π . Here, distance is typically measured using the relative Hamming distance – namely, the fraction of outputs of f that need to be changed to reach a member of Π .

While modeling distance using the relative Hamming distance is natural and convenient, in many settings it may not capture the underlying question (for example, when functions always satisfy a particular format or when some parts in the domain are more important than others). Following the Probably-Approximately-Correct (PAC) learning model, introduced by Valiant in his celebrated work in computational learning theory [49], *distribution-free* algorithms have widely been accepted as a closer abstraction of real-world computational tasks that are required to make decisions based on limited access to the input data. In this spirit, [25] introduced *distribution-free property testing*, where the distance between two functions is with respect to a distribution \mathcal{D} (over inputs to the function), which is *arbitrary* and *unknown* to the testing algorithm. Since \mathcal{D} is unknown, in addition to the query oracle to the input $f : [n] \rightarrow \{0, 1\}$, the tester can draw independent identically distributed random labelled samples $(i, f(i))$ from a *sample oracle*, where each index i is generated independently from the distribution \mathcal{D} . The tester is required to reject any function that is ε -far¹ from Π along the unknown distribution \mathcal{D} , and the only access that the tester has to \mathcal{D} is via the sample oracle.

The distribution-free model of testing naturally complements the PAC-learning model, and profound bidirectional connections are known between them.² Moreover, distribution-free testing is motivated by the fact that it captures the realistic setting where the tester is required to maintain its guarantees despite dealing with data from an unknown environment (i.e., via data samples from some unknown and arbitrary distribution \mathcal{D}). It also deals with situations where not all underlying data points are equally important, e.g., in graphs where certain edges or vertices are more important than others, and one would like to consider distributions that weigh them appropriately.

Following [25], several distribution-free testing algorithms have been designed for function classes including monotone Boolean functions and low-degree polynomials over finite fields [33], k -juntas [37, 11, 3], conjunctions (monotone or non-monotone) and linear threshold functions [19, 13], polynomial threshold functions and decision trees [8], halfspaces [8, 12], and low-degree polynomials on \mathcal{R}^n [18, 2]. Distribution-free testing has also been studied for graph properties including connectivity [34], bipartiteness [22], k -path and degree regularity [23], as well as for word problems like subsequence-freeness [41].

Despite such strides of progress, our understanding of distribution-free testing is much more limited than that of testing with respect to the uniform distribution. This is due to the multitude of challenges that arise in designing algorithms that need to deal with data samples that can come from any arbitrary distribution, which in turn, makes the model significantly more involved.

¹ We say $f : [n] \rightarrow \{0, 1\}$ is ε -far from a (non-empty) property Π along \mathcal{D} , if for every $f' : [n] \rightarrow \{0, 1\}$ such that $f' \in \Pi$, it holds that $\mathbb{P}_{i \sim \mathcal{D}}[f(i) \neq f'(i)] > \varepsilon$.

² In particular, in [25], it is shown that if a class of functions \mathcal{C} has a *proper* PAC-learner using membership queries (where the learner outputs an approximate hypothesis that also belongs to \mathcal{C}), then \mathcal{C} has a distribution-free tester that uses roughly the same number of queries and samples as the learner.

This paper aims to bridge the gap between testing over the uniform distribution and distribution-free testing by capitalising on the power of interactive proofs, and delegating the task of handling the challenges imposed by the distribution-free setting to a powerful, but untrusted, prover.

1.1 Distribution-free Interactive Proofs of Proximity

In this work, we initiate the study of *distribution-free interactive proofs of proximity* (distribution-free IPPs), which are distribution-free testers that are augmented with the help of a prover. In the rest of this paper, for convenience, rather than thinking of the input as a function, we view it as a string $x \in \{0, 1\}^n$ (which can be similarly be viewed as a truth table of a function $f_x : [n] \rightarrow \{0, 1\}$). Correspondingly, we view a property Π of functions as a language L over strings (which may be viewed as truth tables of the functions in Π).

Thus, distribution-free IPPs are protocols where a *sublinear* time, randomised algorithm, called the verifier, interacts with an untrusted prover to decide whether the given input $x \in \{0, 1\}^n$ belongs to the language L or is far from such, where distance is measured with respect to a fixed, but unknown distribution \mathcal{D} over $[n]$. The verifier is given access to the input x through a query oracle, as well as a sample oracle with respect to \mathcal{D} , while the prover can look at the input entirely. We assume that the prover does not know the queries that the verifier makes to either of its oracles.

We require that for any $x \in L$, there exists an honest prover that interacts with the verifier and convinces it to accept with high probability, while when x is ε -far from L with respect to the distribution \mathcal{D} , no cheating prover, even computationally unbounded, will make the verifier accept, except with low probability. Further, we require the distribution-free IPP to meet these requirements, with respect to the underlying (and unknown) distribution \mathcal{D} from which the oracle draws samples.

In this setting, the verifier's *query complexity* and *sample complexity*, the number of bits exchanged in the protocol, i.e., the *communication complexity*, and the verifier's running time should all be sublinear in input length. Other complexity parameters of interest are the number of rounds of interaction, and the (honest) prover's running time.

Distribution-free IPPs capture the distribution-free property testing analogue of interactive proofs (for more information, see Section 1.4). As such, similar to uniform IPPs, distribution-free IPPs can be alternatively viewed as proof systems where the bounded verifier need only be convinced of the fact that the input is close to the language, by interacting with a more powerful prover. One of the main goals of distribution-free IPPs is to overcome the inherent limitations of distribution-free testing algorithms by showing that for certain properties, verifying proximity over arbitrary distributions is considerably faster with a prover than actually testing it. In particular, we want to design distribution-free IPPs (with sublinear query complexity) for rich families of properties that have no known distribution-free testers.

Of close relevance are the well-studied notion of IPPs over the uniform distribution, which we refer to in this work as Uniform IPPs, that were introduced in [16, 44] (and are trivially generalised by distribution-free IPPs). Showcasing the power of interaction, [44] constructed highly non-trivial uniform IPPs for every language that can be decided in bounded depth (e.g., NC), which was recently made near-optimal by [43] (see [36] for the conditional matching lower bound), and strengthened to encompass also bounded space languages [40].

Motivated intrinsically and by natural applications to *delegation of computation*, the study of uniform IPPs has drawn much recent attention on its own right [44, 32, 36, 40, 26, 20]. Moreover, their study has led to interesting models and applications of sublinear time verification, including non-interactive proofs of proximity (or MAPs) [32] (a related model

was studied concurrently and independently by [17]), arguments of proximity [36], testing properties of distributions [14, 35], interactive oracle proofs of proximity [40, 4, 42, 10], verifying machine learning tasks [29], batch verification for UP [39, 43], as well as variants involving zero-knowledge [6] and quantum computation [15].

1.2 Our Results

Our main contribution is constructing distribution-free IPPs for any language in NC, which for any query vs communication trade-off parameter $\tau \leq \sqrt{n}$, matches the complexity of the best known IPPs for most settings of the proximity parameter ε – specifically, when $\varepsilon \geq \tau/n$. We further improve the efficiency of distribution-free IPPs for general ε (i.e., when $\varepsilon < \tau/n$), under specific distribution families such as “smooth” and “learnable” distributions, which are defined below.

In addition, for certain families of languages, such as symmetric and relaxed self-correctable languages, we construct distribution-free IPPs that improve on our general-purpose distribution-free IPPs, then use them to provide separation results that provide further insight into the distribution-free IPP model.

We elaborate on these results next.

1.2.1 Distribution-free IPPs for NC

Our first main result is a sublinear distribution-free IPP for any language computable by low-depth circuits. In more detail, let (logspace-uniform) NC be the set of languages computable by (logspace-uniform) Boolean circuits of polynomial size and poly-logarithmic depth. We show that every language in NC has a distribution-free IPP with sublinear complexity measures, for almost all values of the proximity parameter ε . We emphasize that this is in stark contrast to distribution-free testers, which are only known for a handful of languages based on their combinatorial or algebraic structure. Indeed, the following theorem shows that distribution-free IPPs capture a much richer class of languages that need not have such special structural properties.

► **Theorem 1 (Distribution-Free IPP for NC).** *For every language L in logspace-uniform NC and every trade-off parameter $\tau = \tau(n) \leq \sqrt{n}$, there exists a distribution-free IPP for L with proximity parameter $\varepsilon \geq \Omega\left(\frac{\log^3(n)}{n}\right)$, query complexity $\tau + O\left(\frac{1}{\varepsilon}\right)$, sample complexity $\tau + O\left(\frac{1}{\varepsilon}\right)$ and communication complexity $\tilde{O}\left(\frac{n}{\tau} + \frac{1}{\varepsilon}\right)$.*

Moreover, the verifier runs in time $\tilde{O}\left(\frac{n}{\tau} + \frac{1}{\varepsilon}\right)$, the prover runs in time $\text{poly}(n)$ and the round complexity is $\text{polylog}(n)$.

Here, τ denotes the parameter that trades-off between the query and communication complexities of the distribution-free IPP. Note that, for the above values of τ , our distribution-free IPP has sublinear query and communication complexity even for very small values of the proximity parameter ε of the form $1/n^{1-\delta}$, where $\delta > 0$. An interesting instantiation of our result is obtained by setting τ to \sqrt{n} , and thus, for every $\varepsilon \geq 1/\sqrt{n}$, the query complexity and sample complexities are $O(\sqrt{n})$, while the communication complexity and verifier running times are both $\tilde{O}(\sqrt{n})$.

It is worth noting that, for every $\varepsilon \geq \frac{1}{\tau}$ (and $\tau \leq \sqrt{n}$), this result is conditionally optimal up to poly-logarithmic factors, since [36] show a lower bound of $\Omega(n)$ on the product of the query and communication complexities of a uniform IPP for a language in NC¹, under a strong, but reasonable, cryptographic assumption. Furthermore, for any ε , the query complexity of $\Omega(1/\varepsilon)$ is necessary for any IPP over non-degenerate languages, even over the uniform distribution (see [44, Remark 1.2]).

► **Remark 2.** While Theorem 1 refers to distribution-free IPPs over NC languages, the theorem can be made more general. In particular, it also yields distribution-free IPPs with sublinear query and communication complexities for languages computable by circuits of sub-exponential size and bounded polynomial depth.

Likewise, in a similar fashion to the known literature on uniform IPPs, we can combine our techniques directly with [40] to get a constant-round distribution-free IPP for any language that is computable in $\text{poly}(n)$ time and bounded polynomial space.

Comparison to Uniform IPPs for NC [44, 43]

For any language in NC, Rothblum, Vadhan and Wigderson [44] construct a uniform IPP for any $\tau = \tau(n)$ and proximity parameter $\varepsilon > 0$, with query complexity $\tau + O(1/\varepsilon)^{1+o(1)}$ and communication complexity $\frac{n}{\tau^{1-o(1)}}$. Rothblum and Rothblum [43] improve on this, by reducing the communication complexity to $\frac{n}{\tau} \cdot \text{polylog}(n)$. In particular, the latter obtains an optimal trade-off, up to poly-logarithmic factors, between the query and communication complexities of a uniform IPP (conditionally, from [36]), for every value of τ and $\varepsilon \geq 1/\tau$. While these results are stated in [44, 43] by implicitly setting $\tau = O(1/\varepsilon)$, for any given ε , this IPP formulation parameterised by τ is obtained by inspection (see also [26, Theorem 6.3]). For comparison, in this setting, our distribution-free IPP has the same query (and sample) complexity, while the communication complexity and verifier running times are both $\tilde{O}(\varepsilon \cdot n + 1/\varepsilon)$.³

Theorem 1 gives a construction of a *distribution-free* IPP for any NC language that matches the query and communication complexities of the uniform IPP by [43], when $\varepsilon \geq \tau/n$. Moreover, this obtains the (conditionally) optimal trade-offs between query and communication complexities in the *same regime* of ε , but when $\tau \leq \sqrt{n}$. Indeed, when $\varepsilon \geq 1/\tau$, the product of the query and communication complexities of the distribution-free IPP from Theorem 1 is $\tilde{O}(n + \tau^2)$. Our protocol builds on [44], introducing new ideas that allow us to construct IPPs in the more involved distribution-free setting.

Finally, when the proximity parameter ε is very small, Theorem 1 suffers a blow-up in the communication complexity compared to the uniform IPPs of [44, 43]. In more detail, when $\varepsilon \ll \tau/n$, the communication complexity in our distribution-free IPP is $\tilde{\Omega}(\frac{1}{\varepsilon})$, whereas the communication complexity achieved by the uniform IPPs is $\tilde{O}(\frac{n}{\tau})$ (the query complexity roughly remains the same across all three cases). Thus, our distribution-free IPP has communication complexity at least $\Omega(n/\tau)$ for every value of ε , whereas the communication complexity of the uniform IPPs is much lower when $\varepsilon \ll \tau/n$.

1.2.2 IPPs for NC: The case of small ε

Following the discussion in the last section, we aim to construct distribution-free IPPs that achieve query and communication complexities that match the state-of-the-art uniform IPP for every value of ε . While we are unable to do so in the most general case, we construct such IPPs over *specific families of distributions*, which match the complexities of [44] and, in turn, differ from the complexities of [43] only by a factor of $n^{o(1)}$. For these IPPs, while the underlying distribution is still unknown, it is guaranteed to come from the specific family of distributions under consideration.

³ In fact, we prove that for every value of the parameter τ and ε , the distribution-free IPP from Theorem 1 has communication complexity $\tilde{O}(\tau + n/\tau + 1/\varepsilon)$; thus, setting $\tau = O(1/\varepsilon)$ suffices. An additional point to note is that when $\tau > \sqrt{n}$, the IPP always has worse communication complexity than its uniform counterpart irrespective of the value of ε , and further, never meets the optimal [36] lower bound. As such, we only consider $\tau \leq \sqrt{n}$ as a more interesting regime of study.

To describe our results, it will be convenient throughout this section to identify $[n]$ with the elements of an m -dimensional tensor of size $k \in \mathcal{N}$ in each dimension, such that $k^m = n$. In such a case, we refer to $[n]$ as $[k]^m$ (by fixing some canonical bijection between them).

ρ -Dispersed Distributions

Intuitively speaking, ρ -dispersed distributions capture the sense that for a smooth distribution over $[k]^m$, along any dimension, its probability mass on any element in $[k]^m$ is not much larger than the average of the probability masses of its neighbours. ρ -dispersed distributions relax this requirement by having the probability mass on any element bounded by ρ times the expected mass on any of its neighbours.⁴

We show that for distributions that are reasonably smooth in this sense, i.e. for ρ -dispersed distributions for $\rho \leq k^{o(1)}$, we obtain IPPs for NC over such distributions for every $\tau = \tau(n) < n$ and $\varepsilon > 0$, with query complexity $O(\tau + 1/\varepsilon)^{1+o(1)}$, and communication complexity of $\tilde{O}(\frac{n}{\tau} \cdot \tau^{o(1)})$, thus matching the bounds obtained by [44]. It is worth noting that $k^{o(1)}$ -dispersed distributions are still quite general, e.g. any distribution where the probability mass on any element in $[k]^m$ is in the range $[\frac{1}{an}, \frac{a}{n}]$, for some $a \leq k^{o(1)}$ is $k^{o(1)}$ -dispersed.

► **Theorem 3 (IPP for NC over ρ -dispersed distributions).** *For every language in logspace-uniform NC, every $m, n, k \in \mathcal{N}$ such that $m = \log_k(n)$ (i.e., $k^m = n$) and $\rho \in \mathcal{R}$ such that $\rho \leq k^{o(1)}$, for every proximity parameter $\varepsilon > 0$ and trade-off parameter $\tau > 0$, there exists an IPP over ρ -Dispersed distributions over $[k]^m$ with query and sample complexities $O(\tau + 1/\varepsilon)^{1+o(1)}$ and communication complexity $\tilde{O}(\frac{n}{\tau^{1-o(1)}})$.*

Moreover, the verifier runs in time $n^{o(1)} \cdot (\tau + \frac{n}{\tau} + \frac{1}{\varepsilon})$, the prover runs in time $\text{poly}(n)$ and the round complexity is $\text{polylog}(n)$.

Theorem 3 also holds generally over ρ -dispersed distributions, for any ρ . The query complexity increases with ρ , while the communication complexity is *independent* of ρ . Theorem 3 builds on the ideas used for the distribution-free IPP from Theorem 1 while incorporating new technical insights into the analysis by [44] to generalise over ρ -dispersed distributions. We leave the task of obtaining IPPs over ρ -dispersed distributions that match [43] as future work.

1.2.2.1 Product Distributions in the White-Box model

Note that in the IPPs of Theorems 1 and 3, the verifier does not learn the underlying distribution \mathcal{D} . Hence, we ask the following question: if we could gain more information about \mathcal{D} , or further, learn a reasonably good approximation for \mathcal{D} , can we improve the query complexity of the IPPs, over general values of ε ? We answer this question in the affirmative for product distributions.

We consider the *white-box model* for distribution-free IPPs, where the verifier receives a succinct description of the unknown distribution \mathcal{D} over $[k]^m$ via a *polynomial-sized* sampling circuit C , in addition to query access to the input string. It is worth noting that, for white-box IPPs, the sample complexity is irrelevant since the verifier has a succinct description of the entire distribution. Thus, the main complexity parameters here are the query complexity, communication complexity, and the verifier running time.

⁴ For example, the uniform distribution is the only 1-dispersed distribution, i.e., a maximally smooth distribution in this sense. On the other hand, every distribution over $[k]^m$ is trivially a k -dispersed distribution.

While white-box models have been widely studied in the setting of zero-knowledge proofs [46, 48, 47] and in distribution testing (see survey by [27]), we use this model to construct IPPs for languages in NC over a generalised family of product distributions over $[k]^m$, to get improved complexities for general values of ε , compared to the distribution-free IPP from Theorem 1. We call this family as *m-product distributions*, and denote any such distribution \mathcal{D} as $\mathcal{D} = \mathcal{D}_1 \times \dots \times \mathcal{D}_m$, where each \mathcal{D}_j is supported on $[k]$ and is independent of any other coordinate distributions. In particular, $\mathcal{D}(i_1, \dots, i_m)$ is defined as $\prod_{j=1}^m \mathcal{D}_j(i_j)$.

► **Theorem 4 (IPPs for NC over m-product distributions).** *For every language in logspace-uniform NC, every $\tau = \tau(n)$, $\varepsilon > 0$, and $m, n, k \in \mathcal{N}$ such that $m \leq \log(n)$ and $k^m = n$, there exists a white-box IPP for L over m-product distributions over $[k]^m$. The IPP has query complexity $O(\tau + 1/\varepsilon)^{1+o(1)}$ and communication complexity $(\frac{n}{\tau^{1-o(1)}} \cdot k + k^2) \cdot \text{polylog}(n)$. Moreover, the verifier runs in time $n^{o(1)} (\frac{n}{\tau} \cdot k + \tau + k^2 + \frac{1}{\varepsilon})$ and the round complexity is $\text{polylog}(n)$.*

When m is large enough (like $m = \log(n)$), the query and communication complexity trade-off, as well as the verifier running time of the IPP from Theorem 4 match that of the uniform IPP from [44], while working in this setting.⁵ Theorem 4 builds on the framework of Theorem 1, and uses several new ideas in the construction of the IPP, as well as its analysis, to improve the complexity. Crucially, it uses that any product distribution has a succinct description to be able to *learn* it in the white-box-setting.

It is worth stressing that the IPPs from Theorems 3 and 4 are incomparable. Indeed, there exist m -product distributions $\mathcal{D} = \mathcal{D}_1 \times \dots \times \mathcal{D}_m$ that are poorly dispersed, for eg., \mathcal{D} is no longer smooth when some \mathcal{D}_j has a large probability mass over just one element (one row or more generally, a few rows). For such distributions, the IPP from Theorem 4 provides a much better query and communication trade-off than the IPP from Theorem 3, which is a more general result for smooth distributions.

1.2.3 On the power of distribution-free IPPs

Recall that Theorems 3 and 4 improve the query and communication complexity trade-off of our general distribution-free IPP in Theorem 1, by considering special families of distributions to design the IPPs over. A natural direction that complements this approach is to ask whether we can use additional information about the *language* L instead, to construct super-efficient distribution-free IPPs.

In turn, we study distribution-free IPPs for specific problems of interest. On one hand, for certain problems we can hope to improve the various associated complexity parameters over our general distribution-free IPP by capitalising on the structure of the language. On the other hand, this allows us to obtain complexity-theoretic separations between the power of standard, non-interactive, and interactive distribution-free testers.

1.2.3.1 Symmetric languages

We study the power of distribution-free testers and IPPs for symmetric languages, which are languages that are invariant under permutations. We show that there exist symmetric languages that are hard for distribution-free testers, yet, given interaction with a prover, the symmetrical structure can be leveraged to obtain exponentially faster distribution-free IPPs.

⁵ A subtle point here is that while Theorem 4 is over product distributions over $[k]^m$, when $m = 2$ (or a small constant), we get sublinear complexities only by considering distributions over biased matrices $[k_1] \times [k_2]$.

► **Theorem 5 (Distribution-free IPPs for symmetric languages).** *The following statements hold.*

1. *Let L be a symmetric language. Then, there exist a distribution-free IPP for L with sample complexity $O(1/\varepsilon)$, communication complexity $O(\log^2(n)/\varepsilon)$ and $O(\log(n)/\varepsilon)$ round complexity.*
2. *There exists a symmetric language L' for every $\varepsilon > 0$ such that any distribution-free property tester for L' requires $\Omega(n^{1/3-0.0005})$ queries and labeled samples from the input.*

1.2.3.2 (Relaxed) self-correctable languages

Next, we show that for languages that admit self-correctability, we can transform any IPP into a distribution-free IPP at a negligible cost. In fact, we can deal with a far more general class of languages; namely, languages that are *relaxed locally correctable* [5, 31]. Loosely speaking, these are languages that admit a correcting algorithm that is required to correct the symbol at every location of the codeword, by reading a small number of locations in it, but is allowed to abort if noticing that the given word is corrupted. This family of languages is of central importance in the interactive proofs and probabilistically checkable proofs literature, and in particular, it captures languages of low-degree polynomials, holographic IPPs, and various relaxed locally correctable and decodable languages that were used to prove complexity-theoretic separations (cf. [30]).

► **Proposition 6 (Generic Transformations for IPPs for RLCCs).** *For any subset L of a binary RLCC, $C \subseteq \{0,1\}^n$, if L has an IPP over the uniform distribution with query complexity q and communication complexity c for proximity $\varepsilon > 0$, then there exists a distribution-free IPP for L with the same round complexity, communication complexity and query complexity $q + O(\frac{t}{\varepsilon})$, where t is the query complexity of the corrector of C .*

As a corollary of Proposition 6, we are able to lift complexity-theoretic results concerning uniform IPPs to the setting of distribution-free IPPs. In particular, we obtain strong separations between the power of distribution-free testers, distribution-free non-interactive proofs of proximity (MAPs), and distribution-free IPPs.

► **Corollary 7 (Complexity separations).** *There exists a language L such the following hold true.*

1. *Property Testing: The query complexity of distribution-free testing L (without a proof) is $\Theta(n^{0.999 \pm o(1)})$.*
2. *MAP: L has a distribution-free MAP with query and communication complexities $\Theta(n^{0.499 \pm o(1)})$. Moreover, for every $p \geq 1$, the distribution-free MAP query complexity of L with respect to proofs of length p is $\Theta(\frac{n^{0.999 \pm o(1)}}{p})$.*
3. *IPP: L has a distribution-free IPP with query and communication complexities $\text{polylog}(n)$.*

Complementing this Corollary, we prove the existence of languages that can be tested under the uniform distribution with low query complexity (and thus, have a uniform IPP with low query complexity and no communication), but for which distribution-free IPPs require large query complexity or large communication complexity. This illustrates the difficulty of constructing distribution-free IPPs vs. standard uniform IPPs.

► **Proposition 8 (Distribution-free IPPs vs. uniform testing).** *The following hold true:*

1. *There exists $\varepsilon > 0$ and a language L such that L has a property tester over the uniform distribution with query complexity $O(1/\varepsilon)$ for proximity parameter ε . However, for any distribution-free MAP for L with proximity parameter ε , query complexity q , and proof length p , $\max(q, p) = \Omega(\varepsilon \cdot n)$.*

2. Assuming the existence of exponentially hard pseudo-random generators, there exists $\varepsilon > 0$ such that for all $q = q(n) \leq n$, there exists a language L , such that for any distribution-free IPP for L with proximity parameter ε , communication complexity c , and query complexity q , $\max(c, q) = \Omega(\sqrt{\varepsilon \cdot n})$. However, L has a uniform property tester with query complexity $O(1/\varepsilon)$ for proximity parameter ε .

Table 1 provides a comparison of some of these results with related testing models. It is an interesting open direction to exhibit distribution-free IPPs that improve on the query complexity lower bounds known for distribution-testing functional properties like monotonicity [33], monotone conjunctions [13], or k -juntas [38].

■ **Table 1** This is a table of our main results (TensorSum as defined in [32]). The complexities shown here are those that minimise the sum of the query and communication complexity. Note that while the uniform property tester for symmetric properties is more efficient than the corresponding uniform IPP, this only holds for restricted (constant) values of ε .

	Property Testing	IPP	DF-Property Testing	DF-IPP
Languages in NC	$\Omega(n)$ (e.g., low-degree univariate polynomial)	$\tilde{O}(\sqrt{n})$ [44, 43]	$\Omega(n)$ similarly	$\tilde{O}(\sqrt{n})$ (arbitrary distributions, for $\varepsilon \geq 1/\sqrt{n}$); see Theorem 1 $n^{1/2+o(1)}$ (smooth distributions); see Theorem 3 $n^{1/2+o(1)}$ (product distributions); see Theorem 4
TensorSum	$\Omega(n^{0.99+o(1)})$ [32]	$\text{polylog}(n)$ [32]	$\Omega(n^{0.99+o(1)})$ Trivially, from [32]	$\text{polylog}(n)$; see Corollary 7
Symmetric Properties	$\Theta(1)$ ($\varepsilon = O(1)$) Folklore	$\text{polylog}(n)$ [44]	$\Omega(n^{\frac{1}{3}})$ Theorem 5	$\text{polylog}(n)$; see Theorem 5

1.3 Technical Overview

In this technical overview, we highlight the proofs of Theorems 1, 3, and 4. The general strategy for proving these theorems builds on the Uniform IPPs for NC from [44, 43]. However, the setting of distribution-free testing is more involved, and below, we highlight the key challenges encountered in this setting, and our ideas to overcome them. Our distribution-free IPPs are constructed through an interplay of various techniques and tools from interactive proofs, property testing, and distribution testing.

Note that, for convenience, we show the construction of the distribution-free IPP from Theorem 1 in the setting of $\tau = O(1/\varepsilon)$, for any proximity parameter ε , obtaining query complexity $O(1/\varepsilon)$ and communication complexity $\tilde{O}(\varepsilon \cdot n + 1/\varepsilon)$. This can be shown to be equivalent to the statement of Theorem 1 that is parameterised by τ . Similarly, the IPPs for our other results are parameterised in terms of the proximity parameter ε . For detailed proofs, we refer the reader to the full version [1].

1.3.1 Proof outline of Theorem 1

The [44] protocol (as well as the follow-up work [43]) is centered around a parameterised problem called PVAL. Loosely speaking, the PVAL language contains all strings, whose encoding under a specific code, called the low degree extension, is equal to given values when projected on to the given coordinates. More precisely, the PVAL problem is parameterised by

24:10 Distribution-Free Proofs of Proximity

a (sufficiently large) finite field \mathcal{F} , integers k, m, n such that $k, m < |\mathcal{F}|$ and $k^m = n$, a set of vectors $J = (j_1, \dots, j_t) \subset \mathcal{F}^m$ of size t and a t -length vector $\vec{v} \in \mathcal{F}^t$. An input $X \in \mathcal{F}^{k^m}$ is in $\text{PVAL}(J, \vec{v})$ if it holds that $P_X(j_i) = v_i$, for every $i \in [t]$, where $P_X : \mathcal{F}^m \rightarrow \mathcal{F}$ is the m -variate low-degree extension (LDE) of X .⁶

The interactive reduction from NC to PVAL

Let L be any language in NC and let $\varepsilon > 0$ be the input proximity parameter. Let $X \in \{0, 1\}^n$ be the input to L and \mathcal{D} be the unknown underlying distribution over which the verifier can access X through a sample oracle. The first step in [44] is to show an interactive reduction Π_{NC} from L to (a parameterisation of) PVAL, where the verifier *does not access* the input $X \in \{0, 1\}^n$.⁷

In more detail, let $B_{\mathcal{D}}(X)$ (respectively $B_{\mathcal{U}}(X)$) be the set of binary strings that are at a distance at most ε along the distribution \mathcal{D} (respectively the uniform distribution \mathcal{U}) from X . In [44], the verifier in Π_{NC} generates parameters $(\mathcal{F}, k, m, J, \vec{v})$ for PVAL, where J is a set of t points in \mathcal{F}^m , such that the following hold when t is sufficiently large.

- If $X \in L$, then $X \in \text{PVAL}(J, \vec{v})$.
- If X is ε -far from L along \mathcal{U} then, with high probability over the verifier's randomness, $B_{\mathcal{U}}(X)$ and $\text{PVAL}(J, \vec{v})$ are disjoint. In other words, with high probability, X is ε -far from $\text{PVAL}(J, \vec{v})$ along \mathcal{U} .

Furthermore, the points J output by the reduction Π_{NC} are *distributed uniformly at random* in $(\mathcal{F}^m)^t$. Crucially, [44] show that the guarantees over the outputs of this reduction *only hold* when $t = O(\log(|B_{\mathcal{U}}(X)|))$ many points are picked in J .⁸

Since the size of the set $B_{\mathcal{U}}(X)$ is $\binom{n}{\varepsilon n} \leq O(2^{\varepsilon n \log(n)})$, following from the earlier discussion, by setting $t = O(\log(|B_{\mathcal{U}}(X)|)) = \tilde{O}(\varepsilon n)$, we ensure that the guarantees of Π_{NC} hold. An immediate attempt would be to try to extend this analysis verbatim to distribution-free testing, by setting t to $O(\log(|B_{\mathcal{D}}(X)|))$ instead, and thus having Π_{NC} guarantee that X is ε -far from $\text{PVAL}(J, \vec{v})$ along the distribution \mathcal{D} , for soundness. However, for an arbitrary unknown distribution \mathcal{D} , the size of $B_{\mathcal{D}}(X)$ can be prohibitively large. For example, when \mathcal{D} is supported over the first $\log(n)$ indices, for any value of ε , the size of $B_{\mathcal{D}}(X)$ blows up to at least $2^{n - \log(n)}$. Thus, for our choice of t , we already lose the sublinear time verification and communication complexity, and it is unclear if this reduction can achieve such soundness guarantees for PVAL.

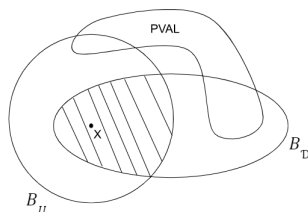
Uniform IPP for PVAL is also “complete” for distribution-free IPPs for NC

Our key idea for constructing the distribution-free IPP for L , is in fact, an interactive reduction Π' to constructing a *uniform* IPP for PVAL (with a different parameterisation for PVAL than that obtained by Π_{NC}). Theorem 1 follows by using the ready-made uniform IPP for PVAL by [43].

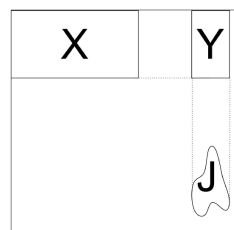
⁶ Recall that the m -variate LDE P_X is the unique polynomial with individual degree $k - 1$ such that P_X agrees with X on $[k]^m$, where we identify $[k]$ with a subset of field elements in some canonical way.

⁷ Technically, an interactive proof is specified by a verifier and an honest prover. However, for the sake of exposition we refer to them both together as Π_{NC} in this section.

⁸ Π_{NC} runs t parallel copies of the interactive reduction from L to PVAL over a single point by [28], with the guarantee that if the input $X \notin L$, the probability that X is also in PVAL over t points, is at most 2^{-t} . Now, if X were instead ε -far from L , then a union bound over all the points in $B_{\mathcal{U}}(X)$ ensures a small probability for the event that there exists a point in $B_{\mathcal{U}}(X)$ that is also in PVAL over t points.



■ **Figure 1** The shaded region $(B_U(X) \cap B_D(X))$ consists of the set of points in $\{0, 1\}^n$ that are ε -close to X with respect to both \mathcal{D} and \mathcal{U} . The soundness promise of the interactive reduction Π' ensures that any string in $\text{PVAL}(J, \vec{v})$ is present in at most one of $B_U(X)$ or $B_D(X)$, but not in both (shaded region) (with high probability).



■ **Figure 2** In the uniform IPP for PVAL, the prover sends the $(m - 1)$ -variate LDE of each row of X evaluated on J_2 (column indices of J), in the form of the purported matrix $Y' \in \mathcal{F}^{k \times t}$. However, to ensure consistency of Y' with respect to $\text{PVAL}(J, \vec{v})$, for any $j = (a, b) \in J$, the univariate LDE of the b^{th} -column of Y' evaluated on a is required to be equal to $\vec{v}[j]$.

Consider a NO input $X \in \{0, 1\}^n$ to L , that is, an input that satisfies the soundness requirement $d_{\mathcal{D}}(X, L) > \varepsilon$, over the unknown distribution \mathcal{D} . To start with, Π' runs the interactive reduction Π_{NC} from L to $\text{PVAL}(J, \vec{v})$ with the same value of $t = |J| = \tilde{O}(\varepsilon n)$.

Setting t to be $O(\log(|B_D(X) \cap B_U(X)|)) \leq O(\log(|B_U(X)|)) = \tilde{O}(\varepsilon n)$, we can generalise the guarantees of Π_{NC} to show that the intersection of $B_U(X)$ and $B_D(X)$ is disjoint from $\text{PVAL}(J, \vec{v})$, with high probability. Indeed, this builds on the earlier argument (and Footnote 8), but over $B_U(X) \cap B_D(X)$, alongside the fact that the size of this set is upper bounded by the size of $B_U(X)$. Thus, X cannot be ε -close to $\text{PVAL}(J, \vec{v})$ along *both* \mathcal{U} and \mathcal{D} , or in other words, X is ε -far from every element of PVAL along at least one of the two distributions (see Figure 1 for details).

Following this, assume that $d_{\mathcal{D}}(X, \text{PVAL}(J, \vec{v})) > \varepsilon$. We construct the next stage of Π' , based on a case analysis whether X is *additionally* ε -far from $\text{PVAL}(J, \vec{v})$ under the uniform distribution or not. Indeed, suppose that X is ε -far from $\text{PVAL}(J, \vec{v})$ under the uniform distribution. This is the easy case; we can catch this with the uniform IPP for $\text{PVAL}(J, \vec{v})$ as usual.

On the other hand, suppose that instead, X is close to $\text{PVAL}(J, \vec{v})$ under the uniform distribution, i.e., $d_{\mathcal{U}}(X, \text{PVAL}(J, \vec{v})) \leq \varepsilon$. At this point, we observe (following [43]) that when J is distributed uniformly at random, with high probability $\text{PVAL}(J, \vec{v})$ is a good error correcting code (i.e., with large minimal distance).⁹ Since the output J of Π_{NC} is distributed uniformly at random, when X is ε -close to $\text{PVAL}(J, \vec{v})$ over the uniform distribution, Π_{NC} guarantees that X is in fact close to a *unique* element $X' \in \text{PVAL}(J, \vec{v})$.

To summarize, so far we have that X is ε -close to $X' \in \text{PVAL}(J, \vec{v})$ along \mathcal{U} , but by our soundness condition, X is ε -far from $\text{PVAL}(J, \vec{v})$, and in particular from X' , along \mathcal{D} . Now, the verifier uses the sample oracle to \mathcal{D} to generate $O(1/\varepsilon)$ samples, which we denote by $I \subseteq [n]$, and the corresponding values in X given by $X|_I$. From the soundness assumption, with high probability there exists an index i in I such that $X_i \neq X'_i$. Combining this with the fact that every element in $\text{PVAL}(J, \vec{v})$ other than X' is ε -far from X along the uniform distribution, X' is not in $\text{PVAL}((J, I), (\vec{v}, X|_I))$, where PVAL is parameterised over a larger set. In other words, we see that X is ε -far from $\text{PVAL}((J, I), (\vec{v}, X|_I))$ along the *uniform distribution* and a uniform IPP for $\text{PVAL}((J, I), (\vec{v}, X|_I))$ suffices.

⁹ It is worth emphasising that this does not hold for every choice of J , for eg., $\text{PVAL}(J, \vec{v})$ is a bad error correcting code when J consists of t copies of the same point.

The argument for completeness trivially holds from the guarantees of Π_{NC} and definition of an LDE of X , since in this case $X \in \text{PVAL}((J, I), (\vec{v}, X|_I))$. We end with a quick note on the complexity of the distribution-free IPP. The query complexity of $O(1/\varepsilon)$ is the same as that of the uniform IPP by [43], and the communication complexity is the sum of the number of bits used to send the $O(1/\varepsilon)$ samples in I in addition to the communication by the uniform IPP, which is $\tilde{O}(\varepsilon n)$. Overall the communication complexity is $\tilde{O}(\frac{1}{\varepsilon} + \varepsilon \cdot n)$ which matches that in [43] (up to poly-logarithmic factors) whenever $\varepsilon \geq 1/\sqrt{n}$.

1.3.2 Proof outlines of Theorems 3 and 4

Next, we describe the proof techniques of Theorems 3 and 4 that construct IPPs for NC over smooth distributions and product distributions, matching the complexities of [44] for every value of ε . This improves over the communication complexity of the distribution-free IPP in Theorem 1 when $\varepsilon \ll 1/\sqrt{n}$ (with roughly the same query complexity). We follow the general strategy by [44] and the main technical challenges arise during the analysis with respect to the new promise on the soundness of an IPP for PVAL. We assume some familiarity with the uniform IPP construction by [44] for this section.

Uniform IPP for PVAL(J, \vec{v})

We start with a summary of the Uniform IPP from [44]. Let the input $X \in [k]^m$, for $k = \log n$ and $n = k^m$. Further, let $|J| = t$.

[44] use a divide and conquer approach, by decomposing the t claims about X into new claims for each individual row instance $X_i \in \mathcal{F}^{k^{m-1}}$, for every $i \in [k]$. In more detail, let $J = (J_1, J_2)$, where the first component $J_1 \subset \mathcal{F}$ and $J_2 \subset \mathcal{F}^{m-1}$. The prover sends the matrix $Y' \in \mathcal{F}^{k \times t}$, where each row Y'_i is the purported set of evaluations of the $(m-1)$ -variate LDE (of individual degree $k-1$) of X_i on J_2 . By the definition of an m -variate LDE on X , the prover cannot lie about the consistency of Y' with \vec{v} , since for each $(a, b) \in J$ (where $b \in J_2$), the verifier can easily check if the univariate LDE of $Y'[\cdot, b]$ (the b^{th} column of Y) evaluated on the coordinate a equals $\vec{v}[(a, b)]$ (see Figure 2).

Thus, the initial PVAL instance is now reduced to k instances $X_i \in \mathcal{F}^{k^{m-1}}$ for $\{\text{PVAL}(J_2, Y'_i)\}$. A natural idea now is for the verifier to send a random vector $z \in \mathcal{F}^k$ to the prover, and ask it back for a “folded” version $X' \in \mathcal{F}^{k^{m-1}}$, that is purported to be $z \cdot X$.¹⁰ Now, the IPP could recurse on a *single input* $X' \in \mathcal{F}^{k^{m-1}}$ that has shrunk in size by a factor of k , to the problem $\text{PVAL}(J_2, z \cdot Y')$. Completeness easily holds, since if X belonged to $\text{PVAL}(J, \vec{v})$, then the honest prover will just send the “true” $Y' \in \mathcal{F}^{k \times t}$ and the verifier checks always pass.

Uniform Distance Preservation Lemma

However showing soundness is not straightforward. Suppose that X is ε -far from $\text{PVAL}(J, \vec{v})$ under the uniform distribution. It turns out that the malicious prover has cheated in at least one row of the purported matrix Y' (if not, since X is not in PVAL, there would be at least one column in Y' which would be inconsistent with the corresponding value in \vec{v} and the verifier would catch the prover in the checks made above).

¹⁰The dot product $z \cdot X \in \mathcal{F}^{k^{m-1}}$ between $z \in \mathcal{F}^k$ and a matrix $X \in \mathcal{F}^{k \times k^{m-1}}$ is given by $\sum_{i=1}^k z_i X_i$.

For any row $X_i \in \mathcal{F}^{k^{m-1}}$ that is a lower-dimensional input instance, let ε_i be the distance between X_i and $\text{PVAL}(J_2, Y'_i)$. To ensure that the verifier catches the cheating prover, the folded instance X' also needs to be reasonably far from PVAL on a lower dimension at the end of a recursive step. In order to capture this, [44] (implicitly) use a *uniform distance preservation lemma*, which states that if X is ε -far from $\text{PVAL}(J, \vec{v})$, then $\sum_{i=1}^k \varepsilon_i > k\varepsilon$.

Using the uniform distance preservation lemma, [44] observe that if the prover ended up cheating (roughly) uniformly across all rows in Y' , then any row X_i would be roughly ε -far from $\text{PVAL}(J_2, z \cdot Y'_i)$, and the IPP would recurse by picking a single row at random. However, the prover could have cheated across multiple rows of Y'_i and the verifier does not know these rows. To accommodate this, the verifier considers $\log(k)$ many random foldings of X , where the Hamming weight of the vectors z used to fold X , range across 1 to k (in powers of 2). In particular, this results in $O(\log(\log(n)))$ recursive instances in $\mathcal{F}^{k^{m-1}}$. Crucially, they use the uniform distance preservation lemma to generalise the intuition above and show that for at least one of these folded instances, the distance is roughly preserved. Moreover, for such a folded instance, the product of the new distance and the effective query complexity (the number of queries on X to compute the value at any index in $z \cdot X$) is $O(1/\varepsilon)$, along with small but super-constant multiplicative factors.

The IPP continues to recursively fold the instance dimension-wise by the above process, until the size of each final folded instance becomes $\tilde{O}(\varepsilon n)$, which happens after $\Omega(\log(n)/\log(\log(n)))$ steps. In such a case, the prover directly sends each final instance. Since there exists an instance \tilde{X}^j at each level of recursion for which distance is preserved, there exists a final folded instance \tilde{X} , such that the verifier catches a cheating prover by uniformly *sampling* a few coordinates of \tilde{X} . Moreover, since the product of the distance and effective query complexities for each \tilde{X}^j are roughly maintained to be small at each step of the recursion, making $O(1/\varepsilon^{1+o(1)})$ many queries to \tilde{X} is sufficient to catch the cheating prover (since the total number of recursive instances after the stated number of steps is roughly $n^{o(1)} = 1/\varepsilon^{o(1)}$). The communication complexity is simply the number of bits used to send all the final folded instances, in addition to sending the matrices Y' of size $k \times t$, and thus is $\tilde{O}(\varepsilon^{1-o(1)}n)$.

IPPs for NC under specific distribution families

We now highlight some key ideas which help us construct IPPs over large distribution families like smooth distributions and product distributions. To begin with, on any input $X \in \{0, 1\}^{k^m}$, we first reduce L to PVAL using Π_{NC} . Recall that in the distribution-free setting, Π_{NC} outputs (J, \vec{v}) , such that for the soundness promise, with high probability X cannot be ε -close to $\text{PVAL}(J, \vec{v})$ along both \mathcal{U} and the unknown distribution from the given family, \mathcal{D} . In other words, X is ε -far from $\text{PVAL}(J, \vec{v})$ along at least one of \mathcal{U} or \mathcal{D} . Building on this observation, we design IPPs for $\text{PVAL}(J, \vec{v})$ over these distribution families, using an intricate case analysis of the soundness condition.

In more detail, if X is ε -far from $\text{PVAL}(J, \vec{v})$ under the uniform distribution, then we can directly use the uniform distance preservation lemma to catch a malicious prover as seen previously in the uniform IPP. If not, suppose that $d_{\mathcal{D}}(X, \text{PVAL}(J, \vec{v})) > \varepsilon$. Next, we briefly describe the soundness analysis, using *specific distance preservation lemmas* for smooth distributions and product distributions. Given this, we build on the strategy of the uniform IPP above to construct an IPP for $\text{PVAL}(J, \vec{v})$ over these distribution families, with the main technical work being that of simultaneously incorporating both the uniform and the respective distance preservation lemmas into the soundness analysis, across the recursive levels.

ρ -dispersed distributions

Recall that ρ -dispersed distributions over $[k]^m$ capture the smoothness of a distribution, by requiring that the probability mass on any element is bounded by ρ times the average mass on any of its neighbours. Adopting similar notation as above, let $\hat{\mathcal{D}}$ be the marginal distribution of \mathcal{D} over $[k]^{m-1}$.

For any row $X_i \in \mathcal{F}^{k^{m-1}}$ that is a lower-dimensional input instance, let ε_i be the distance between X_i and $\text{PVAL}(J_2, Y'_i)$ over $\hat{\mathcal{D}}$. Here, we show a distance preservation lemma for ρ -dispersed distributions, such that for any distribution \mathcal{D} that is ρ -dispersed, $\sum_{i=1}^k \varepsilon_i > (k\varepsilon)/\rho$.¹¹ The idea behind proving this is not obvious immediately; while ε_i measures the distance along marginal distributions, ε is the distance from each element of $\text{PVAL}(J, \vec{v})$ over \mathcal{D} (which could be a joint distribution). However, we crucially use properties about ρ -dispersed distributions to prove this distance preservation lemma.

Using the strategy described earlier, we get an IPP for NC over ρ -dispersed distributions, having query and sample complexities $\frac{\rho^{\log(1/\varepsilon)/\log \log(n)}}{\varepsilon^{1+o(1)}}$, while keeping communication complexity the same. In particular, for $\rho = k^{o(1)}$, the query complexity is $1/\varepsilon^{1+o(1)}$ and matches that of the uniform IPP for all $\varepsilon > 0$.

Product distributions

Let \mathcal{D} be an m -product distribution defined as $\mathcal{D} = \mathcal{D}_1 \times \dots \times \mathcal{D}_m$ over $[k]^m$, where $k = \log(n)$, and each \mathcal{D}_j is an independent distribution supported on $[k]$. In particular, $\mathcal{D}(i_1, \dots, i_m)$ is defined as $\prod_{j=1}^m \mathcal{D}_j(i_j)$.

Our main approach here to construct IPPs over such distributions, is to first *learn* the underlying distribution and then use this as an aid to obtain near-optimal complexity parameters. For more context, consider the following k -dispersed distribution \mathcal{D} over $[k]^m$, that is supported on the first row of the first dimension, i.e. exactly on the set of elements of the form $(1, i_2, \dots, i_m)$ for every $(i_2, \dots, i_m) \in [k]^{m-1}$.¹² We see that the IPP over k -dispersed distributions has query complexity $O(1/\varepsilon^2)$. However, if the verifier “learns” beforehand that \mathcal{D} is only supported on the first row, then it can focus its attention on a smaller instance in $\mathcal{F}^{k^{m-1}}$ and potentially obtain much better query complexity, if \mathcal{D} conditioned on the first row is ρ -dispersed, for a small ρ .

Our main technical idea here is to show a *learning-augmented* distance preservation lemma for product distributions. Let ε_i be the distance between X_i and $\text{PVAL}(J_2, Y'_i)$ over $\hat{\mathcal{D}} = \mathcal{D}_2 \times \dots \times \mathcal{D}_m$. Based on an alternative analysis to that of ρ -dispersed distributions, we prove that for any product distribution \mathcal{D} , $\sum_{i=1}^k \varepsilon_i > C\varepsilon$, for $C > 1$ that *only depends on* \mathcal{D}_1 . Using this key insight, if the verifier “transformed” \mathcal{D}_1 into the uniform distribution over $[a_0 \cdot k]$, where $a_0 \geq 1$ is a small constant, then we get a similar expression as the uniform distance preservation lemma, i.e., $C = O(k)$, despite still measuring distance according to $\hat{\mathcal{D}}$ for the lower dimensional instances.¹³

We briefly highlight the sequence of tools used to implement the latter idea. The verifier learns the probability vector of \mathcal{D}_1 , into an approximation \mathcal{P}_1 , using the *parallel set lower bound protocol* [9] which requires white-box access to \mathcal{D}_1 . Following this, it runs a

¹¹Note that the uniform distribution is a 1-dispersed distribution and we thus generalise the uniform distance preservation lemma.

¹²Intuitively, for any $i_2, \dots, i_m \in [k]^{m-1}$, $\mathcal{D}(1, i_2, \dots, i_m)$ is the only element in the set $\{\ell, i_2, \dots, i_m\}_{\ell \in [k]}$ with a non-zero probability mass and thus is k -times the average of the probability mass on its neighbourhood.

¹³For consistency, $a_0 = 1$, when \mathcal{D}_1 is just \mathcal{U}_k .

“granularising” algorithm taking \mathcal{P}_1 as input, that outputs the probability vector of a new $8k$ -granular distribution \mathcal{E}_1 over $[k+1]$ (i.e., for every i , $\mathcal{E}_1(i)$ is $b_i/8k$), such that in the soundness case, the distance of the input over \mathcal{E}_1 is still ε (up to constant factors). Finally, this granularity set is used to “extend” X into a new input instance $X' \in \{0,1\}^{8k \times k^{m-1}}$, by making copies of each row according to its granularity, and we can thus, equivalently consider the underlying row distribution as the uniform distribution over $[8k]$. The last two steps build on ideas from [24] for testing unknown distributions, while our focus is on the setting of testing with an implicit input.

1.4 Related Work

Proofs of Proximity for Distributions

In a related model, [14, 35] study proofs of proximity for *testing distributions*. In their setting, for a fixed property Π of distributions, the verifier receives samples from an unknown distribution \mathcal{D} , and interacts with the prover to decide whether $\mathcal{D} \in \Pi$ or \mathcal{D} is ε -far from any distribution in Π along the total variation distance. While there are superficial similarities to our model regarding the use of sample oracle, we focus on testing properties (or languages) of strings, where the distribution oracle only provides a means of accessing the input string. In addition, the verifier also has oracle access to the input instance and the distance for the NO instance is measured with respect to the underlying distribution.

Sample-based IPPs

Another related model is that of Sample-based IPPs [20], where the verifier can *only* access the input through an oracle that provides labeled samples over the uniform distribution. They show that any language in logspace-uniform NC has an SIPP with $\tilde{O}(\sqrt{n})$ sample and communication complexities, by in fact constructing a reduction protocol from an SIPP to the query-based IPP by [44]. Our model is more general conceptually, since any protocol in our model needs to be able to test for a language given access to labeled samples over any unknown distribution. On the other hand, to aid with this generality, we also provide the verifier with the more powerful oracle access to the input, which SIPPs do not.

That being said, we can use the uniform SIPP by [20] within the proof of Theorem 1 (instead of the query-based IPP by [43]) to obtain a distribution-free SIPP for NC where the verifier only accesses the input through labeled samples over \mathcal{U} and the unknown distribution \mathcal{D} , for any $\varepsilon \geq \tau/n$.¹⁴ It is unclear whether we can construct distribution-free SIPPs for general values of ε (even over smooth or product distributions) that match the complexities of the uniform IPPs and we leave it as future work.

Interactive Proofs for Agnostic Learning

[29] study the setting of verifying PAC-learners. There, the verifier has sampling access to an unknown distribution \mathcal{D} over labeled examples of the form (i, x_i) , where $i \sim \mathcal{D}$ and x is the underlying input. Its goal is to verify whether a hypothesis $h : \{0,1\}^{\log(n)} \rightarrow \{0,1\}$ given by the prover from a fixed hypothesis class, is the best approximation of \mathcal{D} . From the property testing perspective, the prover wants to convince the verifier that \mathcal{D}' has the property that every hypothesis in the class has error larger than ε over \mathcal{D} , for some $\varepsilon > 0$ (i.e., the best possible approximation of \mathcal{D} by the hypothesis class is at least ε).

¹⁴The uniform SIPP by [20] has communication complexity $\tilde{O}\left(\frac{n}{\tau} + \frac{1}{\varepsilon}\right)$ (for tradeoff $\tau \leq \sqrt{n}$), and using this still gives us the same communication complexity as the query-based distribution-free IPP from Theorem 1.

Similar to the setting of SIPPs, their scenario focuses on the case where the verifier only has access to x via a labeled sample oracle, over an unknown distribution. Furthermore, they focus on testing specific properties pertaining to machine learning, such as closeness to an underlying hypothesis class, with the hope of getting very low sample complexity (with respect to the VC dimension of the hypothesis class). In contrast, we deal with verification of general classes of properties, and in some cases the sample and query complexities are both $\tilde{O}(\sqrt{n})$.

References

- 1 Hugo Aaronson, Tom Gur, Ninad Rajgopal, and Ron Rothblum. Distribution-free proofs of proximity. *Electron. Colloquium Comput. Complex.*, TR23-118, 2023. [arXiv:TR23-118](#).
- 2 Vipul Arora, Arnab Bhattacharyya, Noah Fleming, Esty Kelman, and Yuichi Yoshida. Low degree testing over the reals. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 738–792. SIAM, 2023.
- 3 Aleksandrs Belovs. Quantum algorithm for distribution-free junta testing. In René van Bevern and Gregory Kucherov, editors, *Computer Science – Theory and Applications – 14th International Computer Science Symposium in Russia, CSR 2019, Novosibirsk, Russia, July 1-5, 2019, Proceedings*, volume 11532 of *Lecture Notes in Computer Science*, pages 50–59. Springer, 2019. [doi:10.1007/978-3-030-19955-5_5](#).
- 4 Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast Reed-Solomon interactive oracle proofs of proximity. 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 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. [doi:10.4230/LIPICs.ICALP.2018.14](#).
- 5 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 1–10, 2004.
- 6 Itay Berman, Ron D. Rothblum, and Vinod Vaikuntanathan. Zero-knowledge proofs of proximity. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPICs*, pages 19:1–19:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. [doi:10.4230/LIPICs.ITCS.2018.19](#).
- 7 Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing – Problems and Techniques*. Springer, 2022. [doi:10.1007/978-981-16-8622-1](#).
- 8 Eric Blais, Renato Pinto Jr Ferreira, and Nathaniel Harms. VC dimension and distribution-free sample-based testing. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 504–517, 2021.
- 9 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1), 2006. [doi:10.1561/0400000004](#).
- 10 Sarah Bordage, Mathieu Lhotel, Jade Nardi, and Hugues Randriam. Interactive oracle proofs of proximity to algebraic geometry codes. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 30:1–30:45. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. [doi:10.4230/LIPICs.CCC.2022.30](#).
- 11 Nader H. Bshouty. Almost optimal distribution-free junta testing. In Amir Shpilka, editor, *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*, volume 137 of *LIPICs*, pages 2:1–2:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. [doi:10.4230/LIPICs.CCC.2019.2](#).

- 12 Xi Chen and Shyamal Patel. Distribution-free testing for halfspaces (almost) requires pac learning. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1715–1743. SIAM, 2022.
- 13 Xi Chen and Jinyu Xie. Tight bounds for the distribution-free testing of monotone conjunctions. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 54–71. SIAM, 2016. doi:10.1137/1.9781611974331.ch5.
- 14 Alessandro Chiesa and Tom Gur. Proofs of proximity for distribution testing. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPICs*, pages 53:1–53:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.ITCS.2018.53.
- 15 Marcel Dall’Agnol, Tom Gur, Subhayan Roy Moulik, and Justin Thaler. Quantum proofs of proximity. *Quantum*, 6:834, 2022.
- 16 Funda Ergün, Ravi Kumar, and Ronitt Rubinfeld. Fast approximate probabilistically checkable proofs. *Inf. Comput.*, 189(2):135–159, 2004. doi:10.1016/j.ic.2003.09.005.
- 17 Eldar Fischer, Yonatan Goldhirsh, and Oded Lachish. Partial tests, universal tests and decomposability. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 483–500, 2014.
- 18 Noah Fleming and Yuichi Yoshida. Distribution-free testing of linear functions on \mathbb{R}^n . In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPICs*, pages 22:1–22:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.ITCS.2020.22.
- 19 Dana Glasner and Rocco A. Servedio. Distribution-free testing lower bounds for basic boolean functions. In Moses Charikar, Klaus Jansen, Omer Reingold, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, volume 4627 of *Lecture Notes in Computer Science*, pages 494–508. Springer, 2007. doi:10.1007/978-3-540-74208-1_36.
- 20 Guy Goldberg and Guy N Rothblum. Sample-based proofs of proximity. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
- 21 Oded Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.
- 22 Oded Goldreich. Testing bipartiteness in an augmented VDF bounded-degree graph model. *CoRR*, abs/1905.03070, 2019. arXiv:1905.03070.
- 23 Oded Goldreich. Testing graphs in vertex-distribution-free models. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 527–534, 2019.
- 24 Oded Goldreich. The uniform distribution is complete with respect to testing identity to a fixed distribution. In Oded Goldreich, editor, *Computational Complexity and Property Testing – On the Interplay Between Randomness and Computation*, volume 12050 of *Lecture Notes in Computer Science*, pages 152–172. Springer, 2020. doi:10.1007/978-3-030-43662-9_10.
- 25 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.
- 26 Oded Goldreich and Tom Gur. Universal locally verifiable codes and 3-round interactive proofs of proximity for CSP. *Theoretical computer science*, 878:83–101, 2021.
- 27 Oded Goldreich and Salil P Vadhan. On the complexity of computational problems regarding distributions. *Studies in Complexity and Cryptography*, 6650:390–405, 2011.
- 28 Shafi Goldwasser, Yael Tauman Kalai, and Guy N Rothblum. Delegating computation: interactive proofs for muggles. *Journal of the ACM (JACM)*, 62(4):1–64, 2015.
- 29 Shafi Goldwasser, Guy N Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

- 30 Tom Gur. *On locally verifiable proofs of proximity*. PhD thesis, The Weizmann Institute of Science (Israel), 2017.
- 31 Tom Gur, Govind Ramnarayan, and Ron Rothblum. Relaxed locally correctable codes. *Theory of Computing*, 16(1):1–68, 2020.
- 32 Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. *Comput. Complex.*, 27(1):99–207, 2018. doi:10.1007/s00037-016-0136-9.
- 33 Shirley Halevy and Eyal Kushilevitz. Distribution-free property-testing. *SIAM J. Comput.*, 37(4):1107–1138, 2007. doi:10.1137/050645804.
- 34 Shirley Halevy and Eyal Kushilevitz. Distribution-free connectivity testing for sparse graphs. *Algorithmica*, 51:24–48, 2008.
- 35 Tal Herman and Guy N Rothblum. Verifying the unseen: interactive proofs for label-invariant distribution properties. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1208–1219, 2022.
- 36 Yael Tauman Kalai and Ron D Rothblum. Arguments of proximity. In *Advances in Cryptology—CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 422–442. Springer, 2015.
- 37 Zhengyang Liu, Xi Chen, Rocco A Servedio, Ying Sheng, and Jinyu Xie. Distribution-free junta testing. *ACM Transactions on Algorithms (TALG)*, 15(1):1–23, 2018.
- 38 Zhengyang Liu, Xi Chen, Rocco A. Servedio, Ying Sheng, and Jinyu Xie. Distribution-free junta testing. *ACM Trans. Algorithms*, 15(1):1–1:23, 2019. doi:10.1145/3264434.
- 39 Omer Reingold, Guy N Rothblum, and Ron D Rothblum. Efficient batch verification for UP. In *33rd Computational Complexity Conference (CCC 2018)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.
- 40 Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. *SIAM J. Comput.*, 50(3), 2021. doi:10.1137/16M1096773.
- 41 Dana Ron and Asaf Rosin. Optimal distribution-free sample-based testing of subsequence-freeness with one-sided error. *ACM Transactions on Computation Theory (TOCT)*, 14(1):1–31, 2022.
- 42 Noga Ron-Zewi and Ron D Rothblum. Local proofs approaching the witness length. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 846–857. IEEE, 2020.
- 43 Guy N Rothblum and Ron D Rothblum. Batch verification and proofs of proximity with polylog overhead. In *Theory of Cryptography: 18th International Conference, TCC 2020, Durham, NC, USA, November 16–19, 2020, Proceedings, Part II*, pages 108–138. Springer, 2020.
- 44 Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC’13, Palo Alto, CA, USA, June 1-4, 2013*, pages 793–802. ACM, 2013. doi:10.1145/2488608.2488709.
- 45 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.
- 46 Amit Sahai and Salil P. Vadhan. Manipulating statistical difference. In Panos M. Pardalos, Sanguthevar Rajasekaran, and José Rolim, editors, *Randomization Methods in Algorithm Design, Proceedings of a DIMACS Workshop, Princeton, New Jersey, USA, December 12-14, 1997*, volume 43 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 251–270. DIMACS/AMS, 1997. doi:10.1090/dimacs/043/14.
- 47 Salil P Vadhan. An unconditional study of computational zero knowledge. *SIAM Journal on Computing*, 36(4):1160–1214, 2006.
- 48 Salil Pravin Vadhan. *A study of statistical zero-knowledge proofs*. PhD thesis, Massachusetts Institute of Technology, 1999.
- 49 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

On the Degree of Polynomials Computing Square Roots Mod p

Kiran S. Kedlaya   

Department of Mathematics, University of California San Diego, La Jolla, CA, USA
School of Mathematics of the Institute for Advanced Study (2023–24 academic year),
Princeton, NJ, USA

Swastik Kopparty  

Department of Mathematics and Department of Computer Science, University of Toronto, Canada

Abstract

For an odd prime p , we say $f(X) \in \mathbb{F}_p[X]$ *computes square roots* in \mathbb{F}_p if, for all nonzero perfect squares $a \in \mathbb{F}_p$, we have $f(a)^2 = a$.

When $p \equiv 3 \pmod{4}$, it is well known that $f(X) = X^{(p+1)/4}$ computes square roots. This degree is surprisingly low (and in fact lowest possible), since we have specified $(p-1)/2$ evaluations (up to sign) of the polynomial $f(X)$. On the other hand, for $p \equiv 1 \pmod{4}$ there was previously no nontrivial bound known on the lowest degree of a polynomial computing square roots in \mathbb{F}_p .

We show that for all $p \equiv 1 \pmod{4}$, the degree of a polynomial computing square roots has degree at least $p/3$. Our main new ingredient is a general lemma which may be of independent interest: powers of a low degree polynomial cannot have too many consecutive zero coefficients. The proof method also yields a robust version: any polynomial that computes square roots for 99% of the squares also has degree almost $p/3$.

In the other direction, Agou, Deliglése, and Nicolas [1] showed that for infinitely many $p \equiv 1 \pmod{4}$, the degree of a polynomial computing square roots can be as small as $3p/8$.

2012 ACM Subject Classification Computing methodologies \rightarrow Representation of mathematical functions; Computing methodologies \rightarrow Number theory algorithms; Mathematics of computing \rightarrow Coding theory

Keywords and phrases Algebraic Computation, Polynomials, Computing Square roots, Reed-Solomon Codes

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.25

Funding *Kiran S. Kedlaya*: Research supported by NSF grant DMS-2053473, the UC San Diego Warschawski Professorship, the Simons Fellows in Mathematics program of the Simons Foundation (2023–24 academic year).

Swastik Kopparty: Research supported by an NSERC Discovery Grant.

Acknowledgements Both authors acknowledge support from IAS in 2018–19, where initial discussions towards this paper took place. We thank N. Carella and Igor Shparlinski for valuable comments and pointers to the literature.

1 Introduction

Let p be an odd prime, and let \mathbb{F}_p be the finite field with p elements.

We say $f(X) \in \mathbb{F}_p[X]$ *computes square roots* in \mathbb{F}_p , if for all nonzero perfect squares $a \in \mathbb{F}_p$, we have:

$$f(a)^2 = a.$$

In other words, for each nonzero perfect square $a \in \mathbb{F}_p$, $f(a)$ is one of its two square roots.



© Kiran S. Kedlaya and Swastik Kopparty;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 25; pp. 25:1–25:14

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



When $p \equiv 3 \pmod{4}$, then it is well known that $f(X) = X^{(p+1)/4}$ computes square roots. This degree is surprisingly low, since we are essentially interpolating a polynomial from $(p-1)/2$ evaluations (where the evaluations are specified up to sign). We are interested in whether there is a similar phenomenon for $p \equiv 1 \pmod{4}$.

Concretely, we study the question: what is the smallest degree of a polynomial that computes square roots? Despite being a basic and natural question, there were no nontrivial bounds known for this question for the case of $p \equiv 1 \pmod{4}$.

There is a very simple argument¹ that shows that the degree of such a polynomial $f(X)$ must be at least $\frac{p-1}{4}$; indeed, the nonzero polynomial $f(X)^2 - X$ vanishes at all the $\frac{p-1}{2}$ nonzero perfect squares in \mathbb{F}_p .

Our main result is that, unlike the case of $p \equiv 3 \pmod{4}$, the degree of any polynomial computing square roots in the case of $p \equiv 1 \pmod{4}$ must be significantly higher, about $\frac{1}{3} \cdot p$.

► **Theorem 1.** *Let $p \equiv 1 \pmod{4}$. Then any polynomial that computes square roots in \mathbb{F}_p has degree at least $\frac{p-1}{3}$.*

Our proof is based on expressing the property of computing square roots as a polynomial relation (involving some unknown polynomials), and then eliminating the unknown polynomials through a combination of taking derivatives and truncations. Abstracting out the main steps, we get a general lemma (that can also be derived from the Mason-Stothers *abc*-theorem) which may be of independent interest: the powers of a low-degree polynomial cannot have too many consecutive zero coefficients (Lemma 5).

How does $p \pmod{4}$ play a role in the proof? Our proof ends up showing that for all p , any polynomial $f(X)$ of degree less than $\frac{p}{3}$ that computes square roots must have $f(X)^2 = X^{(p+1)/2}$ (as a polynomial identity), and this is not possible if $p \equiv 1 \pmod{4}$.

A robust version

The degree of a polynomial computing a certain function is quite a brittle notion. Changing just a single value of the function can change the degree drastically. By using the key idea of the Berlekamp-Welch algorithm for decoding Reed-Solomon codes, we can strengthen the above result to get a robust version, given below.

► **Theorem 2.** *Let $p \equiv 1 \pmod{4}$. Then any polynomial that computes square roots in \mathbb{F}_p on all but $e \leq \frac{p-7}{12}$ nonzero perfect squares in \mathbb{F}_p must have degree at least $\frac{p-1}{3} - e$.*

The connection to decoding algorithms for Reed-Solomon codes is not such a surprise. The problem of whether a low-degree polynomial can compute square roots is in fact a list-recovery problem for Reed-Solomon codes [9]; our result effectively shows that a certain algebraic list recovery instance where each input list has size 2 has no solutions. The difficulty is that this lies beyond the regime where we have a good understanding of list-recoverability and list-decodability of Reed-Solomon codes.

More concretely, let C be the Reed-Solomon code of degree d polynomials over \mathbb{F}_p with evaluation set D . Suppose for each $x \in D$ we are given a set $S_x \subseteq \mathbb{F}_p$ with $|S_x| \leq 2$. How can we certify that there are no codewords c of C such that for each coordinate $x \in D$, we have $c_x \in S_x$? It is not known how to give an efficiently verifiable certificate of this *in general* when $d = (\frac{1}{2} + \Omega(1)) |D|$. In our setting D is the set of perfect squares (so $|D| = (p-1)/2$), and d is $p/3$, which is outside the range of known certification methods [9].

¹ This argument works for all p , and thus we get that $X^{(p+1)/4}$ is the lowest degree polynomial computing square roots for $p \equiv 3 \pmod{4}$.

Better upper bounds for special p

It turns out that for some p which are $1 \pmod 4$, there are polynomials computing square roots with degree about $\frac{3}{8} \cdot p$.

► **Theorem 3.** *Let $p \equiv 5 \pmod 8$. Then there is a polynomial that computes square roots in \mathbb{F}_p with degree at most $\frac{3p+1}{8}$.*

This was shown by Agou, Deliglése and Nicolas [1] (see also [5]), based on the Tonelli-Shanks algorithm for computing square roots. Since it is quite simple and short, we include it in this paper for completeness.

The method of Tonelli-Shanks also yields similar phenomena with degree $(\frac{1}{2} - \Omega(1))p$ for p in special residue classes mod 2^j with j constant. Theorem 5 of [1] gives another example of such a phenomenon for $p \equiv 7 \pmod{12}$, giving polynomials (different from $X^{(p+1)/4}$) computing square roots of degree about $\frac{5}{12} \cdot p$.

Upper bounds for general p

Finally, we discuss upper bounds for the case of general p . First, a heuristic. There are $2^{(p-1)/2}$ different square root functions (the choice of sign for each perfect square). If the unique interpolating polynomials of degree $< (p-1)/2$ for these functions had their coefficients behaving randomly, then we would expect a polynomial of degree at most $\frac{1}{2}p - \Omega\left(\frac{p}{\log p}\right)$ that computes square roots.

Formalizing this intuition, we show that there is a polynomial computing square roots with degree $\frac{1}{2}p - \tilde{\Omega}(\sqrt{p})$. This is done by looking at the explicit formulas for the coefficients of the interpolants and arguing their pseudorandomness via the Weil bounds and some elementary Fourier analysis.

We also note that all our results have analogues for computing t -th roots.

Related Work

The work of Agou, Deliglése and Nicolas [1] gave examples, for infinitely many p , of polynomials in \mathbb{F}_p of abnormally low degree that compute square roots. The focus there was on finding polynomials with few monomials, and they gave interesting upper and lower bounds for this. Chang, Kim and Lee [6] gave analogues of these results for computing t -th roots.

Another related line of research has studied lower bounds on the degree of polynomials computing interesting arithmetic functions. Coppersmith and Shparlinski [7] (following an error-free computation result of Mullen and White [12]), gave strong lower bounds on the degree of polynomials computing the discrete logarithm in prime fields \mathbb{F}_p with as many as $(1 - o(1))p$ errors. Winterhof [20] later gave a generalization of this to all finite fields. These results are related to *list-decoding* of Reed-Solomon codes, since for each input there is only one “correct” value which we are hoping the polynomial will compute. As mentioned earlier, the problems we consider are related to *list-recovery* of Reed-Solomon codes, where there are multiple “correct” values for any given input, and we hope the polynomial computes one of them.

Conclusions and Questions

Computing square roots and understanding quadratic residuosity are central topics in algebraic computation and pseudorandomness.

25:4 On the Degree of Polynomials Computing Square Roots Mod p

Perhaps the most interesting and fundamental open question in this area is that of deterministically computing square roots mod p in $\text{poly}(\log p)$ time. As we already saw, when $p \equiv 3 \pmod{4}$, the simple deterministic $\text{poly}(\log p)$ algorithm of raising $x \in \mathbb{F}_p$ to the power $\frac{p+1}{4}$ computes the square root of x . Our results show that the $p \equiv 1 \pmod{4}$ case is qualitatively different in some respects. See [3, 18, 15] for what is known about this computational problem and related number theoretic issues.

Other important questions include the problem of determining the size of the least quadratic residue mod p (this is also connected to deterministic computation of square roots), and understanding the pseudorandomness of the Paley graph (for example, are Paley graphs Ramsey graphs?).

Finally, as mentioned above, our results can be viewed as showing that a certain list-recovery instance for Reed-Solomon codes has no solutions. We close with a conjecture about the list-recoverability of Reed-Solomon codes. The conjecture talks about prime fields; the results of Guruswami and Rudra [8] imply that this assumption cannot be dropped.

► **Conjecture 4.** *Let \mathbb{F}_p be a prime field. Let $\ell \in \mathbb{N}, \epsilon > 0$ be constants. Suppose we are given, for each $x \in \mathbb{F}_p$, a set S_x with $|S_x| \leq \ell$. Then:*

$$|\{P(X) \in \mathbb{F}_p[X] \mid \deg(P) \leq (1 - \epsilon)p, \text{ and for all } x \in \mathbb{F}_p, P(x) \in S_x\}| \leq p^{O_{\epsilon, \ell}(1)}.$$

We hope that our methods can give some insight into understanding the list-recovery capacity of Reed-Solomon codes, and in particular the above conjecture.

2 Lower bound for polynomials computing square roots

We now prove our first theorem about polynomials computing square roots mod p .

► **Theorem 1.** *Let $p \equiv 1 \pmod{4}$. Then the degree of any polynomial that computes square roots in \mathbb{F}_p is at least $\frac{p-1}{3}$.*

Proof. Suppose $f(X)$ is of degree $d < \frac{p-1}{3}$ and computes square roots in \mathbb{F}_p . Then, since $X^{(p-1)/2} - 1$ is the vanishing polynomial of the set of nonzero perfect squares in \mathbb{F}_p , we have:

$$f(X)^2 - X \equiv 0 \pmod{(X^{(p-1)/2} - 1)}.$$

Let $A(X)$ be the polynomial of degree $2d - (p-1)/2$ such that

$$f(X)^2 - X = A(X) \cdot (X^{(p-1)/2} - 1).$$

Let $B(X) = X - A(X)$. Then we get:

$$f(X)^2 = A(X) \cdot X^{(p-1)/2} + B(X), \tag{1}$$

where:

- $\deg(f(X)) = d$.
- $\deg(A(X)), \deg(B(X)) \leq 2d - (p-1)/2$.
- $A(X) \neq 0$. If $A(X) = 0$ then $f(X)^2 = X$, which is impossible for a polynomial $f(X)$.
- $B(X) \neq 0$. Otherwise $A(X) = X$, and $f(X)^2 = X^{(p+1)/2}$, which is possible only if $p \equiv 3 \pmod{4}$.

These conditions together will give us our lower bound on d .

Taking derivatives² of both sides of (1), we get:

$$2f(X)f'(X) = A'(X) \cdot X^{(p-1)/2} - \frac{1}{2}A(X)X^{(p-3)/2} + B'(X). \quad (2)$$

Computing $2f(X)^2f'(X)$ in two ways using (1) and (2), we get:

$$2f'(X)A(X)X^{(p-1)/2} + 2f'(X)B(X) = f(X) \left(X \cdot A'(X) - \frac{1}{2}A(X) \right) X^{(p-3)/2} + f(X)B'(X).$$

Now, using our assumption on d , the degrees of $2f'(X)B(X)$ and $f(X)B'(X)$ are both at most $3d - (p-1)/2 - 1 < (p-3)/2$, and thus taking the above equation mod $X^{(p-3)/2}$,

$$2f'(X)B(X) = f(X)B'(X).$$

Since $B(X) \neq 0$, we get $\frac{2f'(X)}{f(X)} = \frac{B'(X)}{B(X)}$. Since $2\deg(f), \deg(B) < p$, by a basic property of logarithmic derivatives, this implies $f(X)^2 = \lambda B(X)$ for some nonzero λ , contradicting the fact that $A(X) \neq 0$. (See Remark 1 in Section 2.2 for a precise statement and a proof.)

Thus our assumption that $d < \frac{p-1}{3}$ is wrong, and the theorem follows. ◀

2.1 Consecutive zero coefficients in powers of polynomials

We isolate the key step above as the following lemma:

► **Lemma 5.** *Let \mathbb{F} be a field of characteristic p . Let $f(X), A(X), B(X)$ be in $\mathbb{F}[X]$. Suppose*

$$f(X)^t = A(X) \cdot X^\ell + B(X) \quad (3)$$

where:

- $\deg(f(X)) \leq d < \frac{p}{t}$,
- $\deg(B(X)) \leq b < p$,
- $A(X) \neq 0$,
- $B(X) \neq 0$,

Then $d + b \geq \ell$.

In words, this says that if $dt < p$, the t -th power of a polynomial of degree d cannot have d consecutive 0 coefficients unless it is a single monomial (since the RHS of the above equation has at least $\ell - b - 1$ consecutive 0 coefficients).

Proof. Suppose $d + b < \ell$. Observe that this implies that $f(X) \neq 0$.

Taking derivatives of both sides of (3), we get:

$$tf(X)^{t-1}f'(X) = C(X)X^{\ell-1} + B'(X), \quad (4)$$

for some $C(X) \in \mathbb{F}[X]$.

Computing $tf(X)^t f'(X)$ in two different ways using (3), (4), we get:

$$tA(X)f'(X)X^\ell + tf'(X)B(X) = f(X)C(X)X^{\ell-1} + f(X)B'(X).$$

² Throughout this paper, we work with formal derivatives of polynomials.

25:6 On the Degree of Polynomials Computing Square Roots Mod p

Since $\deg(tf'(X)B(X)), \deg(f(X)B'(X)) < d+b \leq \ell-1$, by taking this equation mod $X^{\ell-1}$ we get:

$$tf'(X)B(X) = f(X)B'(X),$$

and since $f(X), B(X)$ are nonzero, we get that:

$$t \frac{f'(X)}{f(X)} = \frac{B'(X)}{B(X)}.$$

By the logarithmic derivative, we get $f(X)^t = \lambda B(X)$ for some nonzero λ , contradicting our assumption that $A(X) \neq 0$.

Thus $d+b \geq \ell$ as claimed. ◀

2.2 Remarks

1. The key fact about logarithmic derivatives that we are using is that if $f(X), B(X) \in \mathbb{F}_p[X]$, $t \cdot \deg(f), \deg(B) < p$, and:

$$t \frac{f'(X)}{f(X)} = \frac{B'(X)}{B(X)},$$

then $f(X)^t = \lambda \cdot B(X)$ for some constant $\lambda \in \mathbb{F}_p$.

We recap a quick proof. The hypothesis implies that $\left(\frac{f(X)^t}{B(X)}\right)' = 0$, and thus $\frac{f(X)^t}{B(X)}$ must be a rational function in X^p . To see the last deduction, note that $(f(X)^t \cdot B(X)^{p-1})' = \left(\frac{f(X)^t}{B(X)} \cdot B(X)^p\right)' = \left(\frac{f(X)^t}{B(X)}\right)' \cdot B(X)^p + \frac{f(X)^t}{B(X)} \cdot p \cdot B(X)^{p-1} \cdot B'(X) = 0 + 0 = 0$. Thus the polynomial $f(X)^t \cdot B(X)^{p-1}$ is a polynomial in X^p , which implies that $\frac{f(X)^t}{B(X)} = \frac{f(X)^t \cdot B(X)^{p-1}}{B(X)^p}$ is a rational function in X^p . Once we know that $\frac{f(X)^t}{B(X)}$ is a rational function in X^p , our assumption about the degrees implies the result.

2. The exact same proof also classifies when low-degree polynomials can compute square roots of very-low-degree polynomials on a multiplicative group.

► **Theorem 6.** *Let $G \subseteq \mathbb{F}_p^*$ be a multiplicative subgroup of size m , with $m \leq \frac{p-1}{2}$. Let $C(X) \in \mathbb{F}_p[X]$ have degree at most $\frac{m}{3}$. Suppose $f(X) \in \mathbb{F}_p[X]$ is such that $f(a)^2 = C(a)$ for all $a \in G$. Then one of the following alternatives must hold:*

- $f(X)^2 = C(X)$,
- $f(X)^2 = C(X) \cdot X^m$,
- $\deg(f) \geq \frac{2m}{3}$.

The above statement for $m = p-1$ and $C(X)$ being a constant follows from a result of Biro [4], who classified low-degree polynomials that take two values on \mathbb{F}_p^* . The proof from [4] is a delicate investigation of certain power sums. Our proof for $m < p-1$ is quite different, and has the flexibility of allowing for the robust version proved in the next section (which gives, for example, a classification of low-degree polynomials that take only 2 values on 99% of G).

In this generality, the bound of $\frac{2m}{3}$ is tight. If m is divisible by 3, then the polynomial $f(X) = (X^{2m/3} + X^{m/3} - \frac{1}{2})$ is such that $f(x)^2 = \frac{9}{4}$ for all $x \in G$ (since $x^{m/3}$ is a cube root of 1).

3. The proof of Lemma 5 also applies as is to *rational powers* of $f(X)$, where we now talk about consecutive 0 coefficients in the power series. We only state it for characteristic 0; it says that the power series expansion of $f(X)^{r/s}$, for $f(X)$ of degree d , does not have d consecutive 0 coefficients. Precisely, we have:

► **Lemma 7.** *Let \mathbb{F} be a field of characteristic 0. Let t be a rational number. Let $f(X) \in \mathbb{F}[X]$ be a polynomial of degree at most d with nonzero constant term. Then any (formal) power series expansion of $f(X)^t$ in $\mathbb{F}[[X]]$ does not have d consecutive zero coefficients.*

This is stronger than the usual bound for this situation (which shows up in polynomial factoring algorithms via the Hilbert irreducibility theorem [10] and the Arora-Sudan low degree test [2]), which goes as follows: Suppose $f(X)^{1/s} = A(X)X^\ell + B(X)$, where $\deg(f) = d$, $\deg(B) = b$ and $A(X) \in \mathbb{F}[[X]]$ is nonzero, then $f(X) - B(X)^s$ is a nonzero polynomial of degree at least ℓ , and so $\ell \leq \max(sb, d)$. Thus if b is large, this bound only guarantees that there is a nonzero coefficient X^i for $i \in [b+1, sb]$ (instead of $[b+1, b+d]$ as guaranteed by Lemma 7).

4. Applying the same method, we can apply this method to the power series expansion of $e^{f(X)}$ too.

► **Lemma 8.** *Let \mathbb{F} be a field of characteristic 0. Let $f(X) \in \mathbb{F}[X]$ be a polynomial of degree at most d with constant term 0. Then the (formal) power series expansion of $e^{f(X)}$ in $\mathbb{F}[[X]]$ does not have d consecutive zero coefficients.*

5. The bounds of Lemma 5, Lemma 7 and Lemma 8 on the number of consecutive 0 coefficients are tight, for example when $f(X)$ is of the form $\alpha X^d + \beta$.
6. We can give another proof of Lemma 5 (but not Lemma 7 or Lemma 8 as far as we know) using the Mason-Stothers *abc*-theorem for polynomials [16, 11].

Indeed, note that $f(X)^t$, $A(X) \cdot X^\ell$ and $B(X)$ all have degree at most dt . Furthermore, the radical of their product divides $f(X) \cdot A(X) \cdot X \cdot B(X)$, and thus has degree at most $d + (dt - \ell) + 1 + b$. By the *abc*-theorem, we get that:

$$dt \leq (d + dt - \ell + 1 + b) - 1 = dt + d - \ell + b,$$

and thus $d + b \geq \ell$.

3 A robust version

Let p be a prime that is 1 mod 4. Let S be the set of nonzero perfect squares in \mathbb{F}_p .

We say a polynomial $f(X) \in \mathbb{F}_p[X]$ computes square roots with error e if:

$$|\{a \in S \mid f(a)^2 \neq a\}| \leq e.$$

We show that any polynomial computing square roots even allowing $\Omega(p)$ error cannot have degree much smaller than $p/3$.

► **Theorem 2.** *Let $p \equiv 1 \pmod{4}$. Suppose $f(X) \in \mathbb{F}_p[X]$ is a polynomial of degree d that computes square roots with error e .*

Then

$$d \geq \begin{cases} \frac{p-1}{3} - e & e \leq \frac{p-7}{12} \\ \frac{p-1}{2} - 3e - 1 & e > \frac{p-7}{12}. \end{cases}$$

Proof. We use the idea of the Berlekamp-Welch Reed-Solomon decoding algorithm [19].

Let $U \subseteq S$ be the set of $a \in S$ where $f(a)^2 \neq a$.

Let $E(X) \in \mathbb{F}_p[X]$ be the vanishing polynomial of U , given by:

$$E(X) = \prod_{u \in U} (X - u).$$

25:8 On the Degree of Polynomials Computing Square Roots Mod p

Note that E is a nonzero polynomial of degree at most e .

Then we have:

$$E(X)^2 \cdot f(X)^2 \equiv E(X)^2 \cdot X \pmod{(X^{(p-1)/2} - 1)}.$$

Let $A(X)$ be the polynomial of degree at most $2(e + d) - (p - 1)/2$ such that:

$$E(X)^2 \cdot f(X)^2 - E(X)^2 \cdot X = A(X) \cdot (X^{(p-1)/2} - 1).$$

Let $g(X) = E(X) \cdot f(X)$, and $B(X) = E(X)^2 \cdot X - A(X)$.

Then

$$g(X)^2 = A(X) \cdot X^{(p-1)/2} + B(X).$$

We have:

- $\deg(g) \leq d + e$,
- $\deg(B) \leq \max(2e + 1, 2(e + d) - (p - 1)/2)$.
- $A(X) \neq 0$. Otherwise $E(X)^2 \cdot f(X)^2 = E(X)^2 \cdot X \implies f(X)^2 = X$, which is impossible.
- $B(X) \neq 0$. Otherwise $E(X)^2 \cdot X = A(X)$, and so $E(X)^2 f(X)^2 = E(X)^2 X^{(p+1)/2}$, which implies that $f(X)^2 = X^{(p+1)/2}$. This is only possible if $p \equiv 3 \pmod{4}$.

Plugging this into Lemma 5, we get:

■

$$(d + e) + \left(2(d + e) - \frac{p - 1}{2}\right) \geq \frac{p - 1}{2},$$

if $2d - (p - 1)/2 \geq 1$,

■

$$(d + e) + (2e + 1) \geq \frac{p - 1}{2},$$

if $2d - (p - 1)/2 \leq 0$.

This tells us that either:

$$d \geq \frac{p - 1}{3} - e,$$

or:

$$d \geq \frac{p - 1}{2} - 3e - 1,$$

and thus:

$$d \geq \min\left(\frac{p - 1}{3} - e, \frac{p - 1}{2} - 3e - 1\right),$$

which gives us the desired claim. ◀

Note that there is another simple lower bound (which applies for all p) of $d + \frac{e}{2} \geq \frac{p-1}{4}$ (the simple lower bound is better for $e > \frac{p-1}{10}$). This is proved by considering the number of roots of the degree $2d$ polynomial $f(X)^2 - X$.

4 Upper bound for special p

In this section, we give an upper bound on the degree of polynomials computing square roots mod p , for infinitely many $p \equiv 1 \pmod{4}$. The upper bound is best when $p \equiv 5 \pmod{8}$, and we only present this case. The result and proof of this section is due to Agou, Deliglése and Nicolas [1]. It remains in this paper only for completeness.

► **Theorem 3.** *Let $p \equiv 5 \pmod{8}$. Then there is a polynomial that computes square roots in \mathbb{F}_p with degree at most $\frac{3p+1}{8}$.*

Proof. Since $p \equiv 1 \pmod{4}$, we get that -1 is a perfect square mod p . Let $i \in \mathbb{F}_p$ be one of the square roots of -1 . Our main ingredient is the Tonelli-Shanks algorithm [14, 17] computing square roots mod p . For $p = 4\ell + 1$, the algorithm essentially gives a formula for the square root depending on two cases. Specifically, let $u : S \rightarrow \mathbb{F}_p$ given by:

$$u(a) = \begin{cases} a^{(p+3)/8} & a^{(p-1)/4} = 1, \\ i \cdot a^{(p+3)/8} & a^{(p-1)/4} = -1. \end{cases}$$

Then for all $a \in \mathbb{F}_p$, $u(a)$ is a square root of a .

This is already quite special; the set S is partitioned into two equal sized parts S_0 and S_1 , and on each S_i we have a polynomial $f_i(X)$ computing the square root of degree about $\frac{1}{2}|S_i|$. (This is the lowest possible degree, since $f_i(X)^2 - X$ is a nonzero polynomial that vanishes on all of S_i .)

Usually if we have this kind of setup, even though the f_i have unusually low degree, the unique polynomial f (obtained from the Chinese remainder theorem) which restricts to f_i on S_i has no reason to have unusually low degree. But in this case it does!

Using the usual Chinese remainder formula, we consider the polynomial $f(X) \in \mathbb{F}_p[X]$ given by:

$$\begin{aligned} f(X) &= \frac{1}{2} \left(X^{(p+3)/8} (X^{(p-1)/4} + 1) - i \cdot X^{(p+3)/8} (X^{(p-1)/4} - 1) \right) \\ &= \frac{1-i}{2} X^{(3p+1)/8} + \frac{1+i}{2} X^{(p+3)/8}. \end{aligned}$$

By design, we have $f(a) = u(a)$ for all $a \in S$. Finally, notice that $\deg(f) \leq \frac{3p+1}{8}$.

As a sanity check, we directly verify that $f(X)^2 \equiv X \pmod{(X^{(p-1)/2} - 1)}$. Indeed,

$$\begin{aligned} f(X)^2 &= \left(\frac{1-i}{2} \right)^2 X^{(3p+1)/4} + 2 \cdot \frac{(1-i)(1+i)}{4} X^{(4p+4)/8} + \left(\frac{1+i}{2} \right)^2 X^{(p+3)/4} \\ &= -\frac{i}{2} X^{(3p+1)/4} + X^{(p+1)/2} + \frac{i}{2} X^{(p+3)/4} \\ &= \left(-\frac{i}{2} X^{(p+3)/4} + X \right) \cdot (X^{(p-1)/2} - 1) + X, \end{aligned}$$

as desired. ◀

5 Upper bounds for general p

In this section, we give a slightly nontrivial upper bound on the degree of polynomials computing square roots for all p . We will show that there is a polynomial with degree somewhat less than $\frac{p}{2}$ which computes square roots.

25:10 On the Degree of Polynomials Computing Square Roots Mod p

► **Theorem 9.** For all odd primes p , for $t = o\left(\frac{\sqrt{p}}{\log^2 p}\right)$, there is a polynomial of degree $\frac{p}{2} - t$ which computes square roots.

Proof. Let $m = (p-1)/2$. Let $S \subseteq \mathbb{F}_p$ be the set of nonzero perfect squares, and note that $|S| = m$. For each $\alpha \in S$, let $\delta_\alpha(X) \in \mathbb{F}_p[X]$ be the unique polynomial of degree $\leq (m-1)$ such that for all $\beta \in S$:

$$\delta_\alpha(\beta) = \begin{cases} 1 & \beta = \alpha, \\ 0 & \beta \neq \alpha. \end{cases}$$

Explicitly, we have:

$$\delta_\alpha(X) = \frac{1}{m} \left(\left(\frac{X}{\alpha}\right)^{m-1} + \left(\frac{X}{\alpha}\right)^{m-2} + \dots + \frac{X}{\alpha} + 1 \right).$$

Then given a function $u : S \rightarrow \mathbb{F}_p$, the unique polynomial $f(X) \in \mathbb{F}_p[X]$ of degree at most $m-1$ such that $f(\alpha) = u(\alpha)$ for all $\alpha \in S$ is given by:

$$f(X) = \sum_{\alpha \in S} u(\alpha) \delta_\alpha(X).$$

Our goal is to pick u where each $u(\alpha)$ is one of the two square roots of α so that many of the leading coefficients of $f(X)$ equal 0.

We now use the structure of S . Let g be a generator of \mathbb{F}_p^* . Then $S = \{g^{2j} \mid 0 \leq j < (p-1)/2\}$. Furthermore, for $\alpha = g^{2j} \in S$, one of the two square roots of α is g^j .

Thus, our problem can be reformulated as choosing a function $v : S \rightarrow \{\pm 1\}$ such that:

$$f(X) = \sum_{j=0}^{(p-1)/2} v(g^{2j}) \cdot g^j \cdot \delta_{g^{2j}}(X)$$

has many leading coefficients equal to 0.

Observe that the coefficient of X^{m-i} in $f(X)$ equals:

$$\frac{1}{m} \sum_{j=0}^{(p-1)/2} v(g^{2j}) g^j \left(\frac{1}{g^{2j}}\right)^{m-i} = \frac{1}{m} \sum_{i=0}^{(p-1)/2} v(g^{2j}) g^{(2i+1)j}.$$

Thus, to get a polynomial $f(X)$ of degree $< m-t$, we want to find a vector $v \in \{\pm 1\}^m$ that lies in the kernel of the Vandermonde-type matrix $M \in \mathbb{F}_p^{t \times m}$, where:

$$M_{i,j} = g^{(2i+1)j}.$$

(The row index i runs from 1 to t , the column index j runs from 0 to $m-1$.)

For later use, for a vector $y \in \mathbb{F}_p^t$, we define $P_y(Z) \in \mathbb{F}_p[Z]$ to be the polynomial:

$$P_y(Z) = \sum_{i=1}^t y_i Z^{2i+1}.$$

Thus for $j \in \{0, 1, \dots, m-1\}$, the j th entry of $M^T y$ equals $P_y(g^j)$.

To show that there exists the desired ± 1 vector, we count the number of such vectors using Fourier analysis. Let ω be a p th root of unity in \mathbb{C} . The number of such ± 1 vectors equals:

$$\begin{aligned}
N &= \sum_{v \in \{\pm 1\}^m} \mathbf{1}_{Mv=0} \\
&= \sum_{v \in \{\pm 1\}^m} \mathbb{E}_{y \in \mathbb{F}_p^t} \left[\omega^{\langle y, Mv \rangle} \right] \\
&= \mathbb{E}_y \left[\sum_v \omega^{\langle M^T y, v \rangle} \right] \\
&= \mathbb{E}_y \left[\sum_v \omega^{\sum_{j=0}^{m-1} (M^T y)_j \cdot v_j} \right] \\
&= \mathbb{E}_y \left[\sum_v \prod_{j=0}^{m-1} \omega^{(M^T y)_j \cdot v_j} \right] \\
&= \mathbb{E}_y \left[\sum_v \prod_{j=0}^{m-1} \omega^{P_y(g^j) \cdot v_j} \right].
\end{aligned}$$

For $y = 0$, the expression inside the expectation equals 2^m . We will show that for the remaining $p^t - 1$ values of y , the expression inside the expectation is very small.

Fix any $y \neq 0$. The expression inside the expectation equals:

$$\sum_{v \in \{\pm 1\}^m} \prod_{j=0}^{m-1} \omega^{P_y(g^j) \cdot v_j} = \prod_{j=1}^m \left(\omega^{P_y(g^j)} + \omega^{-P_y(g^j)} \right). \quad (5)$$

The next lemma (which uses the Weil bounds on mixed character sums) shows that for any nonzero y , the evaluations of the polynomial P_y at $\{1, g, g^2, \dots, g^{m-1}\}$ are well distributed in \mathbb{F}_p .

► **Lemma 10.** *Let $0 \leq \alpha < \beta \leq 1$. Let $y \in \mathbb{F}_p^t \setminus \{0\}$. Then:*

$$\Pr_{j \in \{0, 1, \dots, m-1\}} [P_y(g^j) \in [\alpha p, \beta p]] = (\beta - \alpha) + O\left(\frac{t \log^2 p}{\sqrt{p}}\right).$$

Assuming the lemma, we get that for $t = o\left(\frac{\sqrt{p}}{\log^2 p}\right)$, the product in Equation (5) is at most $2^m \cdot \exp(-m)$. Thus:

$$\begin{aligned}
N &\geq \frac{2^m}{p^t} - \max_{y \neq 0} \left| \prod_{j=1}^m \left(\omega^{P_y(g^j)} + \omega^{-P_y(g^j)} \right) \right| \\
&\geq \frac{2^m}{p^t} - O(2^m \exp(-m)) \\
&\geq 2^m \left(\frac{1}{p^t} - \exp(-m) \right) \\
&> 0,
\end{aligned}$$

where the penultimate inequality holds since

$$t = o\left(\frac{\sqrt{p}}{\log^2 p}\right) \ll \frac{p}{\log p} = \Theta\left(\frac{m}{\log p}\right).$$

This completes the proof. ◀

5.1 Distribution of values of $P_y(g^j)$

We now prove Lemma 10.

Proof. Let I be the interval $[\alpha p, \beta p]$. Let J be the set $\{1, g, g^2, \dots, g^{m-1}\}$. Then the probability in the statement of the lemma equals:

$$\frac{1}{m} \sum_{z \in \mathbb{F}_p} \mathbf{1}_I(P_y(z)) \mathbf{1}_J(z). \quad (6)$$

I is an interval in the additive group of \mathbb{F}_p . By standard results, if we expand $\mathbf{1}_I$ in its additive Fourier expansion:

$$\mathbf{1}_I = \sum_{\psi} \widehat{\mathbf{1}}_I(\psi) \psi$$

(where the ψ are the additive characters), then:

$$\sum_{\psi} |\widehat{\mathbf{1}}_I(\psi)| \leq O(\log p). \quad (7)$$

Similarly, J is an interval in the multiplicative group of \mathbb{F}_p . If we expand $\mathbf{1}_J$ in its additive Fourier expansion:

$$\mathbf{1}_J = \sum_{\chi} \widetilde{\mathbf{1}}_J(\chi) \chi$$

(where the χ are the multiplicative characters), then:

$$\sum_{\chi} |\widetilde{\mathbf{1}}_J(\chi)| \leq O(\log p). \quad (8)$$

Then the probability in Equation (6) equals:

$$\begin{aligned} & \frac{1}{m} \sum_z \left(\sum_{\psi} \widehat{\mathbf{1}}_I(\psi) \psi(P_y(z)) \right) \left(\sum_{\chi} \widetilde{\mathbf{1}}_J(\chi) \chi(z) \right) \\ &= \frac{1}{m} \sum_{\psi, \chi} \widehat{\mathbf{1}}_I(\psi) \widetilde{\mathbf{1}}_J(\chi) \left(\sum_z \psi(P_y(z)) \chi(z) \right) \\ &= \frac{1}{m} \left(\frac{|I|}{p} \cdot \frac{|J|}{p} \cdot p + O \left(\sum_{(\psi, \chi) \neq (1, 1)} |\widehat{\mathbf{1}}_I(\psi)| \cdot |\widetilde{\mathbf{1}}_J(\chi)| \cdot \left| \sum_z \psi(P_y(z)) \chi(z) \right| \right) \right) \\ &= (\beta - \alpha) + \frac{1}{m} \cdot O \left(\sum_{(\psi, \chi) \neq (1, 1)} |\widehat{\mathbf{1}}_I(\psi)| \cdot |\widetilde{\mathbf{1}}_J(\chi)| \cdot \left| \sum_z \psi(P_y(z)) \chi(z) \right| \right) \end{aligned}$$

The Weil bound for mixed character sums (see [13]) shows that whenever (ψ, χ) are not both trivial characters, the inner expression is bounded:

$$\left| \sum_z \psi(P_y(z)) \chi(z) \right| \leq O(t\sqrt{p}).$$

Combined with the bounds in Equations (7), (8), we get the desired bound on the probability. \blacktriangleleft

6 t -th roots

Now we discuss t -th roots in place of square roots. We think of t as a constant, and the prime p growing. Let $p \equiv 1 \pmod t$, so that the set of nonzero t -th powers in \mathbb{F}_p has size $\frac{p-1}{t}$.

Just like in the case $t = 2$, for special p there is a simple formula for computing the t -th root; when $p \equiv 1 - t \pmod{t^2}$, then $a^{\frac{p+t-1}{t^2}}$ is a t -th root of a whenever a is a perfect t -th power in \mathbb{F}_p . Thus there is a polynomial of degree $\frac{1}{t^2} \cdot p + O(1)$ computing t -th roots. This matches the trivial lower bound of $\frac{p-1}{t^2}$ on the degree of polynomials computing t -th root (proved by counting zeroes of the nonzero polynomial $f(X)^t - X$).

An immediate generalization of Theorem 2 shows that for all other p (namely, $p \not\equiv 1 - t \pmod{t^2}$, but we still preserve the condition that $p \equiv 1 \pmod t$), any polynomial of degree d computing t -th roots in \mathbb{F}_p with error $e \leq \frac{t-1}{t^2(t+1)} \cdot (p-1) - 1$ must satisfy

$$d \geq \frac{2}{t(t+1)} \cdot (p-1) - e.$$

This is $\frac{2t}{t+1}$ times (which is about double for large t) the trivial lower bound, but quite far from the obvious upper bound (from Lagrange interpolation) of $\frac{1}{t} \cdot (p-1)$.

The best upper bound we know for p not of the special form $p \equiv 1 - t \pmod{t^2}$ is for p such that $2p \equiv 2 - t \pmod{t^2}$ (there are infinitely many such p for any odd t), and in this case the polynomial $f(X) = X^{\frac{2p+t-2}{t^2}}$ computes t -th roots. This is of the form $\frac{2}{t^2} \cdot p + O(1)$, and thus somewhat close to our lower bound for large t .

Closing these gaps seems like a very basic and interesting open question.

References

- 1 Simon Joseph Agou, Marc Deléglise, and Jean-Louis Nicolas. Short polynomial representations for square roots modulo p . *Designs, Codes and Cryptography*, 28(1):33–44, 2003.
- 2 Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 485–495, 1997.
- 3 Eric Bach and Jeffrey Outlaw Shallit. *Algorithmic number theory: Efficient algorithms*, volume 1. MIT press, 1996.
- 4 András Biró. On polynomials over prime fields taking only two values on the multiplicative group. *Finite Fields and Their Applications*, 6(4):302–308, 2000.
- 5 NA Carella. Formulas for the square root modulo p . *arXiv preprint*, 2011. [arXiv:1101.4605](https://arxiv.org/abs/1101.4605).
- 6 Seunghwan Chang, Bihtnara Kim, and Hyang-Sook Lee. Polynomial representations for n -th roots in finite fields. *Journal of the Korean Mathematical Society*, 52(1):209–224, 2015.
- 7 Don Coppersmith and Igor Shparlinski. On polynomial approximation of the discrete logarithm and the Diffie–Hellman mapping. *Journal of Cryptology*, 13:339–360, 2000.
- 8 Venkatesan Guruswami and Atri Rudra. Limits to list decoding reed-solomon codes. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 602–609, 2005.
- 9 Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 28–37. IEEE, 1998.
- 10 Erich Kaltofen. Effective noether irreducibility forms and applications. In *Proceedings of the twenty-third annual ACM symposium on Theory of Computing*, pages 54–63, 1991.
- 11 Richard Clive Mason. *Diophantine equations over function fields*, volume 96. Cambridge University Press, 1984.
- 12 Gary Mullen and David White. A polynomial representation for logarithms in $\text{gf}(q)$. *Acta arithmetica*, 3(47):255–261, 1986.

25:14 On the Degree of Polynomials Computing Square Roots Mod p

- 13 Wolfgang M Schmidt. *Equations over finite fields: an elementary approach*, volume 536. Springer, 2006.
- 14 Daniel Shanks. Five number-theoretic algorithms. In *Proceedings of the Second Manitoba Conference on Numerical Mathematics (Winnipeg), 1973*, 1973.
- 15 Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge university press, 2009.
- 16 W Wilson Stothers. Polynomial identities and hauptmoduln. *The Quarterly Journal of Mathematics*, 32(3):349–370, 1981.
- 17 Alberto Tonelli. Bemerkung über die Auflösung quadratischer Congruenzen. *Nachrichten von der Königl. Gesellschaft der Wissenschaften und der Georg-Augusts-Universität zu Göttingen*, 1891:344–346, 1891.
- 18 Joachim Von Zur Gathen and Jürgen Gerhard. *Modern computer algebra*. Cambridge university press, 2013.
- 19 Lloyd R Welch and Elwyn R Berlekamp. Error correction for algebraic block codes. *US patent*, 4,633,470, 1983.
- 20 Arne Winterhof. Polynomial interpolation of the discrete logarithm. *Designs, Codes and Cryptography*, 25(1):63–72, 2002.

Dimension Independent Disentanglers from Unentanglement and Applications

Fernando Granha Jeronimo  

Institute for Advanced Studies, Princeton, NJ, USA
Simons Institute, Berkeley, CA, USA

Pei Wu  

Weizmann Institute of Science, Rehovot, Israel

Abstract

Quantum entanglement, a distinctive form of quantum correlation, has become a key enabling ingredient in diverse applications in quantum computation, complexity, cryptography, etc. However, the presence of unwanted adversarial entanglement also poses challenges and even prevents the correct behaviour of many protocols and applications.

In this paper, we explore methods to “break” the quantum correlations. Specifically, we construct a *dimension-independent* k -partite disentangler (like) channel from bipartite unentangled input. In particular, we show: For every $d, \ell \geq k \in \mathbb{N}^+$, there is an efficient channel $\Lambda: \mathbb{C}^{d\ell} \otimes \mathbb{C}^{d\ell} \rightarrow \mathbb{C}^{dk}$ such that for every bipartite separable density operator $\rho_1 \otimes \rho_2$, the output $\Lambda(\rho_1 \otimes \rho_2)$ is close to a k -partite separable state. Concretely, for some distribution μ on states from \mathbb{C}^d ,

$$\left\| \Lambda(\rho_1 \otimes \rho_2) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu(\psi) \right\|_1 \leq \tilde{O} \left(\left(\frac{k^3}{\ell} \right)^{1/4} \right).$$

Moreover, $\Lambda(|\psi\rangle\langle\psi|^{\otimes \ell} \otimes |\psi\rangle\langle\psi|^{\otimes \ell}) = |\psi\rangle\langle\psi|^{\otimes k}$. Without the bipartite unentanglement assumption, the above bound is conjectured to be impossible and would imply $\text{QMA}(2) = \text{QMA}$.

Leveraging multipartite unentanglement ensured by our disentanglers, we achieve the following: (i) a new proof that $\text{QMA}(2)$ admits arbitrary gap amplification; (ii) a variant of the swap test and product test with improved soundness, addressing a major limitation of their original versions. More importantly, we demonstrate that unentangled quantum proofs of almost general real amplitudes capture NEXP, thereby greatly relaxing the non-negative amplitudes assumption in the recent work of $\text{QMA}^+(2) = \text{NEXP}$ [Jeronimo and Wu, STOC 2023]. Specifically, our findings show that to capture NEXP, it suffices to have unentangled proofs of the form $|\psi\rangle = \sqrt{a}|\psi_+\rangle + \sqrt{1-a}|\psi_-\rangle$ where $|\psi_+\rangle$ has non-negative amplitudes, $|\psi_-\rangle$ only has negative amplitudes and $|a - (1-a)| \geq 1/\text{poly}(n)$ with $a \in [0, 1]$. Additionally, we present a protocol achieving an almost largest possible completeness-soundness gap before obtaining $\text{QMA}^{\mathbb{R}}(k) = \text{NEXP}$, namely, a $1/\text{poly}(n)$ additive improvement to the gap results in this equality.

2012 ACM Subject Classification Theory of computation \rightarrow Interactive proof systems; Theory of computation \rightarrow Quantum information theory

Keywords and phrases QMA(2), disentangler, quantum proofs

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.26

Related Version *Full Version:* <https://arxiv.org/abs/2402.15282>

Funding This material is based on work supported by the National Science Foundation under Grant No. CCF-1900460. Part of the work is done when P.W. was at IAS and the Simons Institute.

1 Introduction

Quantum entanglement is a fundamental form of quantum correlation that can be stronger than any classical correlation [13, 5, 11, 21]. It plays a crucial role in a myriad of areas such as quantum computing, quantum information, quantum complexity, quantum cryptography,



© Fernando Granha Jeronimo and Pei Wu;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 26; pp. 26:1–26:28

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



26:2 Dimension Independent Disentglers from Unentanglement and Applications

condensed matter physics, etc [19, 25, 32]. Hence, comprehending both the capabilities and constraints of quantum entanglement stands as a crucial research endeavor. However, entanglement can also pose challenges in numerous applications, such as quantum key distribution and quantum proof systems [28, 23, 26, 15]. This raises the natural question of designing quantum channels that convert quantum states into unentangled states. For the purpose of applications, such channel, also called *disentangler*, $\Phi : \mathcal{H} \rightarrow \mathcal{K} \otimes \mathcal{K}$ can be defined to satisfy two conditions: (i) for any $|\psi\rangle \in \mathcal{K}$, there is preimage $|\phi\rangle$, such that $\Phi(\phi) = \psi \otimes \psi$; and (ii) for any density operator $\phi \in \mathcal{H}$, $\Phi(\phi)$ is close to *separable*.

The *quantum de Finetti* type theorems [10, 24, 28] provide examples of disentanglers. A quantum de Finetti theorem quantifies the closeness of a *permutation-invariant* ℓ -partite quantum state, to k -partite separable states when all but k subsystems are traced out. A standard quantum de Finetti theorem reads

► **Theorem 1** (Quantum de Finetti [24]). *For every $d, \ell \geq k \in \mathbb{N}^+$, the channel $\Lambda : (\mathbb{C}^d)^{\otimes \ell} \rightarrow (\mathbb{C}^d)^{\otimes k}$ defined as $\Lambda(\rho) = \text{Tr}_{\ell-k}(1/\ell! \sum_{\pi \in \text{Sym}_\ell} \pi \rho \pi^\dagger)$ satisfies*

$$\left\| \Lambda(\rho) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \leq \frac{2kd^2}{\ell}.$$

Note that the error bound scales at least¹ as d/ℓ , and in this version of the quantum de Finetti theorem, the parameters are known to be essentially tight. Consequently, if each subsystem is composed of n qubits, then obtaining a non-trivial error bound requires at least $\ell \geq d = 2^n$ subsystems, making this channel impractical for many applications. This is conjectured to be essentially the best you can achieve. In particular, it is conjectured that for any disentangler, the input dimension will be exponential in the output dimension [1] to achieve that the output is always ε close in trace distance to some separable states for any constant $\varepsilon < 1$.

Dimension Independent Disentangler from Unentanglement

While the original disentangler conjecture remains widely open, in this work, we show that there is an explicit, efficient (BQP), and *dimension independent* quantum disentangler for k -partite (output) system starting from a bipartite unentangled system. More precisely, we prove

► **Theorem 2** (Disentangler from unentanglement). *Let $d, \ell \geq k \in \mathbb{N}^+$. There is an efficient channel $\Lambda : (\mathbb{C}^d)^{\otimes \ell} \otimes (\mathbb{C}^d)^{\otimes \ell} \rightarrow (\mathbb{C}^d)^{\otimes k}$ such that for any density operators $\rho_1, \rho_2 \in \mathbb{C}^{d^\ell}$ there is a distribution μ on pure states $|\psi\rangle \in \mathbb{C}^d$ satisfying*

$$\left\| \Lambda(\rho_1 \otimes \rho_2) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \leq \tilde{O} \left(\left(\frac{k^3}{\ell} \right)^{1/4} \right).$$

Furthermore, product states of the form $\rho_1 = \rho_2 = |\psi\rangle\langle\psi|^{\otimes \ell}$ are mapped to $|\psi\rangle\langle\psi|^{\otimes k}$.

In contrast to the de Finetti disentangler, our disentangler from unentanglement features error parameters that are independent of the input dimension entirely! Subsequently, we discuss applications of Theorem 2 in testing product states and the gap amplification in

¹ If instead of making the state permutation invariant, we project it onto the symmetric subspace, which is a perfectly valid and efficient operation in the quantum setting, then the dependence on d in Theorem 1 improves from d^2 to d .

quantum proof systems, culminating in a near-optimal gap amplification for the $\text{QMA}^+(k)$ class: Any improvement on this gap amplification would imply $\text{QMA}^{\mathbb{R}}(k) = \text{NEXP}$.² We also anticipate that our tool will find further applications beyond those discussed in this paper.

1.1 Super Product Test

The *product test* was designed to test if a state $|\phi\rangle$ is close to k -partite product state, i.e., $|\phi\rangle \approx |\phi_1\rangle \otimes \cdots \otimes |\phi_k\rangle$, given two copies of $|\phi\rangle$. This test involves applying a sequence of swap tests to each of the k subsystems of the two copies $|\phi\rangle$. Clearly, if $|\phi\rangle$ is indeed a k -partite product state, all the swap tests accept with certainty. On the other hand, if $|\phi\rangle$ is entangled across the k subsystems, some swap test will reject with a probability that depends on the amount of entanglement. It can be argued that the product test is optimal for ensuring perfect completeness, i.e., accepting product states with certainty [17].

Despite its utility and elegance, the product test has two limitations. Firstly, it only provides a guarantee concerning its input $|\phi\rangle \otimes |\phi\rangle$ which are destroyed after the test, yielding a single classical bit as output. Very often in applications, one also needs some extra certified input states $|\phi\rangle$ to manipulate in subsequent computations after the test. Secondly, and probably more irritatingly, the product test always accepts with some constant probability (say $\geq 1/2$) no matter how far $|\phi\rangle$ is from being k -partite product, i.e., it has poor soundness. These limitations can be resolved if you have more than 2 copies of $|\phi\rangle$ [22, 29]. For instance, given ℓ copies of $|\phi\rangle$, then one can adapt the product test to sequentially apply projections on to symmetric subspace on the first, second, and subsequent subsystems of all the copies of ϕ . Intuitively, this should give us a stronger test whose analysis was left as an open problem in [17]. Recently, She and Yuen [29] analyzed this higher order version of product test achieving improved soundness. We restate this higher order product test as relying on some ℓ unentangled equal copies of $|\psi\rangle$ to deduce a k -partite product structure of the input state. One can require something even stronger on the input to achieve what we call *super product test*.

► **Lemma 3.** *The super product test on input $|\psi\rangle \otimes (|\phi_1\rangle \cdots |\phi_k\rangle)^{\otimes \ell}$ accepts with probability*

$$\frac{\ell}{(\ell + 1)} \cdot |\langle \psi | \phi_1 \rangle \cdots \langle \psi | \phi_k \rangle|^2 + \frac{1}{(\ell + 1)}.$$

This super product test focuses on determining whether a target state $|\psi\rangle$ is a product state or not. In addition to the target state, there are ℓ copies of an already k -partite product state that come to help. This test is very natural and simple, except it seems to ask too much of its inputs: To compare, the high-order product test requires some copies of a state whereas Lemma 3 requires some copies of an already k -partite product state of the form $|\phi_1\rangle \otimes \cdots \otimes |\phi_k\rangle$. We claim the super product test is not really asking for too much because our disentangler channel effectively “amplifies” the number of unentangled systems. In particular, we can rely on just two unentangled proofs to enforce a state close to $(|\phi_1\rangle \otimes \cdots \otimes |\phi_k\rangle)^{\otimes \ell}$ by Theorem 2. For simplicity, consider k unentangled pairs of unentangled proofs where the i^{th} pair applied Theorem 2 yields $|\phi_i\rangle^{\otimes \ell}$. Then run the super product test on a target state

² We don’t want to distract the readers by the issue about quantum states over real or complex numbers. In many cases, quantum computation over reals captures that over complex numbers. However, to the best of the authors’ knowledge, this is unclear in the context of $\text{QMA}(2)$. We have to use $\text{QMA}^{\mathbb{R}}(k)$ to denote the proof systems where the proofs are guaranteed to have real amplitudes.

$|\psi\rangle$ and the ℓ copies of already product states from our disentangler. Furthermore, note that it is very cheap to instead enforce a state close to $(|\phi_1\rangle \otimes \cdots \otimes |\phi_k\rangle)^{\otimes 2\ell}$, allowing us to reserve the extra ℓ copies of $|\phi_1\rangle \otimes \cdots \otimes |\phi_k\rangle$ as once the super product test passes, they can be used in any other computations as a very good proxy of $|\psi\rangle$. With this combination, we achieve arbitrarily good soundness without requiring more than $2k$ unentangled states³ while obtaining a guarantee about the output, rather than having just a single classical bit of output.

1.2 A Gap Amplification for $\text{QMA}^+(2)$ up to Criticality

Next, we turn to the unentangled quantum proofs, the so-called $\text{QMA}(2)$ class [23] and its variants. First, we provide some background on this subject.

The complexity of $\text{QMA}(2)$ was shown to be closely related to a variety of quantum and classical computational problems, e.g., determining if a mixed state is entangled given its classical description, as well as, various forms of classical polynomial/tensor optimization (see [17] for a more comprehensive list). Despite considerable interest and effort (e.g., [12, 1, 6, 4, 7, 14, 30, 27, 9, 8, 18]), we still only know the trivial complexity bounds $\text{QMA} \subseteq \text{QMA}(2) \subseteq \text{NEXP}$.

Even the fact that $\text{QMA}(2)$ admits strong gap amplification is non-trivial and remained open for about 10 years before the seminar work of Harrow and Montanaro [17]. With Theorem 2, it is easy to give a new proof of this fact.

A variant of $\text{QMA}(2)$, denoted $\text{QMA}^+(2)$, with proofs of nonnegative amplitudes was introduced by Jeronimo and Wu in [20]. The goal of this variant was to capture many properties of $\text{QMA}(2)$ while having more structure in order to obtain a greater understanding. Indeed, they showed that $\text{QMA}^+(2) = \text{NEXP}$ by designing a $\text{QMA}^+(2)$ protocol for a NEXP -complete problem with a constant gap. On the other end of their result is the observation that $\text{QMA}^+(2) \subseteq \text{QMA}(2)$ provided that the completeness-soundness gap of $\text{QMA}^+(2)$ is a sufficiently large constant. This makes $\text{QMA}^+(2)$ an intriguing class to study since either (i) showing that $\text{QMA}^+(2) = \text{QMA}(2)$, via possibly a gap amplification approach for $\text{QMA}^+(2)$, would characterize the complexity of $\text{QMA}(2)$, or (ii) showing $\text{QMA}^+(2) \neq \text{QMA}(2)$ would give a better upper bound $\text{QMA}(2) \subsetneq \text{NEXP}$.

By virtue of the unentanglement assumption of $\text{QMA}^+(2)$ and the product test [17], $\text{QMA}^+(2)$ admits some non-trivial gap amplification. For example, a gap of $1/\text{poly}(n)$ can be amplified to a constant gap in which the completeness becomes $1 - \exp(-\text{poly}(n))$ and the soundness becomes some constant strictly less than 1. Recently, Bassirian, Fefferman and Marwaha [3], building on [20], curiously showed that $\text{QMA}^+(1) = \text{NEXP}$ also with a constant gap.⁴ Since in the large constant gap regime of $\text{QMA}^+(1)$, we have $\text{QMA}^+(1) = \text{QMA} \subseteq \text{PP}$, their result rules out the strong gap amplification for $\text{QMA}^+(1)$ unless $\text{NEXP} \subseteq \text{PP}$. Moreover, it also suggests that strategies aimed at amplifying the gap for $\text{QMA}^+(2)$ must rely on the unentanglement assumption. This is precisely where the tools like the product test or our disentangler become essential.

With our disentangler, we make progresses towards understanding of $\text{QMA}^+(2)$ versus $\text{QMA}(2)$. In particular, our progresses can be summarized as two aspects with two motivating questions.

³ Naturally, the $2k$ unentangled states need to get larger in dimension to achieve better soundness.

⁴ It is not clear that their gap can be made as large as the one for $\text{QMA}^+(2) = \text{NEXP}$.

Motivating question 1. How crucial is the nonnegative amplitudes assumption to obtain $\text{QMA}^+(2) = \text{NEXP}$?

Regarding our first motivating question, we show that the nonnegative amplitudes assumption can be almost completely removed by considering unentangled quantum proofs of almost general *real* amplitudes. More precisely, we show that to capture NEXP it suffices to have unentangled proofs of the form $|\psi\rangle = \sqrt{a}|\psi_+\rangle + \sqrt{1-a}|\psi_-\rangle$ where $|\psi_+\rangle$ has nonnegative amplitudes, $|\psi_-\rangle$ only has negative amplitudes and $|a - (1 - a)| \geq 1/\text{poly}(n)$ with $a \in [0, 1]$. In words, we require the proofs to have slightly more ℓ_2 -probability mass ($1/\text{poly}(n)$ extra mass) either on nonnegative or negative amplitudes. We refer to the quantity $|a - (1 - a)|$ as the ℓ_2 -sign bias of $|\psi\rangle$. We call the associated complexity class almost- $\text{QMA}^{\mathbb{R}}(k)$. Our main complexity result can be stated as follows.

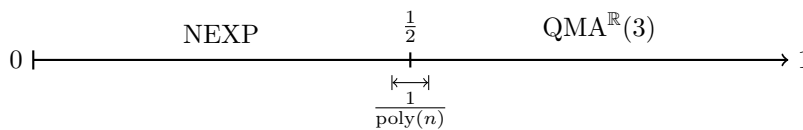
► **Theorem 4.** $\text{NEXP} = \text{almost-QMA}^{\mathbb{R}}(k)$ with unentangled proofs of ℓ_2 -sign bias of $\sqrt[5]{b(n)} \geq \text{poly}(1/n)$ and $k = \text{poly}(1/b(n))$.

We obtain the above result by investigating the other motivating question: Since the power of $\text{QMA}^+(k)$ ranges from NEXP to $\text{QMA}(k)$ depending on the gap,

Motivating question 2. How much can we amplify the gap of $\text{QMA}^+(k)$?

We make significant progress addressing this question. Specifically, we show that a even more relaxed version of $\text{QMA}^+(3)$, featuring a single proof with nonnegative amplitudes and the other two with general amplitudes, equals NEXP, with completeness $1 - \exp(-\text{poly}(n))$ and soundness $1/2 + 1/\text{poly}(n)$. At the first glance, this looks like a “just so so” gap amplification. It is noteworthy that achieving a slightly improved soundness of $1/2 - 1/\text{poly}(n)$ would imply $\text{QMA}^{\mathbb{R}}(3) = \text{NEXP}$. In particular, if $\text{QMA}^{\mathbb{R}}(3) \neq \text{NEXP}$, then there is a sharp phase transition in the complexity around the gap of a half.

► **Theorem 5.** $\text{NEXP} = \text{QMA}^+(3)$ with completeness $c = 1 - \exp(-\text{poly}(n))$ and soundness $s = 1/2 + 1/\text{poly}(n)$. Furthermore, we can assume a particular case of $\text{QMA}^+(3)$ in which two unentangled proofs have arbitrary amplitudes whereas only one unentangled proof has nonnegative amplitudes.



■ **Figure 1** Gap and the complexity regime of the particular version of $\text{QMA}^+(3)$ from Theorem 5. A gap below $1/2 - 1/\text{poly}(n)$ corresponds to NEXP, whereas a gap above $1/2 + 1/\text{poly}(n)$ corresponds to $\text{QMA}^{\mathbb{R}}(3)$, illustrating a sharp phase transition.

1.3 Organization

We introduce notations and review basic concepts and facts in Section 2. In Section 3, we present an efficient multipartite disentangler (like) channel from bipartite unentanglement. This construction relies on new de Finetti type properties concerning the interplay between

⁵ The letter n represents the input size and $b(n)$ is any polynomial time computable function bounded from below by a polynomial, i.e., by $1/n^c$ for some constant $c > 0$.

entanglement and symmetry which we explore in Section 4. In Section 5, we delve into the utility of our disentangler where we elaborate a generic framework in the context of property testing. As one example, we present a new proof that QMA(2) admits strong gap amplification. The final two sections are devoted to design new tests and derive the main complexity results in this paper. In Section 6, we present the super swap and super product test which leverage unentanglement to achieve much improved soundness than the well-known swap and product tests. Finally, we provide protocols for NEXP in Section 7 leading to the main complexity results of this paper, Theorem 4 and Theorem 5.

2 Preliminaries

General

As usual, $\mathbb{N}, \mathbb{R}, \mathbb{C}$ stand for the natural, real, and complex numbers, respectively. We adopt the Dirac notation for vectors representing quantum states, e.g., $|\psi\rangle, |\phi\rangle$, etc. In this paper, all the vectors of the form $|\psi\rangle$ are unit vectors. Given any pure state $|\psi\rangle$, we adopt the convention that its density operator is denoted by the Greek letter without the “ket”, e.g. $\psi = |\psi\rangle\langle\psi|$. The set of density operators in an arbitrary Hilbert space \mathcal{H} is denoted $\mathcal{D}(\mathcal{H})$. A symmetric state $|\psi\rangle \in (\mathbb{C}^d)^{\otimes n}$ is that invariant under any permutation $\pi \in \text{Sym}_n$ where Sym_n is the symmetric group. The action of π on $(\mathbb{C}^d)^{\otimes n}$ is

$$\pi : |\psi_1, \psi_2, \dots, \psi_n\rangle \mapsto |\psi_{\pi(1)}, \psi_{\pi(2)}, \dots, \psi_{\pi(n)}\rangle.$$

The *symmetric subspace* is the subspace of $(\mathbb{C}^d)^{\otimes n}$ that is invariant under Sym_n , denoted by $\vee^n(\mathbb{C}^d)$. Given any set $H \subseteq \mathcal{H}$ for some Hilbert space \mathcal{H} , $\text{conv}(H)$ is the convex hull of H .

One other particularly interesting set of states is the *separable* states. We adopt the following notation for the set of density operators regarding separable states,

$$\text{SEP}(d, r) := \text{conv}(\psi_1 \otimes \dots \otimes \psi_r \mid |\psi_1\rangle, \dots, |\psi_r\rangle \in \mathbb{C}^d).$$

A related notion is that of separable measurement, whose formal definition is given below.

► **Definition 6** (Separable measurement). *A measurement $M = (M_0, M_1)$ is separable if in the yes case, the corresponding positive semi-definite matrix M_1 can be represented as a conical combination of two operators acting on the first and second parts, i.e., for some distribution μ over the tensor product of positive semi-definite matrices α and β on the corresponding space,*

$$M_1 = \int \alpha \otimes \beta \, d\mu.$$

We record the following well-known fact. An interested reader is referred to [16] for a formal proof.

► **Fact 7** (Folklore). *The swap test is separable.*

Matrix Analysis

Given any matrix $M \in \mathbb{C}^{n \times n}$, M^\dagger is its conjugate transpose. Let $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$ denote its singular values. Then the trace norm $\|\cdot\|_1$, Frobenius norm $\|\cdot\|_F$ are defined as below

$$\|M\|_1 = \sum_i \sigma_i, \quad \|M\|_F = \sqrt{\sum_i \sigma_i^2}.$$

The Frobenius norm also equals the square root of sum of squared modulus of each entry, i.e., $\|M\|_F = \sqrt{\sum_{i,j} |M(i,j)|^2}$.

For a positive semi-definite (PSD) matrix M , $\|M\|_F = \sqrt{\text{Tr} M^2}$. For two PSD matrices, there is one (of many) analogous matrix Cauchy-Schwarz inequality.

$$\text{Tr}(\sigma\rho) \leq \|\sigma\|_F \cdot \|\rho\|_F. \quad (2.1)$$

We adopt the notation \succeq to denote the partial order that $\sigma \succeq \rho$ if $\sigma - \rho$ is positive semi-definite.

Distances between Quantum States

A standard notion of distance for quantum states is that of the *trace distance*. The trace distance between ψ and ϕ , denoted $D(\psi, \phi)$, is

$$\frac{1}{2} \|\psi - \phi\|_1 = \frac{1}{2} \text{Tr} \sqrt{(\psi - \phi)^\dagger (\psi - \phi)}. \quad (2.2)$$

We also use the notation $D(|\psi\rangle, |\phi\rangle)$ if we want to emphasize that ψ and ϕ are pure states. The following fact provides an alternative definition for trace distance between pure states.

► **Fact 8.** *The trace distance between $|\phi\rangle$ and $|\psi\rangle$ is given by $D(|\phi\rangle, |\psi\rangle) = \sqrt{1 - |\langle\phi|\psi\rangle|^2}$.*

Two states with small trace distance are indistinguishable to quantum protocols.

► **Fact 9.** *If a quantum protocol accepts a state ϕ with probability at most p , then it accepts ψ with probability at most $p + D(\phi, \psi)$.*

Trace distance enjoys the triangle inequality. For pure states, we can actually strengthen it.

▷ **Claim 10.** Given unit vectors $|\alpha\rangle, |\phi\rangle, |\beta\rangle \in \mathcal{H}$ for some Hilbert space \mathcal{H} . Suppose

$$|\langle\alpha|\phi\rangle|^2 = 1 - \varepsilon, \quad |\langle\beta|\phi\rangle|^2 = 1 - \delta.$$

Then for any $\varepsilon + \delta \leq 1$,⁶

$$|\langle\alpha|\beta\rangle|^2 \geq (\sqrt{(1-\varepsilon)(1-\delta)} - \sqrt{\varepsilon\delta})^2. \quad (2.3)$$

In general, we always have

$$|\langle\alpha|\beta\rangle|^2 \geq 1 - \varepsilon - \delta - 2\sqrt{\varepsilon\delta}. \quad (2.4)$$

Proof. Without loss of generality assume that

$$\begin{aligned} |\alpha\rangle &= \sqrt{1-\varepsilon}|\phi\rangle + \sqrt{\varepsilon}|\mu\rangle, \\ |\beta\rangle &= \sqrt{1-\delta}|\phi\rangle + \sigma\sqrt{\eta}|\mu\rangle + \sqrt{\delta-\eta}|\rho\rangle, \end{aligned}$$

where $|\mu\rangle, |\rho\rangle, |\phi\rangle$ are orthogonal, $0 \leq \eta \leq \delta$ and $\sigma \in \mathbb{C}$ is a relative phase. Using the basis $\{|\phi\rangle, |\mu\rangle, |\rho\rangle\}$, we can write down explicitly the density matrix of α and β :

$$\begin{aligned} \alpha &= \begin{pmatrix} 1-\varepsilon & \sqrt{\varepsilon(1-\varepsilon)} & 0 \\ \sqrt{\varepsilon(1-\varepsilon)} & \varepsilon & 0 \\ 0 & 0 & 0 \end{pmatrix}, \\ \beta &= \begin{pmatrix} 1-\delta & \sigma\sqrt{(1-\delta)\eta} & \sqrt{(1-\delta)(\delta-\eta)} \\ \sigma^*\sqrt{(1-\delta)\eta} & \eta & \sigma\sqrt{\eta(\delta-\eta)} \\ \sqrt{(1-\delta)(\delta-\eta)} & \sigma^*\sqrt{\eta(\delta-\eta)} & \delta-\eta \end{pmatrix}. \end{aligned}$$

⁶ When $\varepsilon + \delta > 1$, then $|\alpha\rangle$ and $|\beta\rangle$ in general can be orthogonal.

Now by definition,

$$\begin{aligned}
 D(|\alpha\rangle, |\beta\rangle)^2 &= \left(\frac{1}{2} \operatorname{Tr} \sqrt{(\alpha - \beta)^\dagger (\alpha - \beta)} \right)^2 \\
 &= \frac{1}{2} \|\alpha - \beta\|_F^2 \\
 &= \frac{1}{2} ((\varepsilon - \delta)^2 + (\varepsilon - \eta)^2 + (\delta - \eta)^2) + \eta(\delta - \eta) \\
 &\quad + |\sqrt{\varepsilon(1 - \varepsilon)} - \sigma\sqrt{(1 - \delta)\eta}|^2 + (1 - \delta)(\delta - \eta) \\
 &\leq \frac{1}{2} ((\varepsilon - \delta)^2 + (\varepsilon - \eta)^2 + (\delta - \eta)^2) + \eta(\delta - \eta) \\
 &\quad + (\sqrt{\varepsilon(1 - \varepsilon)} + \sqrt{(1 - \delta)\eta})^2 + (1 - \delta)(\delta - \eta), \tag{2.5}
 \end{aligned}$$

where the second step holds because $\alpha - \beta$ is Hermitian with trace 0 and rank 0 or 2. We claim that the RHS of (2.5), denote by f , is non-decreasing for $\eta \in [0, \delta]$. By routine calculation,

$$\begin{aligned}
 \frac{df}{d\eta} = -\varepsilon + \sqrt{\frac{\varepsilon}{\eta}(1 - \varepsilon)(1 - \delta)} \geq 0 &\iff (1 - \varepsilon)(1 - \delta) \geq \eta\varepsilon \\
 &\iff (1 - \varepsilon)(1 - \delta) \geq \delta\varepsilon \iff 1 \geq \varepsilon + \delta.
 \end{aligned}$$

As we assumed that $1 \geq \varepsilon + \delta$, $df/d\eta$ is always non-negative. Since the RHS of (2.5) is non-decreasing for $\eta \in [0, \delta]$, plug $\eta = \delta$ into the RHS of (2.5), we obtain

$$D(|\alpha\rangle, |\beta\rangle)^2 \leq (\varepsilon - \delta)^2 + (\sqrt{\varepsilon(1 - \varepsilon)} + \sqrt{(1 - \delta)\delta})^2,$$

In view of Fact 8, (2.3) is proved. The ‘‘in general’’ part is trivially true when $\varepsilon + \delta > 1$ and otherwise follows from (2.3). \triangleleft

Another widely used distance measure between quantum states is that of *fidelity*. For any density operators ρ, σ from the same Hilbert space,

$$F(\rho, \sigma) = \left(\operatorname{Tr} \sqrt{\sqrt{\rho}\sigma\sqrt{\rho}} \right)^2.$$

For our purposes, we only need the fact that when one of the two density operators corresponds to a pure state, then

$$F(\rho, \sigma) = \operatorname{Tr}(\rho\sigma).$$

The well-known data processing inequality for fidelity states that applying quantum operation never decreases the fidelity.

► **Fact 11.** *For any quantum channel (CPTP map) Φ ,*

$$F(\Phi(\rho), \Phi(\sigma)) \geq F(\rho, \sigma).$$

Schmidt Decomposition and Partial Trace

For $|\psi\rangle$ describing quantum states over two subsystems A, B , e.g., $|\psi\rangle \in \mathbb{C}^m \otimes \mathbb{C}^n$, there are two sets of orthonormal states $\{|\alpha_1\rangle, |\alpha_2\rangle, \dots, |\alpha_k\rangle\} \subseteq \mathbb{C}^m$, $\{|\beta_1\rangle, |\beta_2\rangle, \dots, |\beta_k\rangle\} \subseteq \mathbb{C}^n$, and positive numbers $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_k$ for some $k \leq \min\{n, m\}$ such that

$$|\psi\rangle = \sum_{i=1}^k \sqrt{\lambda_i} |\alpha_i\rangle |\beta_i\rangle, \quad \text{and} \quad \sum_{i=1}^k \lambda_i = 1. \tag{2.6}$$

The formula (2.6) is called the *Schmidt decomposition* of $|\psi\rangle$. The set of $\sqrt{\lambda_i}$ is unique, and is called the *Schmidt coefficient* of $|\psi\rangle$. We call $\sqrt{\lambda_1}$ the *top Schmidt coefficient* and $|\alpha_1\rangle|\beta_1\rangle$ the *top Schmidt component*. Note that the top Schmidt component may not be unique ignoring the global phases, in that case we break tie arbitrarily. Since Schmidt decomposition follows from singular value decomposition, the (top) Schmidt coefficients can also be formulated as some optimization problem.

▷ **Claim 12.** Given any state $|\psi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$. Then

$$\lambda_1 = \max_{|\sigma\rangle \in \mathcal{H}_1, |\rho\rangle \in \mathcal{H}_2} |\langle \psi | \sigma, \rho \rangle|^2$$

Often we want to study the density operator of a quantum state $|\psi\rangle$ over the subsystem A , mathematically described by tracing out B , denoted $\text{Tr}_B(\psi)$. We also abbreviate $\psi_A = \text{Tr}_B(\psi)$. Note that fidelity never increases under partial trace due to Fact 11, and similarly, the trace distance never increases under partial trace:

▶ **Fact 13.** For any quantum states ψ and ϕ over systems A and B ,

$$D(\psi, \phi) \geq D(\psi_A, \phi_A).$$

We use subscript to emphasize the systems that an operator is describing, e.g., ψ^{AB} simply means that ψ is a state over systems A and B .

Quantum Merlin-Arthur Systems

We now formally define the class almost-QMA^ℝ(k), but first we will need the ℓ_2 -sign bias definition, which, roughly speaking, quantifies the imbalance in ℓ_2 mass between the positive and negative amplitudes parts of a state.

▶ **Definition 14** (ℓ_2 -sign bias). Given $|\psi\rangle \in \mathbb{R}^n$, we can uniquely write it as $|\psi\rangle = \sqrt{a}|\psi_+\rangle + \sqrt{1-a}|\psi_-\rangle$, where $a \in [0, 1]$, $|\psi_+\rangle$ and $|\psi_-\rangle$ are unit vectors with only positive and negative amplitudes, respectively. The ℓ_2 -sign bias of $|\psi\rangle$ is defined as $|a - (1-a)|$.

Note that a non-negative amplitude state has ℓ_2 -sign bias of 1 whereas a general state has bias at least 0. Almost-QMA^ℝ(k) will be defined based on ℓ_2 -sign as a natural relaxation of QMA⁺(k) towards the general QMA(k).

▶ **Definition 15** (almost-QMA^ℝ(k)). Let $k: \mathbb{N} \rightarrow \mathbb{N}$ be a polynomial time computable function. A promise problem $\mathcal{L}_{\text{yes}}, \mathcal{L}_{\text{no}} \subseteq \{0, 1\}^*$ is in almost-QMA^ℝ(k) if there exists a BQP verifier V such that for every $n \in \mathbb{N}$ and every $x \in \{0, 1\}^n$,

- **Completeness:** If $x \in \mathcal{L}_{\text{yes}}$, then there exist unentangled states $|\psi_1\rangle, \dots, |\psi_{k(n)}\rangle$, each of ℓ_2 -sign bias $1/\text{poly}(n)$ and on at most $\text{poly}(n)$ qubits, s.t. $\Pr[V(x, |\psi_1\rangle \otimes \dots \otimes |\psi_{k(n)}\rangle) \text{ accepts}] \geq 9/10$.
- **Soundness:** If $x \in \mathcal{L}_{\text{no}}$, then for every unentangled states $|\psi_1\rangle, \dots, |\psi_{k(n)}\rangle$, each of each of ℓ_2 -sign bias $1/\text{poly}(n)$ and on at most $\text{poly}(n)$ qubits, we have $\Pr[V(x, |\psi_1\rangle \otimes \dots \otimes |\psi_{k(n)}\rangle) \text{ accepts}] \leq 1/10$.

3 The Disentangler from Unentanglement

In this section, we show how to obtain the dimension independent k -partite disentangler (like) channel from bi-partite unentanglement establishing Theorem 2. We will actually work mainly with a more refined procedure which we call quantum probably approximately product output (PAPO) procedure, from which the claimed disentangler can be easily constructed. We define PAPO as follows.

26:10 Dimension Independent Disentangler from Unentanglement and Applications

► **Definition 16** (PAPO). *Let $d, \ell, k \in \mathbb{N}$ and $\varepsilon, \delta \in [0, 1]$. A $(d, \ell, k, \varepsilon, \delta)$ -PAPO is a quantum procedure Λ satisfying:*

- **Completeness:** $\forall |\psi\rangle \in \mathbb{C}^d$, $\Lambda(\rho_1 \otimes \rho_2) = |\psi\rangle\langle\psi|^{\otimes k}$ where $\rho_1 = \rho_2 = |\psi\rangle\langle\psi|^{\otimes \ell}$,
- **Soundness:** $\forall \rho \in \text{SEP}(d^\ell, 2)$, with probability at least $1 - \delta$, $\Lambda(\rho)$ either rejects or outputs a state ε -close in trace distance to a separable state.

The main result in this section is an efficient PAPO procedure with parameter ℓ that is independent of the dimension d .

► **Theorem 17.** *For every $d, k \in \mathbb{N}$ and $\varepsilon, \delta \in [0, 1]$, there is an efficient $(d, \ell, k, \varepsilon, \delta)$ -PAPO with $\ell = O(k^3 \varepsilon^{-2} \delta^{-2} \log \delta^{-1})$.*

In Algorithm 1, we give a detailed description of our PAPO procedure. The procedure takes input two unentangled states, each over ℓ subsystems. We name the ℓ systems A_1, A_2, \dots, A_ℓ for the first state, and B_1, B_2, \dots, B_ℓ for the second state. The PAPO procedure is very simple, which we consider an advantage for such a fundamental task. It should be compared with the product test [17]: the PAPO procedure further takes advantage of symmetric subspace and that projection onto the symmetric subspace is efficient for quantum algorithms.

■ Algorithm 1 PAPO.

Input: $\rho^{A_1, A_2, \dots, A_\ell} \otimes \rho^{B_1, B_2, \dots, B_\ell} \in \text{SEP}(d^\ell, 2)$.

- Sample $\ell' \in [\ell - k]$ uniformly at random.
 - For $i = 1, \dots, \ell'$:
 1. Project $\rho^{A_i, \dots, A_\ell}$ onto the symmetric space.
 2. Project $\rho^{B_i, \dots, B_\ell}$ onto the symmetric space.
 3. If any of the projections fails: *Reject*.
 4. If $i \neq \ell'$, $\text{SwapTest}(\rho^{A_i}, \rho^{B_i})$.
 5. If the SwapTest fails: *Reject*.
 - Output $\rho^{A_{\ell'}, \dots, A_{\ell'+k-1}}$.
-

3.1 Analysis of PAPO

The efficiency of the protocol is trivial. Indeed projection onto the symmetric subspace can be implemented efficiently, see for example [2], and swap test is a special case of projection onto the symmetric subspace. So in the remainder of the section, we argue that our procedure satisfies the completeness and soundness criterion in Definition 16. We start with the following definition of *termination index*.

► **Definition 18** (Termination Index). *We set i^* to be the least element in $[\ell - k]$ such that either $\rho^{A_{i^*}, \dots, A_\ell}$ or $\rho^{B_{i^*}, \dots, B_\ell}$ is orthogonal to the symmetric subspace; we set $i^* = \infty$ if no such element exists.*

Here the “termination” means absolute termination (rejection) by projection into the symmetric subspace and has nothing to do with a particular execution of Algorithm 1. Most likely, projecting a general state into the symmetric subspace can success or fail. When a state can be successfully projected into the symmetric subspace with nonzero probability, then PAPO continues to run with nonzero probability. Such case is not counted as absolute termination. ⁷

⁷ Note that the swap test has no danger of absolute termination since it is always applied to separable states in Algorithm 1 and the swap test has soundness $1/2$. Thus in the definition of termination index, we don't worry about the swap test.

▷ **Claim 19.** The state $\rho^{A_i, \dots, A_\ell, B_i, \dots, B_\ell}$ at the i^{th} iteration of the for loop in Algorithm 1 is separable across $\rho^{A_i, \dots, A_\ell}$ and $\rho^{B_i, \dots, B_\ell}$.

Proof. Because the SwapTest is separable across A and B part given it accepts by Fact 7 and projection into the symmetric subspace for A and B part individually is also separable. Therefore $\rho^{A_i, \dots, A_\ell, B_i, \dots, B_\ell}$ is separable across A and B part. ◁

► **Definition 20** (Bad Index). *We say that an index $i \in [\ell]$ is η -bad*

1. *If $i \geq i^*$, (see Definition 18)*
2. *or if $\text{SwapTest}(\rho^{A_i}, \rho^{B_i})$ accepts with probability at most $1 - \eta$.*

▷ **Claim 21.** $\text{SwapTest}(\rho, \sigma)$ accepts with probability $\frac{1 + \text{Tr}(\rho\sigma)}{2} \leq \frac{3}{4} + \frac{\text{Tr}(\sigma^2)}{4}$.

Proof. Apply (2.1) for the density operators,

$$\text{Tr}(\rho\sigma) \leq \sqrt{\text{Tr}(\sigma^2) \cdot \text{Tr}(\rho^2)} \leq \frac{\text{Tr} \sigma^2 + \text{Tr} \rho^2}{2},$$

where the second step uses the AM-GM inequality. Note that $\text{Tr} \rho^2 \leq 1$, we are done. ◁

One more technical tool that we are going to need is the following, whose proof we defer to the next section.

► **Theorem 22.** *Given state $\sigma^{A_1 \dots A_k} \in \text{conv}(\sqrt{k}(\mathbb{C}^d))$. Then there is some distribution μ on pure states $|\phi\rangle \in \mathbb{C}^d$, such that*

$$\left\| \sigma - \int \phi^{\otimes k} d\mu \right\|_1 \leq O\left(\sqrt{k^3(1 - \text{Tr}(\sigma_{A_1})^2)}\right).$$

Proof of Theorem 17.

Completeness: For a desired output of $|\psi\rangle\langle\psi|^{\otimes k}$, we give two unentangled copies of $|\psi\rangle^{\otimes \ell}$ to Λ as input. In this case, Algorithm 1 indeed outputs $|\psi\rangle\langle\psi|^{\otimes k}$ w.p. 1.

Soundness: Let $\rho \in \text{SEP}(d^\ell, 2)$ be the input of Λ . Set

$$\eta = \varepsilon^2/k^3.$$

Due to Claim 19, $\rho^{A_{\ell'} \dots A_\ell, B_{\ell'} \dots B_\ell}$ is separable just before the ℓ'^{th} iteration (assuming successfully reaching this iteration). For ℓ' that is not a bad index, after projection onto the symmetric subspace, $\rho^{A_{\ell'} \dots A_{\ell'+k-1}} \in \text{conv}(\sqrt{k}(\mathbb{C}^d))$. It follows from Claim 21 that $\text{Tr}_{A_{\ell'}}(\rho^{A_{\ell'} \dots A_{\ell'+k-1}})^2 \geq 1 - 4\eta$. Thus we conclude that if ℓ' is not a bad index, then the output (if no rejection) is ε -close in trace distance to a convex combination of product states by Theorem 22 and our choice of parameter η . Therefore to prove the theorem, it suffices to bound the probability that Algorithm 1 outputs (not rejects) when ℓ' is a bad index.

Next we consider two cases. The first case: If the number of the η -bad indices among the first $\ell - k$ subsystems are less than $\delta(\ell - k)$, then with probability at least $1 - \delta$, the random index ℓ' is not η -bad. Therefore, Definition 16 is satisfied.

The second case: This fraction is larger than δ . Now conditioning on the event that ℓ' is a bad index, then ℓ' is a uniformly random bad index. Therefore, the chance that the set of indices $\{1, 2, \dots, \ell'\}$ contains less than $\delta/2$ fraction of bad indices is at most $\delta/2$. Thus with probability at least $1 - \delta/2$, we have seen at least $\delta/2 \cdot \delta(\ell - k) - 1$ bad indices in the execution of Algorithm 1 in the first ℓ' iterations. Since for each bad index the probability of not rejecting by the swap test is at most $1 - \eta$, the total probability of not rejecting is at most

$$(1 - \eta)^{\delta^2(\ell-k)-1} = \exp(-\Omega(\eta\delta^2\ell)) = \exp\left(-\Omega\left(\frac{\ell}{\varepsilon^{-2}\delta^{-2}k^3}\right)\right). \quad (3.1)$$

For $\ell = \Omega(k^3\varepsilon^{-2}\delta^{-2}\log\delta^{-1})$, we have $e^{-\eta\delta^2\ell} \leq \delta/2$. In this case, Definition 16 is also satisfied. \blacktriangleleft

3.2 The Disentangler from Unentanglement

We now construct our disentangler using the PAPO procedure, thereby proving Theorem 2 (restated below).

► **Theorem 2 (Disentangler from unentanglement).** *Let $d, \ell \geq k \in \mathbb{N}^+$. There is an efficient channel $\Lambda: (\mathbb{C}^d)^{\otimes \ell} \otimes (\mathbb{C}^d)^{\otimes \ell} \rightarrow (\mathbb{C}^d)^{\otimes k}$ such that for any density operators $\rho_1, \rho_2 \in \mathbb{C}^{d^\ell}$ there is a distribution μ on pure states $|\psi\rangle \in \mathbb{C}^d$ satisfying*

$$\left\| \Lambda(\rho_1 \otimes \rho_2) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \leq \tilde{O}\left(\left(\frac{k^3}{\ell}\right)^{1/4}\right).$$

Furthermore, product states of the form $\rho_1 = \rho_2 = |\psi\rangle\langle\psi|^{\otimes \ell}$ are mapped to $|\psi\rangle\langle\psi|^{\otimes k}$.

Proof. We set $\varepsilon = \delta$, whose exact values will be determined later. Let Λ_0 be the $(d, \ell, k, \varepsilon, \delta)$ -PAPO procedure guaranteed by Theorem 17. Suppose that we have an input state $\rho \in \text{SEP}(d^\ell, 2)$. The channel Λ will be defined as follows. Run the PAPO procedure Λ_0 on input ρ , then

1. If $\Lambda_0(\rho)$ succeeds, Λ outputs $\Lambda_0(\rho)$.
 2. Otherwise, Λ outputs a fixed product state say $|0\rangle\langle 0|^{\otimes k}$.
- If $\rho = \rho_1 \otimes \rho_2$ with $\rho_1 = \rho_2 = |\psi\rangle\langle\psi|^{\otimes \ell}$, then Λ outputs $|\psi\rangle\langle\psi|^{\otimes k}$ as desired. If the Λ_0 rejects, Λ outputs a product state. Therefore by the soundness of Λ_0 , firstly, with probability at least $1 - \delta$, Λ outputs a state σ which is ε -close to a mixture of product states, i.e., for some distribution μ on $\mathcal{D}(\mathbb{C}^d)$,

$$\left\| \sigma - \int_{|\psi\rangle} |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \leq \varepsilon;$$

and secondly, with probability $\leq \delta$, we output a state ρ_{error} . Overall, we have

$$\Lambda(\rho) = (1 - \delta')\sigma + \delta'\rho_{\text{error}}.$$

Therefore,

$$\begin{aligned} & \left\| \Lambda(\rho) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \\ &= \left\| (1 - \delta')\sigma + \delta'\rho_{\text{error}} - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \\ &\leq \left\| \sigma - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 + \|\delta'\sigma + \delta'\rho_{\text{error}}\|_1 \\ &\leq \varepsilon + 2\delta. \end{aligned}$$

In view of Theorem 17, for $\varepsilon = \delta$,

$$\left\| \Lambda(\rho) - \int |\psi\rangle\langle\psi|^{\otimes k} d\mu \right\|_1 \leq \tilde{O}\left(\left(\frac{k^3}{\ell}\right)^{1/4}\right),$$

concluding the proof. \blacktriangleleft

4 Quantum Slicing de Finetti Theorem

In this section, we prove Theorem 22. In spirit, it is a de Finetti type theorem with the constraint that there is little entanglement across some cut. We refer to such type of theorem as the slicing de Finetti theorem.

4.1 One-versus-Many Slicing de Finetti

To start, we study the following most basic scenario that a given permutation-invariant pure quantum state from $\vee^k(\mathbb{C}^d)$ has a large top Schmidt coefficient over cut between the first and the remaining subsystems. We obtain a dimension independent quantum de Finetti theorem under slicing constraints from first principles.

► **Theorem 23** (One-versus-many Slicing de Finetti). *Let $|\sigma\rangle^{A_1 \dots A_k} \in \vee^k(\mathbb{C}^d)$. If the largest Schmidt coefficient across the cut $A_1 : A_2 \dots A_k$ is at least $\sqrt{1 - \varepsilon}$, then*

$$\max_{|\phi\rangle \in \mathbb{C}^d} |\langle \sigma |^{A_1 \dots A_k} |\phi\rangle^{\otimes k}|^2 \geq 1 - 8k^3 \cdot \varepsilon.$$

To prove this theorem, we first establish the following duplicate lemma. It says that when a symmetric state $|\sigma\rangle$ is close to some product state $|\phi\rangle|\rho\rangle$, then you can find a new state close to $|\sigma\rangle$ that with two $|\phi\rangle$ and harms the closeness only mildly.

► **Lemma 24** (Duplicate Lemma). *Let $|\sigma\rangle \in \vee^k(\mathbb{C}^d)$. Consider some arbitrary decomposition of $\{A_1, A_2, \dots, A_k\} = \mathcal{A} \cup \mathcal{B} \cup \mathcal{C}$, such that $|\mathcal{A}| = |\mathcal{B}|$. Suppose $|\langle \sigma |^{\mathcal{A}\mathcal{B}\mathcal{C}} |\phi\rangle^{\mathcal{A}} |\rho\rangle^{\mathcal{B}\mathcal{C}}|^2 \geq 1 - \varepsilon$. Then, there is a state $|\zeta\rangle^{\mathcal{A}\mathcal{B}\mathcal{C}}$ such that $|\zeta\rangle = |\phi\rangle^{\mathcal{A}} |\phi\rangle^{\mathcal{B}} |\gamma\rangle^{\mathcal{C}}$ for some $|\gamma\rangle^{\mathcal{C}}$, and*

$$|\langle \sigma | \zeta \rangle|^2 \geq 1 - 8\varepsilon.$$

Furthermore, if $\rho_{\mathcal{C}}$ is a pure state, then $\gamma = \rho_{\mathcal{C}}$.

Proof. We assume that $\varepsilon < 1/8$, otherwise the statement is trivially true. Apply Schmidt decomposition to $|\rho\rangle^{\mathcal{B}\mathcal{C}}$ for the $\mathcal{B} : \mathcal{C}$ cut,

$$|\rho\rangle^{\mathcal{B}\mathcal{C}} = \sum_i \sqrt{\lambda_i} |\beta_i\rangle^{\mathcal{B}} |\gamma_i\rangle^{\mathcal{C}}.$$

Let

$$|\rho'\rangle^{\mathcal{A}\mathcal{C}} = \sum_i \sqrt{\lambda_i} |\beta_i\rangle^{\mathcal{A}} |\gamma_i\rangle^{\mathcal{C}}.$$

Since $|\sigma\rangle \in \vee^k(\mathbb{C}^d)$, we have

$$|\langle \sigma |^{\mathcal{A}\mathcal{B}\mathcal{C}} |\phi\rangle^{\mathcal{A}} |\rho\rangle^{\mathcal{B}\mathcal{C}}|^2 = |\langle \sigma |^{\mathcal{A}\mathcal{B}\mathcal{C}} |\phi\rangle^{\mathcal{B}} |\rho'\rangle^{\mathcal{A}\mathcal{C}}|^2 = 1 - \varepsilon.$$

By Claim 10,

$$(1 - 2\varepsilon)^2 \leq |\langle \phi |^{\mathcal{A}} \langle \rho |^{\mathcal{B}\mathcal{C}} |\phi\rangle^{\mathcal{B}} |\rho'\rangle^{\mathcal{A}\mathcal{C}}|^2 = \left(\sum_i \lambda_i |\langle \phi | \beta_i \rangle|^2 \right)^2. \quad (4.1)$$

Abbreviate $\eta_i = |\langle \phi | \beta_i \rangle|^2$. Note that

$$\sum \eta_i \leq 1, \quad \sum \lambda_i = 1.$$

26:14 Dimension Independent Disentangled from Unentanglement and Applications

Therefore, immediately from (4.1),

$$\lambda_1, \max \eta_i \geq 1 - 2\varepsilon, \quad (4.2)$$

which is at least $3/4$ since $\varepsilon < 1/8$. If $\eta_1 \neq \max \eta_i$, then

$$1 - 2\varepsilon \leq \sum_i \lambda_i \eta_i \leq \lambda_1(1 - \max \eta_i) + \max \eta_i \cdot (1 - \lambda_1) \leq 4\varepsilon,$$

which is impossible as $\varepsilon < 1/8$. Therefore, $\eta_1 \geq 1 - 2\varepsilon$.

We push it further,

$$\begin{aligned} 1 - 2\varepsilon &\leq \sum_i \lambda_i \eta_i \leq \lambda_1 \eta_1 + (1 - \lambda_1)(1 - \eta_1) = 2\lambda_1 \eta_1 - \lambda_1 - \eta_1 + 1 \\ &\leq 2\lambda_1 \eta_1 - 2\sqrt{\lambda_1 \eta_1} + 1 \\ &= 2 \left(\sqrt{\lambda_1 \eta_1} - \frac{1}{2} \right)^2 + \frac{1}{2}, \end{aligned}$$

where the second step is due to AM-GM inequality. Since $\lambda_1, \eta_1 > 3/4$, and $\varepsilon < 1/8$,

$$\lambda_1 \eta_1 \geq \left(\frac{1}{2} + \sqrt{\frac{1}{4} - \varepsilon} \right)^2 \geq 1 - 3\varepsilon,$$

where the last inequality holds for $\varepsilon \in [0, 1/8]$. Note that

$$|\langle \phi |^{\mathcal{A}} \langle \rho |^{\mathcal{BC}} | \phi \rangle^{\mathcal{A}} | \phi \rangle^{\mathcal{B}} | \gamma_1 \rangle^{\mathcal{C}}|^2 \geq \lambda_1 \eta_1 \geq 1 - 3\varepsilon.$$

By Claim 10 and that $1 - 3\varepsilon > 1/2$, it can be verified that

$$\begin{aligned} |\langle \sigma |^{\mathcal{ABC}} | \phi \rangle^{\mathcal{A}} | \phi \rangle^{\mathcal{B}} | \gamma_1 \rangle^{\mathcal{C}}|^2 &\geq (\sqrt{(1 - \varepsilon)(1 - 3\varepsilon)} - \sqrt{3\varepsilon})^2 \\ &= 1 - 4\varepsilon + 6\varepsilon^2 - 2\sqrt{3\varepsilon}\sqrt{(1 - \varepsilon)(1 - 3\varepsilon)} \\ &\geq 1 - 8\varepsilon. \end{aligned} \quad \blacktriangleleft$$

Now Theorem 23 is a simple consequence of Lemma 24: Duplicate the the first subsystem taken from the top Schmidt component of $|\sigma\rangle$.

Proof of Theorem 23. Let $|\sigma_0\rangle = |\phi\rangle|\gamma\rangle$ be the top Schmidt component of $|\sigma\rangle$ for the $A_1 : A_2 \dots A_k$ cut. By assumption of the theorem statement,

$$|\langle \sigma | \sigma_0 \rangle|^2 \geq 1 - \varepsilon.$$

Let $m = \lfloor \log k \rfloor, m^* = \lceil \log k \rceil$. For $i = 1, 2, \dots, m$, apply the Duplicate Lemma on $|\sigma_{i-1}\rangle$ with $\mathcal{A} = \{A_1, A_2, \dots, A_{2^{i-1}}\}, \mathcal{B} = \{A_{2^{i-1}+1}, A_{2^{i-1}+2}, \dots, A_{2^i}\}$. Let $|\sigma_i\rangle$ be the $|\zeta\rangle$ guaranteed by the Duplicate Lemma.

If $2^m < k$, apply the Duplicate Lemma one more time on $|\sigma_m\rangle$ with $\mathcal{A} = \{A_1, A_2, \dots, A_{k-2^m}\}, \mathcal{B} = \{A_{2^m+1}, A_{2^m+2}, \dots, A_k\}$, and let $|\sigma_{m^*}\rangle$ be the state guaranteed by the Duplicate Lemma. Then, a straightforward induction shows

1. $|\langle \sigma | \sigma_{m^*} \rangle|^2 \geq 1 - 8^{m^*} \varepsilon \geq 1 - 8k^3 \varepsilon,$
2. $|\sigma_{m^*}\rangle = |\phi\rangle^{\otimes k}.$

That finishes the proof. \blacktriangleleft

We make a remark about Theorem 23. Note that some polynomial dependence on k is unavoidable in this analysis for our procedure. Consider the following state:

$$\frac{1}{\sqrt{k+1}}|\vec{0}\rangle + \frac{1}{\sqrt{k+1}}\sum_{i=1}^k|\vec{e}_i\rangle.$$

To obtain a tight version of the above theorem with linear dependency on k is an interesting problem.

4.2 Many-versus-Many Slicing de Finetti

In Theorem 23, we considered top Schmidt coefficient being large on a 1 vs $k - 1$ cut for pure state. By looking at the example we mentioned in the end of the previous subsection, it is natural to think that if the top Schmidt coefficient is large among a balanced cut, then we can obtain better trace distance. That is indeed the case. In fact, that top Schmidt coefficient is large for a balanced cut always implies the top Schmidt coefficient is large for a less balanced cut for a symmetric state. In this subsection, our goal is to formalize this intuition.

► **Theorem 25** (Many-versus-many Slicing de Finetti). *Let $|\sigma\rangle^{A_1\dots A_k} \in \mathcal{V}^k(\mathbb{C}^d)$. Suppose for some $1 \leq \ell \leq k/2$, the top Schmidt coefficient of $|\sigma\rangle$ over the $A_1 \dots A_\ell : A_{\ell+1} \dots A_k$ cut is $\sqrt{1 - \varepsilon}$. Then there is $|\phi\rangle \in \mathbb{C}_d$, such that*

$$|\langle \sigma, \phi^{\otimes k} \rangle|^2 \geq 1 - O((k/\ell)^3 \varepsilon).$$

We start by collecting a couple of useful facts. The first one says that if a symmetric state from $(\mathbb{C}^d)^{\otimes k}$ is close to a product state, then it is also close to a symmetric product state, i.e., $|\phi^{\otimes k}\rangle$ for some $|\phi\rangle \in \mathbb{C}^d$.

► **Lemma 26.** *Given a symmetric state $|\sigma\rangle \in \mathcal{V}^k(\mathbb{C}^d)$ and a k -partite product state $|\psi\rangle \in (\mathbb{C}^d)^{\otimes k}$. Suppose $|\langle \psi | \sigma \rangle|^2 \geq 1 - \varepsilon$. Then there is $|\phi\rangle \in \mathbb{C}^d$ that satisfies*

$$|\langle \sigma | \phi^{\otimes k} \rangle|^2 \geq 1 - 9\varepsilon.$$

Proof. We take advantage of $|\sigma\rangle$ being symmetric in a way similar to that of Lemma 24. As $|\sigma\rangle \in \mathcal{V}^k(\mathbb{C}^d)$, we have for any permutation $\pi \in \text{Sym}_k$, $|\langle \sigma | \pi\psi \rangle|^2 \geq 1 - \varepsilon$. By Claim 10,

$$|\langle \psi | \pi\psi \rangle|^2 \geq 1 - 4\varepsilon.$$

Say $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \dots \otimes |\psi_k\rangle$, then,

$$\begin{aligned} (1 - 4\varepsilon)^{k!} &\leq \prod_{\pi \in \text{Sym}_k} |\langle \psi | \pi\psi \rangle|^2 = \left(\prod_{i \in [k]} \prod_{j \in [k]} |\langle \psi_i | \psi_j \rangle|^2 \right)^{(k-1)!} \\ &\leq \left(\mathbb{E}_{i \in [k]} \prod_{j \in [k]} |\langle \psi_i | \psi_j \rangle|^2 \right)^{k!}, \end{aligned} \tag{4.3}$$

where the last step uses the AM-GM inequality. It follows from (4.3), there must exist $i \in [k]$ such that

$$1 - 4\varepsilon \leq \prod_{j \in [k]} |\langle \psi_i | \psi_j \rangle|^2 \iff 1 - 4\varepsilon \leq |\langle \psi_i^{\otimes k} | \psi \rangle|^2.$$

Apply Claim 10 one more time, we obtain our lemma. ◀

26:16 Dimension Independent Disentglers from Unentanglement and Applications

The second fact due to Harrow and Montanaro [17, Appendix B Lemma 2] and Soleimanifar and Wright [31] establishes some criteria when a pure state is close to a product state.

► **Lemma 27.** *Given any quantum state $|\psi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_k$ for some arbitrary Hilbert space $\mathcal{H}_1, \dots, \mathcal{H}_k$. Suppose*

$$\mathbb{E}_{\mathcal{S} \subseteq [k]} [\text{Tr } \psi_{\mathcal{S}}^2] \geq 1 - \varepsilon.$$

Then for some product state $|\phi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle \otimes \cdots \otimes |\phi_k\rangle$,

$$|\langle \psi | \phi \rangle|^2 \geq 1 - 3\varepsilon.$$

Combining the above two lemmas, we obtain

► **Corollary 28.** *Given any state $|\sigma\rangle \in \mathbb{V}^k(\mathbb{C}^d)$. Suppose*

$$\mathbb{E}_{\mathcal{S} \subseteq [k]} [\text{Tr } \sigma_{\mathcal{S}}^2] \geq 1 - \varepsilon.$$

Then for some state $|\phi\rangle \in \mathbb{C}^d$,

$$|\langle \sigma | \phi^{\otimes k} \rangle|^2 \geq 1 - 27\varepsilon.$$

From the above discussion, to prove Theorem 25, it suffices to bound $\text{Tr } \psi_{\mathcal{S}}^2$ for any subset \mathcal{S} . The following “cut lemma” establishes such bounds.

► **Lemma 29 (Cut Lemma).** *Let $|\sigma\rangle \in \mathbb{V}^k(\mathbb{C}^d)$. Suppose for some $1 \leq \ell \leq k/2$, the top Schmidt coefficient of $|\sigma\rangle$ over the $A_1 \dots A_\ell : A_{\ell+1} \dots A_k$ cut is $\sqrt{1 - \varepsilon}$. Let $\mathcal{S} \subseteq [k]$ be some arbitrary subset. Then,*

$$\text{Tr } \sigma_{\mathcal{S}}^2 \geq \begin{cases} 1, & |\mathcal{S}| = 0; \\ 1 - 6\varepsilon, & \min\{|\mathcal{S}|, k - |\mathcal{S}|\} \in \{1, 2, \dots, \ell - 1\}; \\ 1 - O((|\mathcal{S}|/\ell)^3 \varepsilon), & \min\{|\mathcal{S}|, k - |\mathcal{S}|\} \in \{\ell, \dots, k/2\}. \end{cases}$$

Proof. For $\mathcal{S} = \emptyset$, the statement is trivial as σ is pure. Since $|\sigma\rangle \in \mathbb{V}^k(\mathbb{C}^d)$, without loss of generality, assume that $\mathcal{S} = \{1, 2, \dots, m\}$ for some $1 \leq m \leq k/2$. This is because $\text{Tr } \sigma_{\mathcal{S}}^2 = \text{Tr } \sigma_{\mathcal{S}^c}^2$ when σ is a pure state. Let $|\phi\rangle^{A_1 \dots A_\ell} |\zeta\rangle^{A_{\ell+1} \dots A_k}$ be the top Schmidt component associated with the coefficient $\sqrt{1 - \varepsilon}$.

Case 1. $m < \ell$. Let $\mathcal{A} = \{1, 2, \dots, m\}$, $\mathcal{B} = \{m + 1, \dots, \ell\}$, $\mathcal{C} = \{\ell + 1, \dots, k - \ell + m\}$, $\mathcal{D} = \{k - \ell + m + 1, \dots, k\}$. Write down the Schmidt decomposition of $|\phi\rangle$ over the \mathcal{A} and \mathcal{B} cut, $|\zeta\rangle$ over the \mathcal{C} and \mathcal{D} cut,

$$|\phi\rangle = \sum_i \sqrt{\lambda_i} |\alpha_i\rangle |\beta_i\rangle, \quad |\zeta\rangle = \sum_i \sqrt{\eta_i} |\gamma_i\rangle |\delta_i\rangle.$$

Since \mathcal{B} and \mathcal{D} has the same size, and that $|\sigma\rangle \in \mathbb{V}^k(\mathbb{C}^d)$, we have for the state $|\phi\rangle |\zeta\rangle$, if we switch the subsystem of \mathcal{B} and \mathcal{D} , then the overlap with $|\sigma\rangle$ is still $1 - \varepsilon$. Therefore, by Claim 10, we have

$$\begin{aligned}
(1 - 2\varepsilon)^2 &\leq \left| \left\langle \sum_{i,j} \sqrt{\lambda_i \eta_j} \langle \alpha_i |^{\mathcal{A}} \langle \beta_i |^{\mathcal{B}} \langle \gamma_j |^{\mathcal{C}} \langle \delta_j |^{\mathcal{D}}, \sum_{i,j} \sqrt{\lambda_i \eta_j} |\alpha_i\rangle^{\mathcal{A}} |\beta_i\rangle^{\mathcal{B}} |\gamma_j\rangle^{\mathcal{C}} |\delta_j\rangle^{\mathcal{D}} \right\rangle \right|^2 \\
&= \left(\sum_{i,j} \lambda_i \eta_j |\langle \beta_i | \delta_j \rangle|^2 \right)^2 \leq \lambda_1^2 \left(\sum_{i,j} \eta_j |\langle \beta_i | \delta_j \rangle|^2 \right)^2 \\
&\leq \lambda_1^2 \left(\sum_j \eta_j \sum_i |\langle \beta_i | \delta_j \rangle|^2 \right)^2 \leq \lambda_1^2.
\end{aligned}$$

Immediately,

$$\begin{aligned}
\text{Tr } \sigma_{\mathcal{A}}^2 &\geq (1 - \varepsilon)^2 \text{Tr}[(\text{Tr}_{\mathcal{BCD}}(\phi \otimes \zeta))^2] \\
&= (1 - \varepsilon)^2 \text{Tr} \left[\left(\text{Tr}_{\mathcal{BCD}} \left(\sum_i \lambda_i \alpha_i \otimes \beta_i \otimes \zeta \right) \right)^2 \right] \\
&\geq (1 - \varepsilon)^2 \lambda_1^2 \geq (1 - \varepsilon)^2 (1 - 2\varepsilon)^2 \\
&\geq 1 - 6\varepsilon.
\end{aligned} \tag{4.4}$$

The first step is true because $\sigma \succeq (1 - \varepsilon)\phi \otimes \zeta$, therefore $\text{Tr}_{\mathcal{BCD}} \sigma \succeq (1 - \varepsilon) \text{Tr}_{\mathcal{BCD}}(\phi \otimes \zeta)$ as partial trace is completely positive. It then follows that $\text{Tr } \sigma_{\mathcal{A}}^2 \succeq (1 - \varepsilon)^2 \text{Tr}(\text{Tr}_{\mathcal{BCD}}(\phi \otimes \zeta))^2$.

Case 2. $\ell < m \leq k/2$. We are much like the situation of Theorem 23. Let $t = \lceil \log(m/\ell) \rceil$. For $i = 1$ to t , we apply the Duplicate Lemma and obtain a state $|\sigma_i\rangle$, such that for $i = 1, 2, \dots, t$

$$\begin{aligned}
\text{Tr}_{\{\ell \cdot 2^{i+1}, \dots, k\}} \sigma_i &= \phi^{\otimes 2^i}, \\
\text{Tr}_{\{\ell \cdot 2^{i+1}, \dots, k\}} \sigma_i^2 &= 1,
\end{aligned} \tag{4.5}$$

$$|\langle \sigma | \sigma_i \rangle|^2 \geq 1 - 8^i \varepsilon. \tag{4.6}$$

By our choice of parameter, $2^{t-1}\ell < m \leq 2^t\ell$. If $m = 2^t\ell$, then $(\sigma_t)_{\mathcal{S}} = \text{Tr}_{\ell \cdot 2^{t+1}, \dots, k} \sigma_t$ is pure by (4.5). Then

$$\begin{aligned}
\sqrt{\text{Tr } \sigma_{\mathcal{S}}^2} &= \sqrt{\text{Tr } \sigma_{\mathcal{S}}^2 \cdot \text{Tr}(\sigma_t)_{\mathcal{S}}^2} \geq \text{Tr}(\sigma_{\mathcal{S}} \cdot (\sigma_t)_{\mathcal{S}}) = F(\sigma_{\mathcal{S}}, (\sigma_t)_{\mathcal{S}}) \\
&\geq F(\sigma, \sigma_t) = |\langle \sigma | \sigma_t \rangle|^2 \geq (1 - 8^{\log(m/\ell)}\varepsilon),
\end{aligned}$$

where the first step and third step are true because $(\sigma_t)_{\mathcal{S}}$ is pure; the second step uses (2.1); the fourth step is by Fact 11, the data processing inequality for fidelity; then fifth step is again by purity of the states; and the final step uses (4.6). It follows that

$$\text{Tr } \sigma_{\mathcal{S}}^2 \geq 1 - O((m/\ell)^3 \varepsilon).$$

If $m < 2^t\ell$, then we can apply Case 1. Let $\mathcal{A} = \{1, 2, \dots, 2^t\ell\}$, $\mathcal{B} = \{2^t\ell + 1, \dots, k\}$. Then in view of (4.6), the top Schmidt coefficient of $|\sigma\rangle$ among the $\mathcal{A} : \mathcal{B}$ cut is at least $\sqrt{1 - 8^t \varepsilon}$ by Claim 12. Thus by (4.4),

$$\text{Tr } \sigma_{\mathcal{S}}^2 \geq 1 - 6 \cdot 8^t \varepsilon \geq 1 - O((m/\ell)^3 \varepsilon). \quad \blacktriangleleft$$

Now Theorem 25 follows from Corollary 28 and Lemma 29.

4.3 Proof of Theorem 22

Now we record a version of the slicing de Finetti theorem for the mixture of symmetric states. A natural generalization of the top Schmidt coefficient among some $A : B$ cut for a state σ being large is that $\text{Tr } \sigma_A^2$ being large. In particular,

► **Lemma 30.** *Let $\sigma \in \mathbb{C}^n \otimes \mathbb{C}^m$ be some density operator, and A, B are the systems with respect to the space \mathbb{C}^n and \mathbb{C}^m , respectively. Suppose*

$$\text{Tr } \sigma_A^2 \geq 1 - \varepsilon.$$

Let μ be some distribution on pure states induced by σ , then

$$\mathbb{E}_{\rho \sim \mu} \lambda_1(\rho) \geq 1 - \varepsilon.$$

Proof. Let $m = |\text{supp } \mu|$ be a finite number, this is without loss of generality. Let $\rho_1, \rho_2, \dots, \rho_m$ be the pure states in $\text{supp } \mu$. Further, write the Schmidt decomposition for each ρ_i

$$|\rho_i\rangle = \sum_j \sqrt{\lambda_{ij}} |\phi_{ij}\rangle^A |\sigma_{ij}\rangle^B, \quad \lambda_{i1} \geq \lambda_{i2} \geq \dots.$$

Then

$$\sigma_A = \sum_i \mu(\rho_i) \sum_j \lambda_{ij} |\phi_{ij}\rangle \langle \phi_{ij}|.$$

Thus,

$$\begin{aligned} \text{Tr } \sigma_A^2 &= \sum_i \mu(\rho_i)^2 \sum_j \lambda_{ij}^2 + \sum_{i \neq i'} \mu(\rho_i) \mu(\rho_{i'}) \sum_{j, j'} \lambda_{ij} \lambda_{i'j'} |\langle \phi_{ij} | \phi_{i'j'} \rangle|^2 \\ &\leq \sum_i \mu(\rho_i)^2 \sum_j \lambda_{ij}^2 + \sum_{i \neq i'} \mu(\rho_i) \mu(\rho_{i'}) \lambda_{i1} \sum_{j, j'} \lambda_{i'j'} |\langle \phi_{ij} | \phi_{i'j'} \rangle|^2 \\ &\leq \sum_i \mu(\rho_i)^2 \sum_j \lambda_{ij}^2 + \sum_{i \neq i'} \mu(\rho_i) \mu(\rho_{i'}) \lambda_{i1} \sum_{j'} \lambda_{i'j'} \\ &\leq \sum_i \mu(\rho_i)^2 \sum_j \lambda_{ij}^2 + \sum_{i \neq i'} \mu(\rho_i) \mu(\rho_{i'}) \lambda_{i1} \\ &= \sum_i \mu(\rho_i)^2 \sum_j \lambda_{ij}^2 + \sum_i \mu(\rho_i) (1 - \mu(\rho_i)) \lambda_{i1} \\ &\leq \sum_i \mu(\rho_i)^2 \lambda_{i1} + \sum_i \mu(\rho_i) (1 - \mu(\rho_i)) \lambda_{i1} \\ &= \sum_i \mu(\rho_i) \lambda_{i1}, \end{aligned}$$

where the third step holds because for fixed i, i', j' , $\sum_j |\langle \phi_{ij} | \phi_{i'j'} \rangle|^2 \leq 1$. ◀

► **Theorem 31.** *Given density operator $\sigma^{A_1 \dots A_k}$ that describes states from $\text{conv}(\vee^k(\mathbb{C}^d))$. For any $1 \leq \ell \leq k/2$ and $\mathcal{A} = \{A_1, A_2, \dots, A_\ell\}$, there is some distribution μ on $|\phi\rangle \in \mathbb{C}^d$,*

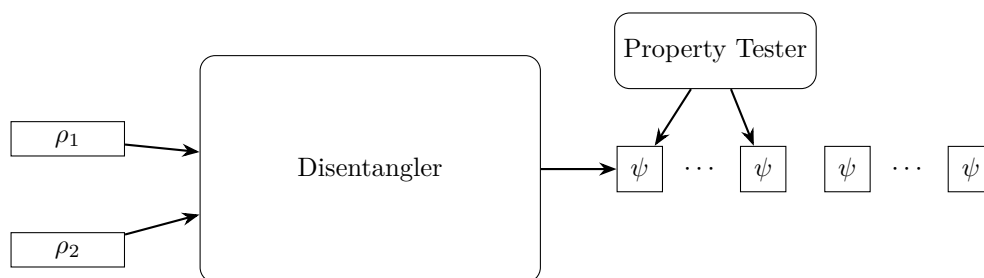
$$\left\| \sigma - \int |\phi\rangle \langle \phi|^{\otimes k} d\mu \right\|_1 \leq O\left(\sqrt{(k/\ell)^3 (1 - \text{Tr } \sigma_{\mathcal{A}}^2)}\right). \quad (4.7)$$

Proof. Let μ be the distribution on pure symmetric states induced by σ . Let $\text{Tr } \sigma_{\mathcal{A}}^2 = 1 - \varepsilon$. The theorem follows immediately by combining Fact 8, Lemma 30, Theorem 25, and triangle inequality. ◀

5 A Framework: Multiplexing Unentangled States for Property Testing

In this section, we present a general template illustrating the utility of our disentangler Theorem 2. We will then use this template multiple of times. Initially, we provide two examples as warm-ups for what is to come. Subsequently, in later sections, we apply this template with carefully designed testers to obtain new complexity results.

Our disentangler leverages a bipartite unentanglement assumption between two states of the form $\rho_1 \otimes \rho_2$ into an (approximate) multipartite unentanglement assumption of the form $\int |\psi\rangle\langle\psi|^{\otimes k} d\mu$. Having sufficiently many unentangled copies of a state ψ is particularly important in the context of quantum property testing as some properties require this assumption for testability. Indeed, many of other information processing tasks like quantum state tomography often assumes the input is of this form $|\psi\rangle\langle\psi|^{\otimes k}$. Moreover, multiple copies allow the tester to be executed multiple times amplifying its probability of distinguishing the closeness to the desired property. Finally, a property tester may end up destroying the copies $\psi^{\otimes k}$ when it measures this state, so it is desirable to have additional copies that can be used in further information processing tasks once the closeness to the desired property is certified. In Figure 2, we provide an illustration of a property tester being used in conjunction with our disentangler in order to obtain the aforementioned benefits.



■ **Figure 2** Schematic picture of our disentangler being used to (approximately) ensure multiple unentangled copies of a state as output. Part of these copies are used to test a given desired property. If the test passes, the remaining “certified” copies can be used in further information processing tasks.

Product Tester and Preparing Multipartite Separable States

To make this illustration more concrete, first we consider a scenario where the tester is the product test [17]. More precisely, the product test requires two unentangled copies of $|\psi\rangle \in \mathbb{C}^d$ and checks whether $|\psi\rangle$ is close to a product state of the form $|\phi_1\rangle \otimes \dots \otimes |\phi_s\rangle \in \mathbb{C}^{d_1} \otimes \dots \otimes \mathbb{C}^{d_s}$, where $d = d_1 \dots d_s$. For context, recall that (an abridged version of) their main result provides the following guarantees for this tester.

► **Theorem 32** (Product Test [17]). *Given $|\psi\rangle \in \mathbb{C}^{d_1} \otimes \dots \otimes \mathbb{C}^{d_s}$, let*

$$1 - \varepsilon = \max \left\{ |\langle\psi | \phi_1, \dots, \phi_s\rangle|^2 : |\phi_i\rangle \in \mathbb{C}^{d_i}, 1 \leq i \leq s \right\}.$$

Let $P_{test}(|\psi\rangle\langle\psi|)$ be the probability that the product test passes when applied to $|\psi\rangle$. Then, we have $P_{test}(|\psi\rangle\langle\psi|) = 1 - \Theta(\varepsilon)$.

Combining our disentangler from Theorem 2 and the product test from Theorem 32, we obtain the following corollary giving all the desired qualities alluded above in a more quantitative way.

► **Corollary 33.** Let $\mathcal{H} = \mathbb{C}^{d_1} \otimes \dots \otimes \mathbb{C}^{d_s}$. For every $k, k', \ell \in \mathbb{N}$ such that $\ell \geq k + 2k'$, there is a channel $\Gamma: \mathcal{D}(\mathcal{H}^{\otimes \ell} \otimes \mathcal{H}^{\otimes \ell}) \rightarrow \mathcal{D}(\mathcal{H}^{\otimes k} \otimes \mathbb{C}^2)$ such that for every $\rho_1, \rho_2 \in \mathcal{D}(\mathcal{H}^{\otimes \ell})$, there exists $\sigma \in \mathcal{D}(\mathcal{H}^{\otimes k} \otimes \mathbb{C}^2)$ defined as

$$\sigma = \int |\psi\rangle\langle\psi|^{\otimes k} \otimes \left(P_{\text{test}}(|\psi\rangle\langle\psi|)^{k'} |1\rangle\langle 1| + (1 - P_{\text{test}}(|\psi\rangle\langle\psi|)^{k'}) |0\rangle\langle 0| \right) d\mu,$$

such that

$$\|\Gamma(\rho_1 \otimes \rho_2) - \sigma\|_1 \leq \tilde{O} \left(\left(\frac{(k + 2k')^3}{\ell} \right)^{1/4} \right).$$

Furthermore, $\Gamma(\rho_1 \otimes \rho_2) = (|\psi\rangle\langle\psi|)^{\otimes k} \otimes |1\rangle\langle 1|$ provided $\rho_1 = \rho_2 = (|\psi\rangle\langle\psi|)^{\otimes \ell}$, where $|\psi\rangle = |\phi_1\rangle \otimes \dots \otimes |\phi_s\rangle$ for some $|\phi_i\rangle \in \mathbb{C}^{d_i}$ for $1 \leq i \leq s$.

Proof. Define another channel $\Gamma': \mathcal{D}(\mathcal{H}^{\otimes(k+2k')}) \rightarrow \mathcal{D}(\mathcal{H}^{\otimes k} \otimes \mathbb{C}^2)$ that takes as input the output of the disentangler Λ which is comprised of $k + 2k'$ registers of the space \mathcal{H} . We define the channel Γ' to act as identity on the first k registers. On the last $2k'$ registers it performs the product test on each pair of registers, outputting a single qubit $|1\rangle\langle 1|$ if all tests pass, otherwise outputting $|0\rangle\langle 0|$. Next we show $\Gamma = \Gamma' \circ \Lambda$, the composed channel, satisfies the statement.

Given general input $\rho_1 \otimes \rho_2$, by the guarantee of our disentangler, $\Lambda(\rho_1 \otimes \rho_2)$ satisfies

$$\left\| \Lambda(\rho_1 \otimes \rho_2) - \int |\psi\rangle\langle\psi|^{\otimes k+2k'} d\mu \right\|_1 \leq \tilde{O} \left(\left(\frac{(k + 2k')^3}{\ell} \right)^{1/4} \right).$$

Note that Γ' applied to $\int |\psi\rangle\langle\psi|^{\otimes k+2k'} d\mu$ results in

$$\int |\psi\rangle\langle\psi|^{\otimes k} \otimes \left(P_{\text{test}}(|\psi\rangle\langle\psi|)^{k'} |1\rangle\langle 1| + (1 - P_{\text{test}}(|\psi\rangle\langle\psi|)^{k'}) |0\rangle\langle 0| \right) d\mu. \quad (5.1)$$

Thus, the composed channel output $\Gamma(\rho_1 \otimes \rho_2)$ is $\tilde{O}(((k+2k')^3/\ell)^{1/4})$ close, in trace distance, to the state of (5.1).

The furthermore part is straightforward. Suppose that $|\psi\rangle = |\phi_1\rangle \otimes \dots \otimes |\phi_s\rangle$, where $|\phi_i\rangle \in \mathbb{C}^{d_i}$ for $1 \leq i \leq s$, and $\rho_1 = \rho_2 = (|\psi\rangle\langle\psi|)^{\otimes \ell}$. In this case, $\Lambda(\rho_1 \otimes \rho_2) = (|\psi\rangle\langle\psi|)^{\otimes k+2k'}$ and $\Gamma'(\Lambda(\rho_1 \otimes \rho_2)) = (|\psi\rangle\langle\psi|)^{\otimes k} \otimes |1\rangle\langle 1|$ since $|\psi\rangle$ is a product state and product test accepts with probability 1. ◀

QMA(2) Tester – Gap Amplification for QMA(2)

The gap amplification of QMA(2) was first proved in the seminar work of Harrow and Montanaro [17]. Using our template, we provide a conceptually more straightforward proof: Take the old QMA(2) protocol as the property tester in Figure 2.

► **Theorem 34.** Given a language $\mathcal{L} = (\mathcal{L}_{\text{yes}}, \mathcal{L}_{\text{no}})$. Suppose that $\mathcal{L} \in \text{QMA}(2)$ with completeness c and soundness s , where $c - s > 1/\text{poly}(n)$. Then, $\mathcal{L} \in \text{QMA}(2)$ with completeness $c' = 1 - \exp(-\text{poly}(n))$ and soundness $s' = 1/\text{poly}(n)$.

Proof. Let \mathcal{P} be the protocol for \mathcal{L} with the promised completeness c and soundness s . Therefore, for any fixed input x there is a measurement M acting on a space $\mathcal{H}^{\otimes 2}$ where $\mathcal{H} = \mathbb{C}^d$, such that,

$$\begin{aligned} \exists \sigma \otimes \rho \in \mathcal{D}(\mathcal{H}^{\otimes 2}), \text{Tr}(M(\sigma \otimes \rho)) &\geq c, & \text{if } x \in \mathcal{L}_{\text{yes}} \\ \forall \sigma \otimes \rho \in \mathcal{D}(\mathcal{H}^{\otimes 2}), \text{Tr}(M(\sigma \otimes \rho)) &\leq s, & \text{if } x \in \mathcal{L}_{\text{no}}. \end{aligned}$$

In the new protocol, choose $k = \text{poly}(n)/(c - s)^2$ and $\ell = \text{poly}(k)$ for some large enough polynomial. We ask for two proofs $|\rho_1\rangle, |\rho_2\rangle \in \mathcal{D}(\mathcal{H}'^{\otimes \ell})$, where $\mathcal{H}' = \mathbb{C}^2 \otimes \mathcal{H}$. In words, \mathcal{H}' is \mathcal{H} with one extra qubit. Apply the disentangler Λ from Theorem 2 on $\rho_1 \otimes \rho_2$, obtaining a separable state $\phi = \int d\mu |\psi\rangle\langle\psi|^{\otimes k}$, such that

$$\left\| \Lambda(\rho_1 \otimes \rho_2) - \int d\mu |\psi\rangle\langle\psi|^{\otimes k} \right\|_1 = \frac{1}{\text{poly}(n)}. \quad (5.2)$$

Consider the new measurement $M' = |01\rangle\langle 01| \otimes M$. We apply $M'^{\otimes(k/2)}$ to $\Lambda(\rho_1 \otimes \rho_2)$. Accept if more than $(c + s)/2$ fraction of the applications of M' accepts; reject otherwise. Next, we calculate the completeness and soundness of the new protocol.

Completeness. Suppose that $x \in \mathcal{L}_{\text{yes}}$, then the faithful prover will provide

$$|\rho_1\rangle = |\rho_2\rangle = \left(\frac{|0, \sigma\rangle + |1, \rho\rangle}{\sqrt{2}} \right)^{\otimes \ell}, \text{ and } \Lambda(\rho_1 \otimes \rho_2) = \left(\frac{|0, \sigma\rangle + |1, \rho\rangle}{\sqrt{2}} \right)^{\otimes k}.$$

Calculating the probability that M' accepts $(|0, \sigma\rangle + |1, \rho\rangle)^{\otimes 2}/2$,

$$\text{Tr} \left(M' \left(\frac{|0, \sigma\rangle + |1, \rho\rangle}{\sqrt{2}} \right)^{\otimes 2} \right) = \frac{1}{4} \text{Tr}(M(\sigma \otimes \rho)) \geq c/4.$$

By Chernoff bound, with probability at least $1 - \exp(-\Omega((c - s)^2 k)) = 1 - \exp(-\text{poly}(n))$, the new protocol accepts.

Soundness. Suppose that $x \in \mathcal{L}_{\text{no}}$. Calculating the probability that M' accepts $(\alpha|0, \sigma\rangle + \beta|1, \rho\rangle)^{\otimes 2}$ for arbitrary $\alpha, \beta \in \mathbb{C}$ and arbitrary $\sigma, \rho \in \mathcal{H}$ such that $|\alpha|^2 + |\beta|^2 = 1$,

$$\text{Tr}(M'(\alpha|0, \sigma\rangle + \beta|1, \rho\rangle)^{\otimes 2}) = |\alpha\beta|^2 \text{Tr}(M(\sigma \otimes \rho)) \leq s/4.$$

Therefore the probability to accept ϕ , an arbitrary convex combination of $|\psi\rangle^{\otimes k}$ is at most $\exp(-\Omega((c - s)^2 k))$ by Chernoff bound. Finally, by (5.2), the probability of accepting $\Lambda(\rho_1 \otimes \rho_2)$ is at most $1/\text{poly}(n)$. \blacktriangleleft

6 The Super Swap and Super Product Tests

In this section, we take another look at the product test as well as the swap test, considering one of the strongest possible generalization of the two.

We start with the more elementary swap test, which is a widely used to test if two quantum states, say $|\psi\rangle$ and $|\phi\rangle$, are equal. One fundamental limitation of the swap test is that it always accepts with probability at least $1/2$ even if the states are orthogonal. More precisely, its acceptance probability is $(1 + |\langle\psi|\phi\rangle|^2)/2$. Ideally, it would be much more useful to have a test with acceptance probability of $|\langle\psi|\phi\rangle|^2$, which is impossible with only one copy for each state. In the presence of many unentangled copies of $|\phi\rangle$ but just a single copy of $|\psi\rangle$, we show that it is possible to approach this goal with an arbitrarily small error overcoming the inherent limitation of the swap test. Therefore, we call this test the *super swap* test and we provide a description of it in Algorithm 2. In particular, this super swap test can be useful when it is difficult to produce a state $|\psi\rangle$, but much easier to produce copies of $|\phi\rangle$ and we want the tester's acceptance probability to more accurately capture how close $|\psi\rangle$ is to $|\phi\rangle$. In Section 7, the special state $|\psi\rangle$ will be a nonnegative amplitudes state which has a greater cost in the context of complexity protocols there, whereas $|\phi\rangle$ will have general amplitudes being a cheaper resource in that context.

The acceptance probability of the super swap test is established next.

26:22 Dimension Independent Disentangler from Unentanglement and Applications

■ **Algorithm 2** SuperSwap($|\psi\rangle, |\phi\rangle^{\otimes \ell}$).

Input: $|\psi\rangle, |\phi\rangle^{\otimes \ell}$.

1. Project $|\psi\rangle|\phi\rangle^{\otimes \ell}$ onto the symmetric space $\vee^{\ell+1}(\mathbb{C}^d)$.
 2. If the projection succeeds *accept*; else *reject*.
-

► **Lemma 35.** *The super swap test accepts with probability*

$$\frac{\ell \cdot |\langle \psi | \phi \rangle|^2}{\ell + 1} + \frac{1}{\ell + 1}.$$

Proof. Let $\Pi = (1/(\ell + 1)!) \sum_{\pi \in \text{Sym}_{\ell+1}} \pi$ be the projector onto $\vee^{\ell+1}(\mathbb{C}^d)$. Indeed, we have

$$\langle \psi | \langle \phi |^{\otimes \ell} \Pi | \psi \rangle | \phi \rangle^{\otimes \ell} = \frac{1}{\ell + 1} \langle \psi | \psi \rangle \langle \phi | \phi \rangle^\ell + \frac{\ell}{\ell + 1} |\langle \psi | \phi \rangle|^2 \langle \phi | \phi \rangle^{\ell-2},$$

concluding the proof. ◀

At first glance, it may seem inconvenient to assume multiple (ℓ -many) unentangled copies of $|\phi\rangle$. However, due to our disentangler channel, we can enforce a distribution over product states $|\phi\rangle^{\otimes \ell}$ by assuming only bipartite unentanglement.

Next we turn to the product test which checks whether a state is close to a k -partite product state [17]. It has a similar drawback to the usual swap test, namely, it always accepts with probability at least $1/2$ even if the state $|\psi\rangle$ is very far from product. As before, we will arbitrarily improve the soundness of the product test by having multiple unentangled copies. We call this new test the *super product test* and we describe it in Algorithm 3.

■ **Algorithm 3** SuperProduct($|\psi\rangle, (|\phi_1\rangle \dots |\phi_k\rangle)^{\otimes \ell}$).

Input: $|\psi\rangle, (|\phi_1\rangle \dots |\phi_k\rangle)^{\otimes \ell}$

1. Project $|\psi\rangle(|\phi_1\rangle \dots |\phi_k\rangle)^{\otimes \ell}$ onto the symmetric space $\vee^{\ell+1}((\mathbb{C}^d)^{\otimes k})$.
 2. If the projection succeeds *accept*; else *reject*.
-

► **Lemma 36.** *The super product test accepts with probability*

$$\frac{\ell}{(\ell + 1)} \cdot |\langle \psi | \phi_1 \rangle \dots \langle \phi_k \rangle|^2 + \frac{1}{(\ell + 1)}.$$

Proof. We view each copy of the state $|\phi_1\rangle \dots |\phi_k\rangle$ as a single state $|\phi\rangle$ and apply the super swap test to $|\psi\rangle$ and $|\phi\rangle^{\otimes \ell}$. The acceptance probability of the super product test now follows from Lemma 35. ◀

Analogously, it may seem inconvenient to assume multiple (ℓ -many) unentangled copies of $|\phi_1\rangle \dots |\phi_k\rangle$. However, that is not an issue by Corollary 33: We can enforce a distribution over product states $(|\phi_1\rangle \dots |\phi_k\rangle)^{\otimes \ell}$ by assuming only 2 unentangled states.

7 Gap Amplification for $\text{QMA}^+(k)$ up to Criticality and Almost- $\text{QMA}(k) = \text{NEXP}$

In the previous section, we described a very strong version of swap test and product test, noting that our disentangler channel has a good synergy with the new tests to overcome the drawbacks in their original versions. In this section, we put the tools in the context of quantum Merlin-Arthur games with unentangled provers, establishing our main complexity results Theorems 4 and 5.

7.1 Gap Amplification for $\text{QMA}^+(k)$ up to Criticality

The gap amplification for $\text{QMA}^+(k)$ is much less straightforward than $\text{QMA}(2)$. Indeed, a full gap amplification would imply $\text{QMA}(2) = \text{NEXP}$. To give our half gap amplification promised in Theorem 5, we start by showing how to simulate a $\text{QMA}^+(k)$ protocol \mathcal{P} given the following kinds of proofs:

1. one nonnegative-amplitudes proof $|\psi\rangle$;
2. abundant equal copies of an arbitrary proofs over reals $|\phi\rangle$.

Note we are relaxing k nonnegative-amplitudes proofs in a $\text{QMA}^+(k)$ protocol with only one nonnegative-amplitudes proof and general-amplitudes states. The motivation is, roughly, to remove as many nonnegative-amplitudes proofs in a $\text{QMA}^+(k)$ protocol as possible, so we get closer to a general $\text{QMA}(k)$ protocol.

We will check whether $|\phi\rangle^{\otimes k}$ is close to $|\psi\rangle$. Either they are close and then we can use the many copies of $|\phi\rangle^{\otimes k}$ to simulate \mathcal{P} , or else they are far apart and an application of the super product test can detect this condition. A description of this simulation procedure is given in Algorithm 4, which we denote as the symmetric simulator (since it assumes many equal copies of $|\phi\rangle$).

Algorithm 4 SymSimulator.

Input: $\text{QMA}^+(k)$ protocol \mathcal{P} , $|\psi\rangle = \sum_i \beta_i |i\rangle$: $\beta_i \geq 0$, $|\phi\rangle^{\otimes 2k\ell}$.

- If SuperProduct($|\psi\rangle$, $(|\phi\rangle^{\otimes k})^{\otimes \ell}$) fails, then *reject*.
 - For $i = 1, \dots, \ell$
 - Run the $\text{QMA}^+(k)$ protocol \mathcal{P} on a new copy of $|\phi\rangle^{\otimes k}$.
 - If protocol rejects, then *reject*.
 - *Accept*.
-

We now analyze the completeness and soundness of this simulation.

► **Lemma 37.** *Suppose \mathcal{P} is a $\text{QMA}^+(k)$ protocol with completeness c and soundness s . Let $p(n)$ be a non-decreasing function such that $p(n) \geq C_0$ for a sufficiently large constant $C_0 > 0$. If $\ell \geq 8p(n)^2 \ln(2)$ and $s \leq 1/8p(n)^2$, then SymSimulator has completeness c^ℓ and soundness at most $1/2 + 1/p(n)$.*

Proof. In the completeness case, we can assume that the proofs $|\phi\rangle$ have nonnegative amplitudes and $|\psi\rangle = |\phi\rangle^{\otimes k}$. Thus, SymSimulator accepts with probability at least c^ℓ .

Now, suppose that we are in the soundness case. Set $\varepsilon = |\langle \psi | \phi^{\otimes k} \rangle|^2$. By Lemma 3, the super product test accepts with probability

$$\left(\frac{\varepsilon \ell}{\ell + 1} + \frac{1}{\ell + 1} \right).$$

26:24 Dimension Independent Disentangler from Unentanglement and Applications

Since $\ell \geq 2p(n)$, if $\varepsilon < 1/2 + 1/2p(n)$, then the acceptance probability due to the super product test alone is at most $1/2 + 1/p(n)$ and we are done. Therefore, from now on, we assume that $\varepsilon \geq 1/2 + 1/2p(n)$.

Suppose $|\phi\rangle = \sum_i \alpha_i |i\rangle$, and let $|\phi_+\rangle = \sum_i |\alpha_i| |i\rangle$. Thus, $|\phi_+\rangle$ is a valid nonnegative-amplitudes state. Since $|\psi\rangle$ has nonnegative amplitudes by assumption, we should have

$$|\langle \psi | \phi_+^{\otimes k} \rangle|^2 \geq |\langle \psi | \phi^{\otimes k} \rangle|^2 = \varepsilon. \quad (7.1)$$

This is because the latter inner product incurs some cancellations due to negative values, which are avoided in the former inner product. (7.1) together with Claim 10 implies that

$$|\langle \phi^{\otimes k}, \phi_+^{\otimes k} \rangle|^2 \geq 2\varepsilon - 1.$$

Since we are assuming $\varepsilon > 1/2$, the trace distance between $|\phi\rangle^{\otimes k}$ and $|\phi_+\rangle^{\otimes k}$ can be bounded as below

$$D(\phi^{\otimes k}, \phi_+^{\otimes k}) \leq 2\sqrt{\varepsilon(1-\varepsilon)} \quad (7.2)$$

Note that \mathcal{P} accepts $|\phi_+^{\otimes k}\rangle$ with probability at most s by the soundness of \mathcal{P} . Therefore, each execution of the protocol \mathcal{P} on $|\phi\rangle^{\otimes k}$ accepts with probability, by Fact 9, at most

$$\min\{1, 2\sqrt{\varepsilon(1-\varepsilon)} + s\}.$$

The overall soundness of SymSimulator becomes

$$\left(\varepsilon \frac{\ell}{\ell+1} + \frac{1}{\ell+1} \right) \left(\min\{1, 2\sqrt{\varepsilon(1-\varepsilon)} + s\} \right)^\ell.$$

Now take $\varepsilon \geq 1/2 + 1/2p(n)$, and compute, we have

$$2\sqrt{\varepsilon(1-\varepsilon)} \leq 2\sqrt{\frac{1}{4} - \frac{1}{4p(n)^2}} \leq 1 - \frac{1}{2p(n)^2} + O\left(\frac{1}{p(n)^4}\right) \leq 1 - \frac{1}{4p(n)^2},$$

where the last inequality relies on $p(n) \geq C_0$ for a large enough constant $C_0 > 0$. Using that $s \leq 1/8p(n)^2$ and $\ell \geq 8p(n)^2 \ln(2)$, the final acceptance probability is

$$\left(2\sqrt{\varepsilon(1-\varepsilon)} + s \right)^\ell \leq \left(1 - \frac{1}{8p(n)^2} \right)^\ell \leq \frac{1}{2},$$

concluding the proof. ◀

To remove the symmetric assumption of having multiple identical copies of $|\phi\rangle$ in SymSimulator, we use the PAPO channel Λ and the PAPO channel takes just two unentangled proofs $|\phi'\rangle$ and $|\phi''\rangle$ (of arbitrary amplitudes) as its input. In other words, we now simulate a $\text{QMA}^+(k)$ protocol \mathcal{P} with:

- (i) one nonnegative-amplitudes proof $|\psi\rangle$;
- (ii) two general states $|\phi'\rangle, |\phi''\rangle$.

A formal description of the new simulation is given in Algorithm 5.

The analysis of Algorithm 5 is similar to that of Lemma 37. Therefore, instead of presenting an analysis of Algorithm 5 in isolation, we now apply this simulation for a $\text{QMA}^+(k)$ protocol \mathcal{P} that solves a NEXP-complete problem. In particular, we will need the following characterization of $\text{QMA}^+(2)$ from [20] as shown in the following theorem.

Algorithm 5 Simulator.

Input: QMA⁺(k) protocol \mathcal{P} , $|\psi\rangle = \sum_i \beta_i |i\rangle$: $\beta_i \geq 0$, $|\phi'\rangle, |\phi''\rangle$

- Let ρ be the output of our disentangler $\Lambda(\phi' \otimes \phi'')$ (i.e. Theorem 2).
 - If SymSimulator($\mathcal{P}, |\psi\rangle, \rho$) accepts, then *accept*; else *reject*.
-

► **Theorem 38** ([20]). QMA⁺(2) = NEXP.

Algorithm 5 gives rise to a protocol for NEXP that improves the above theorem in two aspects. First, the new protocol uses three unentangled proofs among which only one is required to have nonnegative amplitudes. Second, the completeness and soundness gap of this protocol is about 1/2. This seemingly mediocre gap is in fact a critical point, which we discuss in the next section.

► **Theorem 5.** NEXP = QMA⁺(3) with completeness $c = 1 - \exp(-\text{poly}(n))$ and soundness $s = 1/2 + 1/\text{poly}(n)$. Furthermore, we can assume a particular case of QMA⁺(3) in which two unentangled proofs have arbitrary amplitudes whereas only one unentangled proof has nonnegative amplitudes.

Proof. From Theorem 38, we apply the standard gap amplification by asking for more unentangled proofs to obtain a QMA⁺(k) protocol \mathcal{P} with completeness $c = 1 - \exp(-\text{poly}(n))$ and soundness $s = \exp(-\text{poly}(n))$, where $k = \text{poly}(n)$. Simulate \mathcal{P} using Algorithm 5. By Theorem 2, $\rho = \Lambda(\phi' \otimes \phi'')$ is $1/\text{poly}(n)$ -close to a convex combination of product states $\int |\phi\rangle\langle\phi|^{\otimes 2k\ell} d\mu$ with $\ell = \text{poly}(n)$. Invoking the symmetric simulator, by Lemma 37, the completeness becomes $c^\ell \geq 1 - \exp(-\text{poly}(n))$ and the soundness $1/2 + 1/\text{poly}(n)$ for a suitable choice of polynomial $\ell = \text{poly}(n)$. ◀

7.2 Almost-QMA^ℝ(k) = NEXP

Next, we show how to go from the nonnegative amplitudes assumptions to almost general amplitudes. Recall that the ℓ_2 -sign bias of a state $|\psi\rangle = \sqrt{a}|\psi_+\rangle + \sqrt{1-a}|\psi_-\rangle$, where $|\psi_+\rangle$ and $|\psi_-\rangle$ are the normalized nonnegative and negative amplitudes parts of $|\psi\rangle$, is defined as $|a - (1-a)|$ (see Definition 14).

► **Theorem 4.** NEXP = almost-QMA^ℝ(k) with unentangled proofs of ℓ_2 -sign bias of⁸ $b(n) \geq \text{poly}(1/n)$ and $k = \text{poly}(1/b(n))$.

Proof. We start with the QMA⁺(3) protocol from Theorem 5 with two general proofs $|\phi'\rangle, |\phi''\rangle$ and only one nonnegative proof $|\psi\rangle$. Let M be the verifier measurement. In the completeness case, we can assume that $|\psi\rangle$ has nonnegative amplitudes so we proceed to analyze the soundness case.

In the almost-QMA^ℝ(3) protocol, $|\psi\rangle$ will no-longer be assumed to have nonnegative amplitudes. Instead, we write $|\psi\rangle = \sqrt{a}|\psi_+\rangle + \sqrt{1-a}|\psi_-\rangle$, where $|\psi_+\rangle$ and $|\psi_-\rangle$ are its nonnegative- and negative-amplitudes normalized states. Without loss of generality, suppose that $a \geq 1/2$. Furthermore, under the ℓ_2 -sign bias assumption, we may assume that

$$a \geq 1/2 + \sqrt{100/p(n)}. \quad (7.3)$$

⁸ The letter n represents the input size and $b(n)$ is any polynomial time computable function bounded from below by a polynomial, i.e., by $1/n^c$ for some constant $c > 0$.

Let $|\phi'\rangle$ and $|\phi''\rangle$ be some quantum states (ignoring the ℓ_2 -bias requirement) as to be used in the simulation Algorithm 5. The combined proofs of the almost-QMA^R(3) protocol can be expressed as $|\xi\rangle = \sqrt{a}|\xi_0\rangle + \sqrt{1-a}|\xi_1\rangle$, where $|\xi_0\rangle = |\phi'\rangle \otimes |\phi''\rangle \otimes |\psi_+\rangle$ and $|\xi_1\rangle = |\phi'\rangle \otimes |\phi''\rangle \otimes |\psi_-\rangle$. Denote s the soundness of QMA⁺(3) protocol from Theorem 5. Then we can assume

$$s \leq 1/2 + 6/p(n). \quad (7.4)$$

Calculating the accepting probability of M on ξ ,

$$\begin{aligned} \langle \xi | M | \xi \rangle &= a \langle \xi_0 | M | \xi_0 \rangle + (1-a) \langle \xi_1 | M | \xi_1 \rangle \\ &\quad + \sqrt{a(1-a)} \langle \xi_0 | M | \xi_1 \rangle + \sqrt{a(1-a)} \langle \xi_1 | M | \xi_0 \rangle \\ &\leq s + \sqrt{a(1-a)} (\langle \xi_0 | M | \xi_0 \rangle + \langle \xi_1 | M | \xi_1 \rangle) \\ &\leq (1 + 2\sqrt{a(1-a)})s. \end{aligned} \quad (7.5)$$

where the first inequality follows from M being PSD, i.e., since $(\langle \xi_0 | - \langle \xi_1 |)M(|\xi_0\rangle - |\xi_1\rangle) \geq 0$ implies $\langle \xi_0 | M | \xi_0 \rangle + \langle \xi_1 | M | \xi_1 \rangle \geq \langle \xi_0 | M | \xi_1 \rangle + \langle \xi_1 | M | \xi_0 \rangle$. By (7.3) and (7.4), we have

$$(1 + 2\sqrt{a(1-a)})s \leq \left(2 - \frac{8}{p(n)}\right)s \leq \left(2 - \frac{8}{p(n)}\right) \left(\frac{1}{2} + \frac{1}{p(n)}\right) \leq 1 - \frac{2}{p(n)}.$$

Note that by a suitable choice of polynomial $p(n)$ and the initial completeness $c = 1 - \exp(-\text{poly}(n))$ of the QMA⁺(3) protocol of Theorem 5, we obtain a gap of $\Omega(1/p(n))$. To conclude the proof, we apply standard gap amplification using $k = \text{poly}(p(n))$ proofs in almost-QMA^R(k). ◀

We emphasize an important observation following from the above analysis: The “half” gap amplification in Theorem 5 is almost optimal. A larger gap in Theorem 5 by an additive term $1/\text{poly}(n)$ (e.g., if the soundness was at most $1/2 - 1/\text{poly}(n)$) would allow us to completely discard the ℓ_2 -sign bias assumption in Theorem 4, showing $\text{NEXP} = \text{QMA}^{\text{R}}(k)$. This can be easily seen in (7.5), when $s < 1/2 - 1/\text{poly}(n)$, the RHS will be at most $1 - 1/\text{poly}(n)$. It means that $s = 1/2 \pm 1/\text{poly}(n)$ in Theorem 5 is a critical point. In the case that $\text{QMA}(k)^{\text{R}} \neq \text{NEXP}$, there is a sharp phase transition.

References

- 1 Scott Aaronson, Salman Beigi, Andrew Drucker, Bill Fefferman, and Peter Shor. The power of unentanglement. In *Proceedings of the 23rd IEEE Conference on Computational Complexity (CCC)*, pages 223–236, 2008. doi:10.1109/CCC.2008.5.
- 2 Adriano Barenco, André Berthiaume, David Deutsch, Artur Ekert, Richard Jozsa, and Chiara Macchiavello. Stabilization of quantum computations by symmetrization. *SIAM Journal on Computing*, 26(5), 1997.
- 3 Roozbeh Bassirian, Bill Fefferman, and Kunal Marwaha. Quantum Merlin-Arthur and Proofs Without Relative Phase. In *Proceedings of the 15th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 287, pages 9:1–9:19, 2024. doi:10.4230/LIPIcs.ITCS.2024.9.
- 4 Salman Beigi. NP vs QMAlog(2). *Quantum Info. Comput.*, 2010. doi:10.5555/2011438.2011448.
- 5 J. S. Bell. On the einstein podolsky rosen paradox. *Physics Physique Fizika*, 1, November 1964. doi:10.1103/PhysicsPhysiqueFizika.1.195.
- 6 Hugue Blier and Alain Tapp. All languages in NP have very short quantum proofs. In *2009 Third International Conference on Quantum, Nano and Micro Technologies*, pages 34–37, 2009. doi:10.1109/icqnm.2009.21.

- 7 Fernando G. S. L. Brandão, Matthias Christandl, and Jon Yard. Faithful squashed entanglement. *Communications in Mathematical Physics*, 2011. doi:10.1007/s00220-011-1302-1.
- 8 Fernando G. S. L. Brandao and Aram W. Harrow. Estimating operator norms using covering nets, 2015. arXiv:1509.05065.
- 9 Fernando G.S.L. Brandão and Aram W. Harrow. Quantum de finetti theorems under local measurements with applications. In *Proceedings of the 45th ACM Symposium on Theory of Computing (STOC)*, 2013. doi:10.1145/2488608.2488718.
- 10 Matthias Christandl, Robert König, Graeme Mitchison, and Renato Renner. One-and-a-half quantum de finetti theorems. *Communications in mathematical physics*, 273(2):473–498, 2007.
- 11 John F. Clauser, Michael A. Horne, Abner Shimony, and Richard A. Holt. Proposed experiment to test local hidden-variable theories. *Phys. Rev. Lett.*, 23, October 1969. doi:10.1103/physrevlett.24.549.
- 12 Andrew C. Doherty, Pablo A. Parrilo, and Federico M. Spedalieri. Complete family of separability criteria. *Physical Review A*, 69, 2004. doi:10.1103/physreva.69.022308.
- 13 A. Einstein, B. Podolsky, and N. Rosen. Can quantum-mechanical description of physical reality be considered complete? *Phys. Rev.*, 47, May 1935. doi:10.1007/978-3-322-91080-6_6.
- 14 François Le Gall, Shota Nakagawa, and Harumichi Nishimura. On QMA protocols with two short quantum proofs. *Quantum Info. Comput.*, 2012. doi:10.26421/qic12.7-8-4.
- 15 Jeongwan Haah, Aram W. Harrow, Zhengfeng Ji, Xiaodi Wu, and Nengkun Yu. Sample-optimal tomography of quantum states. *IEEE Transactions on Information Theory*, 2017.
- 16 Aram W Harrow. The church of the symmetric subspace. *arXiv preprint*, 2013. arXiv:1308.6595.
- 17 Aram W. Harrow and Ashley Montanaro. Testing product states, quantum merlin-arthur games and tensor optimization. *J. ACM*, 60(1), February 2013. doi:10.1145/2432622.2432625.
- 18 Aram W. Harrow, Anand Natarajan, and Xiaodi Wu. An improved semidefinite programming hierarchy for testing entanglement. *Communications in Mathematical Physics*, 2017. doi:10.1007/s00220-017-2859-0.
- 19 Ryszard Horodecki, Paweł Horodecki, Michał Horodecki, and Karol Horodecki. Quantum entanglement. *Rev. Mod. Phys.*, 81:865–942, June 2009. doi:10.1103/RevModPhys.81.865.
- 20 Fernando Granha Jeronimo and Pei Wu. The Power of Unentangled Quantum Proofs with Non-negative Amplitudes. In *Proceedings of the 55th ACM Symposium on Theory of Computing (STOC)*, 2023.
- 21 Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. MIP*=RE, 2020. doi:10.1145/3485628.
- 22 Masaru Kada, Harumichi Nishimura, and Tomoyuki Yamakami. The efficiency of quantum identity testing of multiple states. *Journal of Physics A: Mathematical and Theoretical*, 41(39):395309, September 2008. doi:10.1088/1751-8113/41/39/395309.
- 23 Hirotada Kobayashi, Keiji Matsumoto, and Tomoyuki Yamakami. Quantum merlin-arthur proof systems: Are multiple merlins more helpful to arthur? In *Algorithms and Computation*, 2003. doi:10.1007/978-3-540-24587-2_21.
- 24 Robert König and Renato Renner. A de Finetti representation for finite symmetric quantum states. *Journal of Mathematical Physics*, 46(12), 2005.
- 25 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. doi:10.5555/1972505.
- 26 Ryan O’Donnell and John Wright. Efficient quantum tomography. In *Proceedings of the 48th ACM Symposium on Theory of Computing (STOC)*, 2016.
- 27 Attila Pereszlényi. Multi-prover quantum merlin-arthur proof systems with small gap, 2012. arXiv:1205.2761.
- 28 Renato Renner. Security of quantum key distribution. *International Journal of Quantum Information*, 2008.
- 29 Adrian She and Henry Yuen. Unitary property testing lower bounds by polynomials. In *Proceedings of the 14th Innovations in Theoretical Computer Science Conference (ITCS)*, 2023.

26:28 Dimension Independent Disentangled from Unentanglement and Applications

- 30 Yaoyun Shi and Xiaodi Wu. Epsilon-net method for optimizations over separable states. In *Proceedings of the 39th International Colloquium on Automata, Languages and Programming (ICALP)*, 2012. doi:10.1016/j.tcs.2015.03.031.
- 31 Mehdi Soleimanifar and John Wright. Testing matrix product states. In *Proceedings of the 33rd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1679–1701, 2022. doi:10.1137/1.9781611977073.68.
- 32 John Watrous. *The Theory of Quantum Information*. Cambridge University Press, 2018. doi:10.1017/9781316848142.

Baby PIH: Parameterized Inapproximability of Min CSP

Venkatesan Guruswami  

Simons Institute for the Theory of Computing, 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

Sai Sandeep  

Department of EECS, University of California, Berkeley, CA, USA

Abstract

The Parameterized Inapproximability Hypothesis (PIH) is the analog of the PCP theorem in the world of parameterized complexity. It asserts that no FPT algorithm can distinguish a satisfiable 2CSP instance from one which is only $(1 - \varepsilon)$ -satisfiable (where the parameter is the number of variables) for some constant $0 < \varepsilon < 1$.

We consider a minimization version of CSPs (Min-CSP), where one may assign r values to each variable, and the goal is to ensure that every constraint is satisfied by some choice among the $r \times r$ pairs of values assigned to its variables (call such a CSP instance r -list-satisfiable). We prove the following strong parameterized inapproximability for Min CSP: For every $r \geq 1$, it is W[1]-hard to tell if a 2CSP instance is satisfiable or is not even r -list-satisfiable. We refer to this statement as “Baby PIH”, following the recently proved Baby PCP Theorem (Barto and Kozik, 2021). Our proof adapts the combinatorial arguments underlying the Baby PCP theorem, overcoming some basic obstacles that arise in the parameterized setting. Furthermore, our reduction runs in time polynomially bounded in both the number of variables and the alphabet size, and thus implies the Baby PCP theorem as well.

2012 ACM Subject Classification Theory of computation \rightarrow Problems, reductions and completeness

Keywords and phrases Parameterized Inapproximability Hypothesis, Constraint Satisfaction Problems

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.27

Funding *Venkatesan Guruswami*: Research supported in part by NSF grants CCF-2228287 and CCF-2211972 and a Simons Investigator award.

Xuandi Ren: Supported in part by NSF CCF-2228287.

Sai Sandeep: Supported in part by NSF CCF-2228287.

1 Introduction

Approximation algorithms and fixed parameter-tractability (FPT) are two ways to cope with NP-hard problems. Recently, there have been many works that unite the two by obtaining approximation algorithms for NP-Hard problems that run in FPT time. Examples include VERTEX COLORING [13, 40], k -PATH DELETION [27], VERTEX CYCLE PACKING [35], FLOW TIME SCHEDULING [43], MAX k -VERTEX COVER in d -uniform hypergraphs [42, 38], k -MEANS and k -MEDIAN [28, 5, 23, 1, 7, 12]. On the other hand, there are also various developments in FPT hardness of approximation, for example, for k -BICLIQUE [29], k -CLIQUE [6, 31, 32, 24, 34, 8], k -SETCOVER [6, 10, 25, 30, 26, 33] and so on. We refer to the survey by Feldmann, Karthik, Lee, and Manurangsi [19] for a more comprehensive list of both FPT approximation algorithms and FPT hardness of approximation results.



© Venkatesan Guruswami, Xuandi Ren, and Sai Sandeep;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 27; pp. 27:1–27:17

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



However, it is worth pointing out that the techniques used in proving FPT hardness of approximation results have been rather problem-specific, and there remain other basic problems for which even constant inapproximability is unknown. This situation is due to the lack of a PCP-like theorem in the FPT world. In classical polynomial time hardness of approximation, the PCP theorem is a fundamental result that leads to a plethora of other results via reductions. The analog of the PCP theorem in the FPT regime was explicitly proposed by Lokshtanov et al [36] and named *Parameterized Inapproximability Hypothesis (PIH)*. The PIH states that there is an absolute constant $\varepsilon > 0$ such that for a 2CSP on k variables with alphabet size n , there is no algorithm running in time $f(k) \cdot n^{O(1)}$ that can distinguish a satisfiable instance from one where at most $(1 - \varepsilon)$ fraction of constraints can be simultaneously satisfied.

Analogous to the PCP theorem, PIH not only implies many FPT time inapproximability results including k -CLIQUE, k -SETCOVER, k -EXACTCOVER, and k -SETPACKING, but also unifies the inapproximability landscape by serving as a versatile problem-agnostic starting point. Previously, PIH was known to be implied by Gap-ETH [16, 11], a strong assumption with an inherent gap in it. Establishing PIH under a gap-free hypothesis has been a significant open problem in parameterized complexity. Very recently, following the posting of this work, PIH was established under ETH [22] using the technique of parallelization. The authors first reduce 3SAT to a vectorized problem called Vector-Valued CSP, and then design parallel probabilistically checkable proofs to verify its satisfiability. However, the proof in [22] fails to get any inapproximability of 2CSP under $W[1] \neq \text{FPT}$, since it is not known how to reduce a $W[1]$ -complete problem, say k -CLIQUE, to their Vector-Valued CSP. In fact, it was pointed out in [22] that Vector-Valued CSP is likely to be $M[1]$ -complete and thus not likely to be $W[1]$ -hard, where $M[1]$ is an intermediate complexity class between FPT and $W[1]$ [20, 9]. Proving PIH under $W[1] \neq \text{FPT}$, therefore, still remains an important open problem.

In this work, we prove a list version of the PIH, which we call *Baby PIH*, under the minimal hypothesis $W[1] \neq \text{FPT}$. In Baby PIH, we study r -list assignments to the underlying 2CSP instance. An r -list assignment has a list $L(u)$ of at most r values to each variable u , and a constraint between variables u, v is said to be satisfied by the r -list assignments if there is at least one pair of values $x \in L(u), y \in L(v)$ such that (x, y) satisfies the constraint between the variables u, v .

► **Definition 1 (Baby PIH).** *For all constants $r, C \geq 1$ and every computable function f , there is no algorithm running in time $f(k) \cdot n^C$ that can distinguish between the following cases, on input a 2CSP instance on k variables with alphabet size n .*

- (Completeness) *There is an assignment satisfying all the constraints.*
- (Soundness) *No r -list assignment satisfies all the constraints.*

The Baby PIH can be compared with the “Baby PCP theorem,” which was established via a purely combinatorial gap theorem in a remarkable recent work of Barto and Kozik [4] (who also coined the term Baby PCP). The Baby PCP theorem asserts the NP-completeness of distinguishing satisfiable instances of a 2CSP from those which lack a satisfying r -list assignment, for all constants $r > 1$. Baby PCP differs from Baby PIH in the sense that it concerns 2CSP on n variables with constant alphabet size. The concept of Baby PCP is itself not new and was studied in the early PCP days under the guise of a certain minimization version of Label Cover [2] (see also [17]).¹

¹ Label Cover is stronger than Baby PCP as stated above, as it imposes that the 2CSP relations are functions (usually referred to as projection property in PCP parlance). However, Barto and Kozik’s version also has this projection property.

The term “Baby” stems from the fact that after reducing the soundness parameter to an arbitrarily small positive constant via parallel repetition, one can deduce the Baby PCP theorem from the PCP theorem. This strategy also works in the PIH setting. Namely, we can use the canonical “clause-variable” construction to build a 2CSP instance with projection property, then apply parallel repetition [41] to amplify the soundness to below $\frac{1}{r^2}$. Suppose a 2CSP instance is r -list satisfiable, then by randomly picking a value from each variable’s list, we can conclude there is a single assignment that satisfies at least $\frac{1}{r^2}$ fraction of constraints. Therefore, being unable to approximate 2CSP within a $\frac{1}{r^2}$ factor implies that it is hard to distinguish whether an instance is satisfiable or not r -list satisfiable. In other words, PIH implies Baby PIH. Thus establishing the Baby PIH is a necessary step towards proving the PIH itself, and we also believe that it is a valuable intermediate step.

In this work, we prove Baby PIH under the minimal complexity assumption $W[1] \neq \text{FPT}$.

► **Theorem 2.** *Assuming $W[1] \neq \text{FPT}$, Baby PIH is true.*

Our proof of Theorem 2 is combinatorial and follows the framework of Barto and Kozik’s proof of the Baby PCP theorem. Specifically, given a 2CSP instance with variable set X and a list size r , we choose large enough integers a, b depending on r , and construct a bipartite direct product 2CSP instance, whose variable set is $\binom{X}{a} \cup \binom{X}{b}$. Given that the product instance is r -list satisfiable, we can repeatedly choose smaller integers $a' \leq a, b' \leq b$ and extract assignments for the instance with variable set $\binom{X}{a'} \cup \binom{X}{b'}$. The new assignments still list satisfy the smaller instance, but the size of each list on one side is decreased by 1, which helps us to do induction.

We highlight that although this proof strategy looks simple, there are basic obstacles to employ it in the Baby PIH setting (compared to the Baby PCP setting). In the PCP world, the alphabet size $|\Sigma|$ is at most some constant. Thus, to extract assignments for the smaller instance, it is affordable to pick a, b large enough depending on $|\Sigma|$. The running time of this reduction is therefore $|X|^{O_r, |\Sigma|^{(1)}}$. However in the PIH case, $|\Sigma|$ can be as large as the input size, and the running time of the reduction can only be $f(|X|) \cdot |\Sigma|^{O_r(1)}$ for some computable function f . To overcome this barrier, we non-trivially use the underlying structure of r -list satisfying assignments. We further note that our reduction runs in time $|X|^{O_r(1)}$, which is polynomial also in $|X|$. Therefore, our methods give a *unified proof of both the Baby PCP theorem and Baby PIH*.²

As we mentioned earlier, PIH implies Baby PIH, and thus our result can be viewed as a first step towards proving the former. An intermediate problem between them is the following average version of Baby PIH: let an r -average list assignment be a labeling $L(u)$ for each variable u such that the average cardinality of $L(u)$ over all the variables u is at most r .

► **Conjecture 3 (Average Baby PIH).** *For any constants $r > 1, C$ and any computable function f , no algorithm can given as input a 2CSP instance on k variables with size n , distinguish between the following two cases in $f(k) \cdot n^C$ time:*

- (Completeness) *There is an assignment satisfying all the constraints.*
- (Soundness) *No r -average list assignment satisfies all the constraints.*

² Barto and Kozik [4] derive Baby PCP with the stronger projection property. Using our techniques, we can get Baby PCP or Baby PIH with rectangular constraints, which is a slightly weaker property but still enough for many downstream reductions.

Note that the difference between the Baby and Average Baby versions is to use the ℓ_∞ vs. ℓ_1 norms of the number of values given to the variables. Once again, Average Baby PIH has a counterpart in the PCP world, namely the minimization version of Label Cover with ℓ_1 total cost, which fueled early inapproximability results for SETCOVER and basic problems that concern linear codes and lattices [2], as surveyed in [3].

Average Baby PIH is an intriguing open problem and could help in making further progress towards PIH. Furthermore, the Average Baby PIH is strong enough to imply some non-trivial inapproximability results. Notably, with an additional property called rectangular constraints (which we will define later), it implies constant inapproximability of k -EXACTCOVER, which we previously only knew under PIH [39].

Towards a better understanding of Average Baby PIH, we give a counterexample showing that the direct product construction we use to prove Baby PIH as is cannot establish the average version. This suggests in order to get Average Baby PIH or full PIH, one may need other techniques or constructions. As a candidate, we pose a question that whether the W[1]-hardness of approximating k -CLIQUE can help us bypass this counterexample. Please refer to Section 5 for details.

Organization. In Section 2, we introduce some preliminaries, including the problems considered in this paper and related complexity hypotheses. In Section 3, we prove Baby PIH via the direct product construction. We then discuss Average Baby PIH in Section 4, and conclude with some open problems in Section 5.

2 Preliminaries

We first start by formally defining 2-CSP.

► **Definition 4 (2CSP).** *An instance of arity-2 constraint satisfaction problem (2CSP) is a tuple $\Pi = (X, \Sigma, \Phi)$, where:*

- $X = \{x_1, \dots, x_k\}$ is the set of variables;
- $\Sigma = \{\Sigma_{x_1}, \dots, \Sigma_{x_k}\}$ is the set of their respective domains of values. We use $|\Sigma|$ to denote the maximum size of any domain, and call it the alphabet size of Π .
- $\Phi = \{\phi_1, \dots, \phi_m\}$ is the set of constraints. Each constraint ϕ_i is ordered tuple $\langle w_i, R_i \rangle$, where $w_i = (w_{i,1}, w_{i,2}) \in X^2$ is a pair of variables, and R_i is a 2-ary relation on $\Sigma_{w_{i,1}}$ and $\Sigma_{w_{i,2}}$.

An assignment of the 2CSP instance is a function from all variables in X to their respective domains. A constraint ϕ_i is said to be satisfied by an assignment σ if $(\sigma(w_{i,1}), \sigma(w_{i,2})) \in R_i$. We say an assignment σ satisfies the 2CSP instance if all constraints are satisfied by σ .

For each constraint ϕ_i , we can without loss of generality assume $w_{i,1} \neq w_{i,2}$, since unary constraints can be removed by restricting the domain of that variable.

We define r -list satisfiability, which generalizes the above satisfiability.

► **Definition 5 (r -List Satisfiability).** *Given a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, a multi-assignment is a function mapping each variable to a subset of its domain. We define the size of a multi-assignment σ as $\max_{x \in X} |\sigma(x)|$.*

We say a multi-assignment σ r -list satisfies Π if σ is of size at most r , and for every constraint ϕ_i , there exists a pair of values $u \in \sigma(w_{i,1})$ and $v \in \sigma(w_{i,2})$, such that $(u, v) \in R_i$.

Normal satisfiability can be viewed as 1-list satisfiability. Note that as r increases, it becomes easier to r -list satisfy a 2CSP instance.

The relations in the 2CSPs that we construct using the direct product (see Definition 7) satisfy a useful structural property, namely, rectangular relations.

► **Definition 6** (Rectangular Relation). *A relation $R \subseteq A \times B$ is said to be rectangular if there is a set C and functions $\pi : A \rightarrow C$ and $\sigma : B \rightarrow C$ such that $(a, b) \in R$ if and only if $\pi(a) = \sigma(b)$. Equivalently, R is rectangular if for all $a, a' \in A$ and $b, b' \in B$ such that $(a, b) \in R$, $(a, b') \in R$, and $(a', b) \in R$, we have $(a', b') \in R$.*

Rectangular relations can be informally viewed as consistency checks, and they are often satisfied by 2CSPs in product constructions. Projection relation, a stronger version of rectangular relation, is ubiquitous in PCP-based hardness reductions.

We now formally define the direct product construction that we use in our proof.

► **Definition 7** (Direct Product Construction). *Given a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, its t -wise direct product, denoted as $\Pi^{\odot t}$, is the following 2CSP instance (X', Σ', Φ') :*

- $X' = \binom{X}{t}$, where each variable is a t -sized subset of variables in Π .
- The domain of each variable $S \in X'$ is the set of all partial satisfying assignments for S in Π , i.e., all function σ that maps each $x \in S$ to its domain in Π , such that all constraints in Π induced by S are satisfied.
- Φ' has of a consistency constraint for each pair of distinct variables in X' . For $S, T \in X'$, the assignments σ_S, σ_T satisfy the constraint if and only if they are consistent on the values for variables in $S \cap T$.

Our results are based on the hypothesis $W[1] \neq \text{FPT}$, which is closely related to k -CLIQUE, a fundamental problem in parameterized complexity theory.

► **Definition 8** (k -CLIQUE). *An instance of (multicolored) k -CLIQUE problem is an undirected graph $G = (V = V_1 \dot{\cup} \dots \dot{\cup} V_k, E)$, where each V_i is an independent set. The goal is to decide whether we can find $v_1 \in V_1, \dots, v_k \in V_k$ which form a clique. For $r > 1$, the r -gap version of k -CLIQUE asks to distinguish between the following two cases:*

- (Yes) There exists $v_1 \in V_1, \dots, v_k \in V_k$ which form a clique.
- (No) The maximum clique in G has size at most k/r .

The $W[1] \neq \text{FPT}$ hypothesis states that for any computable function f , no algorithm can solve a k -CLIQUE instance with size n in $f(k) \cdot n^{O(1)}$ time. For a k -CLIQUE instance $G = (V = V_1 \dot{\cup} \dots \dot{\cup} V_k, E)$, we can build k variables x_1, \dots, x_k , letting V_i be the domain of variable x_i and making the edge set between V_i and V_j the constraint relation for x_i and x_j . Thus, G corresponds to a 2CSP instance with k variables and alphabet size at most n . It is easy to see that there is a clique of size k in G if and only if the 2CSP instance is satisfiable. Thus, we can restate the $W[1] \neq \text{FPT}$ hypothesis as follows.

► **Hypothesis 9** ($W[1] \neq \text{FPT}$). *For any computable function f , no algorithm can decide whether a given 2CSP instance $\Pi = (X, \Sigma, \Phi)$ is satisfiable in $f(|X|) \cdot |\Sigma|^{O(1)}$ time.*

The approximation version (with respect to the fraction of constraints that can be satisfiable) of $W[1] \neq \text{FPT}$ is called Parameterized Inapproximability Hypothesis (PIH).

► **Hypothesis 10** (Parameterized Inapproximability Hypothesis (PIH)[36]).³ *For any computable function f and some constant $\varepsilon > 0$, no algorithm can given as input a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, distinguish between the following two cases in $f(|X|) \cdot |\Sigma|^{O(1)}$ time:*

- (Completeness) Π is satisfiable.
- (Soundness) Any assignment of Π violates at least ε fraction of constraints.

³ Note that the original PIH in [36] states that constant approximating 2CSP parameterized by $|X|$ is $W[1]$ -hard. Here we use a relaxed form.

27:6 Baby PIH: Parameterized Inapproximability of Min CSP

We formally define k -SETCOVER, the parameterized version of the classical SETCOVER problem, and the exact version of it.

► **Definition 11** (k -SETCOVER, k -EXACTCOVER). *An instance of k -SETCOVER problem is a tuple $\Pi = (\mathcal{S}, U)$, where \mathcal{S} is a collection of subsets $\{S_1, \dots, S_n\}$ over the universe U , and the goal is to decide whether there are k sets in \mathcal{S} , whose union is U . For $r > 1$, the r -gap version of k -SETCOVER asks to distinguish between the following two cases:*

- (Yes) There are k sets whose union is U .
- (No) The union of any $r \cdot k$ sets is not U .

Furthermore, if in the yes case, the k sets are non-intersecting, i.e., they form a partition of U , then we also denote this gap problem as k -EXACTCOVER.

Finally, we define the (T, m) -set gadget, an important gadget used in reductions to k -SETCOVER and k -EXACTCOVER.

► **Definition 12** ((T, m) -Set Gadget). *A (T, m) -set gadget consists of a universe \mathcal{M} and some of its subsets C_1, \dots, C_m with the following property: Every collection of at most T sets out of $C_1, \overline{C_1}, C_2, \overline{C_2}, \dots, C_m, \overline{C_m}$ that is a set cover for \mathcal{M} must include both C_i and $\overline{C_i}$ for some i .*

It was proved in [37] that a (T, m) -set gadget can be constructed efficiently:

► **Lemma 13.** *There is an algorithm that given any T, m , runs in time $\text{poly}(m, 2^T)$ and outputs a (T, m) -set gadget with universe size $\text{poly}(m, 2^T)$.*

3 Baby PIH

In this section, we analyze the direct product construction to prove Baby PIH under $W[1] \neq \text{FPT}$.

► **Theorem 14 (Main).** *For any integer $r > 1$, there is an integer $t > 0$ such that for any 2CSP instance $\Pi = (X, \Sigma, \Phi)$ and its t -wise direct product instance $\Pi^{\odot t} = (X', \Sigma', \Phi')$:*

- (Completeness) *If Π is satisfiable, then $\Pi^{\odot t}$ is satisfiable as well.*
- (Soundness) *If Π is not satisfiable, then $\Pi^{\odot t}$ is not r -list satisfiable.*

Since t is a constant depending solely on r , the number of variables in the new instance $|X'| = \binom{|X|}{t}$ depends only on $|X|$ rather than $|\Sigma|$, and the alphabet size of the new instance $|\Sigma'| \leq |\Sigma|^t$ is polynomial in $|\Sigma|$. Thus for any constant C and any computable function f , $f(|X'|) \cdot |\Sigma'|^C$ time is upper bounded by $g(|X|) \cdot |\Sigma|^{C \cdot t}$ time for some computable function g . Therefore, we have the following corollary from Theorem 14:

► **Corollary 15.** *Assuming $W[1] \neq \text{FPT}$, Baby PIH is true.*

Note that the completeness in Theorem 14 follows trivially by assigning the restriction of the satisfying assignment on X to each subset of variables. The main challenge is to show that when $\Pi^{\odot t}$ has a r -list satisfying assignment, Π is satisfiable. To prove this, we will first work on a bipartite version of the direct product of 2CSP.

► **Definition 16** (Bipartite Direct Product Construction). *Given a 2CSP instance $\Pi = (X, \Sigma, \Phi)$ and positive integers a, b , the (a, b) -bipartite direct product 2CSP instance $\Pi^{\odot(a,b)} = (X', \Sigma', \Phi')$ is constructed as follows.*

- *The variable set X' consists of all a -sized subsets of X on the left side, and all b -sized subsets of X on the right side. With a little abuse of notation, we have $X' = \binom{X}{a} \cup \binom{X}{b}$.*

- The domain of each variable $S \in X'$ is the set of all partial satisfying assignments for S in Π .
- For every $S \in \binom{X}{a}$ and $T \in \binom{X}{b}$, we have a constraint in Φ' that checks whether σ_S and σ_T are consistent on the values for variables in $S \cap T$.

If for a 2CSP instance Π , $\Pi^{\odot t}$ is r -list satisfiable for $t = \max(a, b)$, by taking restrictions of its assignments on the smaller sets, it is easy to see $\Pi^{\odot(a,b)}$ is r -list satisfiable as well. Thus, our goal is to show that if the bipartite instance $\Pi^{\odot(a,b)}$ is r -list satisfiable, then the original instance Π is satisfiable.

Our proof idea is borrowed from the Baby PCP theorem recently proved by Barto and Kozik [4]. However, their theorem crucially relies on the fact that the alphabet $|\Sigma|$ is as small as a constant, which helps them to extract satisfying assignments for the smaller instance. The running time of their reduction is, therefore, $|X|^{O(|\Sigma|)}$, which is not affordable here since $|\Sigma|$ is as large as the input size. We resolve this issue by making use of the structural properties of the assignments in the direct product construction arising from the fact that they satisfy r -list consistency. If we fix some set S on one side and consider its r assignments, each set on the other side that intersects S must have one of the r , which is a constant, assignments for their intersection part. We use this simple yet very useful observation when extracting the assignments in the inductive proof.

In the following, we first prove Lemma 17, which is crucial to extract list satisfying assignments for the smaller subsets. Then in Lemma 18, we analyze a special case of the bipartite direct product construction when each variable on the right (bigger) side has only one assignment, and the consistency requirement is a slightly weaker one. In Lemma 19, we finish the analysis of the bipartite direct product construction, from which we get Theorem 14 as a corollary.

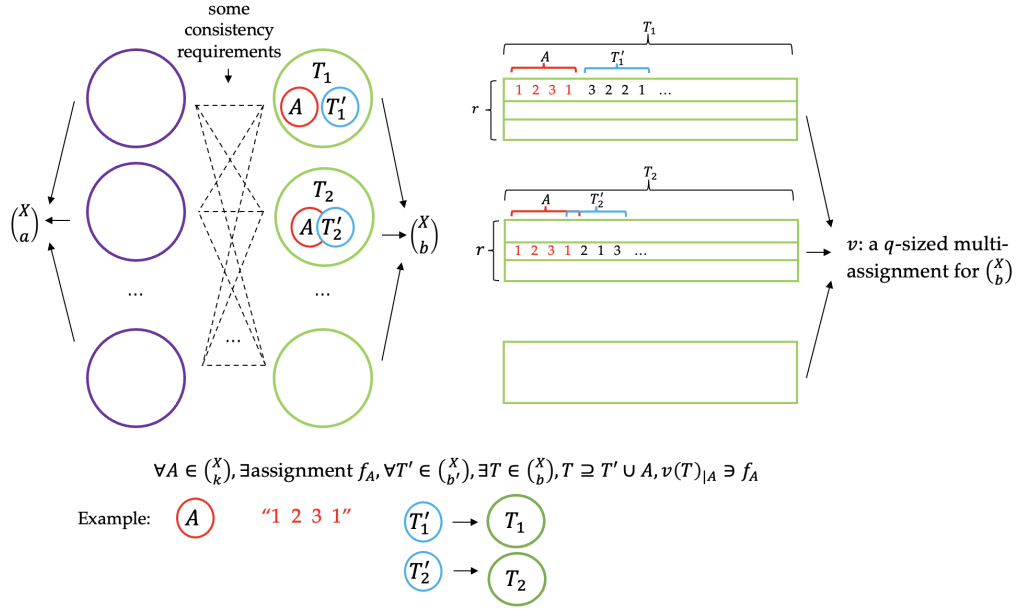
► **Lemma 17.** *Let k, r, q, a, b, b' be integers satisfying $r, q > 0, a \geq k, b \geq r \cdot b' + a$. Consider the (a, b) -bipartite direct product 2CSP instance based on $\Pi = (X, \Sigma, \Phi)$. Let u be an r -sized multi-assignment for $\binom{X}{a}$ and v be a q -sized multi-assignment for $\binom{X}{b}$. Suppose for every $S \in \binom{X}{a}, T \in \binom{X}{b}$ with $T \supseteq S$, $v(T)|_S \cap u(S) \neq \emptyset$. Then for every $A \in \binom{X}{k}$, there is an assignment f_A for A such that for every $T' \in \binom{X}{b'}$, there is some $T \in \binom{X}{b}$ satisfying $T \supseteq T' \cup A$ and $v(T)|_A \ni f_A$.*

Proof. Suppose for the sake of contradiction that there is no such f_A for some set $A \in \binom{X}{k}$. In other words, for any assignment f on A , there exists a set $T'_f \in \binom{X}{b'}$, such that for every $T \in \binom{X}{b}$ satisfying $T \supseteq T'_f \cup A$, $v(T)|_A \not\ni f$.

Pick an arbitrary $S \in \binom{X}{a}$ with $S \supseteq A$. This can be done since $a \geq k$. Consider any set $T \in \binom{X}{b}$ which contains $\cup_{f \in u(S)|_A} (T'_f \cup S)$. Such a T exists since $b \geq r \cdot b' + a$. By the assumption about A , $v(T)|_A$ does not contain any value in $u(S)|_A$. This contradicts the consistency guarantee in the hypothesis of the lemma, namely that for any $T \in \binom{X}{b}$ with $T \supseteq S$, $v(T)|_S$ must contain some value in $u(S)$. ◀

See Figure 1 for an illustration of Lemma 17.

► **Lemma 18.** *For any integer $r > 0$, let $a = r$ and $b = (2r)^r$. Consider the (a, b) -bipartite direct product 2CSP instance based on $\Pi = (X, \Sigma, \Phi)$. Let u be an r -sized multi-assignment for $\binom{X}{a}$ and v be a 1-sized assignment for $\binom{X}{b}$. Suppose for every $S \in \binom{X}{a}$ and $T \in \binom{X}{b}$ with $S \subseteq T$, $v(T)|_S \in u(S)$. Then there is a global satisfying assignment σ to the 2CSP instance Π .*



■ **Figure 1** An illustration of Lemma 17.

Proof. We apply induction on r . When $r = 1$, we have $a = 1, b = 2$, and both u, v are 1-sized assignments. We claim that u is the desired global satisfying assignment of Π . For each constraint on $(x, y) \in X^2$, $v(\{x, y\}) = (x \mapsto u(x), y \mapsto u(y))$ by our consistency guarantee. By the construction of bipartite direct product 2CSP instance, the domain for each $T \in \binom{X_b}{b}$ consists only of partial satisfying assignments. Thus the fact that $(x \mapsto u(x), y \mapsto u(y))$ lies in the domain of $T = \{x, y\}$ implies u satisfies the constraint on (x, y) .

When $r > 1$, the idea is to extract consistent assignments for the (a', b') -bipartite direct product 2CSP instance and to decrease r , for some $a' \leq a, b' \leq b$. At a high level, if for some $x \in X$, every set in $\binom{X_a}{a}$ has at least two different values for it under u , we can keep only one of them and peel the other off to decrease r by 1; otherwise we can prove the unique assignments are already satisfying.

In the following, define $k = 1$, and $a' = a - 1, b' = (2(r - 1))^{r-1}$, i.e., a', b' are parameters with respect to $r - 1$. It's easy to see $k \leq a$ and $r \cdot b' + a \leq r \cdot (2r)^{r-1} + r \leq (2r)^r = b$. According to Lemma 17, for every $A = \{x_i\} \in \binom{X_k}{k}$, there is an assignment f_A for A such that for every $T' \in \binom{X_{b'}}{b'}$, there is some $T \in \binom{X_b}{b}$ satisfying $T \supseteq T' \cup A$ and $v(T)|_A \ni f_A$. Now $v(T)$ is of size-1, so we can simply write $v(T)|_A = f_A$.

Consider the following two cases:

Case 1. For some $A = \{x_i\}$, Lemma 17 holds for different assignments p, q . In other words, for every $T' \in \binom{X_{b'}}{b'}$, there are $T_1, T_2 \in \binom{X_b}{b}$ satisfying $T_1, T_2 \supseteq T' \cup \{x_i\}$ and $v(T_1)|_{\{x_i\}} = p, v(T_2)|_{\{x_i\}} = q$.

Since for $r \geq 2$ we have $a = r \leq (2(r - 1))^{r-1} = b'$, for each $S \in \binom{X_a}{a}$ containing x_i , we can pick an arbitrary $T' \supseteq S$ and consider the corresponding sets T_1, T_2 above. By the consistency assumption between (S, T_1) and between (S, T_2) in Lemma 18, we can infer $u(S)|_{\{x_i\}} \ni p, q$.

We construct new assignments u', v' for the (a', b') -bipartite direct product 2CSP instance of Π , such that they still meet the consistency requirements in Lemma 18, and the size of u' is at most $r - 1$. For each $S' \in \binom{X}{a'}$, $u'(S')$ will be inherited from $u(S)$ for some $S \in \binom{X}{a}$ satisfying $S \supseteq S' \cup \{x_i\}$. Similarly for each $T' \in \binom{X}{b'}$, $v'(T')$ will be inherited from $v(T)$ for some $T \in \binom{X}{b}$ satisfying $T \supseteq T' \cup \{x_i\}$.

Suppose there is a arbitrary fixed order of all variables in X . We construct S from S' as follows, and output $u'(S') = \{\sigma \in u(S) \mid \sigma(x_i) = p\}_{|S'}$:

- If $x_i \notin S'$, let $S = S' \cup \{x_i\}$.
- If $x_i \in S'$, S is obtained by adding the lexicographical smallest element not in S' to S' . Note that we only keep the assignments in $u(S)$ whose restriction on $\{x_i\}$ is p , and discard those whose restriction on $\{x_i\}$ is q . Thus u' is of size at most $r - 1$ as desired.

For each $T' \in \binom{X}{b'}$, we first construct $T'' \in \binom{X}{b'}$ as follows:

- If $x_i \notin T'$, simply let $T'' = T'$.
- If $x_i \in T'$, T'' is obtained by adding the lexicographical smallest element not in T' to T' , and delete x_i .

After that, we find a $T \in \binom{X}{b}$ satisfying $T \supseteq T'' \cup \{x_i\}$ and $v(T)_{|\{x_i\}} = p$, and output $v'(T') = v(T)_{|T'}$.

By our construction, $S' \subseteq T'$ implies $S \subseteq T$, so the new instance still meets the required consistency requirements, and the induction proceeds.

Case 2. Suppose for A being each $\{x_i\}$, Lemma 17 holds for a unique assignment z_i .

We claim that $(x_i \mapsto z_i)_{x_i \in X}$ is a global satisfying assignment. Indeed, consider an arbitrary constraint ψ between variables $(x_i, x_j) \in X^2$. Applying Lemma 17 to the choice $A = \{x_i, x_j\}$, we know there is an assignment f_A such that for every $T' \in \binom{X}{b'}$, there is some $T \in \binom{X}{b}$ satisfying $T \supseteq T' \cup A$ and $v(T)_{|A} = f_A$. Since $v(T)$ satisfies all constraints within T , this means that f_A must satisfy the constraint ψ . By the uniqueness assumption of this case, we must have $f_A(x_i) = z_i$ and $f_A(x_j) = z_j$, which means that the assignment (z_i, z_j) satisfies the constraint ψ between (x_i, x_j) . ◀

We will now finish the analysis of the bipartite direct product construction.

► **Lemma 19.** *For integers $r, q > 0$, let $a = (2r)^{r+2q}, b = (2r)^{r+2q+2}$. Consider the (a, b) -bipartite direct product 2CSP instance based on $\Pi = (X, \Sigma, \Phi)$. Let u be an r -sized multi-assignment for $\binom{X}{a}$ and v be a q -sized multi-assignment for $\binom{X}{b}$. If u, v list-satisfy all bipartite constraints, then there is a global satisfying assignment σ to the 2CSP instance Π .*

Proof. We apply induction on q , the size of the right multi-assignment. When $q = 1$, we have $a = (2r)^{r+2} \geq r$ and $b = (2r)^{r+4} \geq (2r)^r$. We can extract consistent satisfying assignments for the $(r, (2r)^r)$ -bipartite direct product 2CSP instance, and invoke Lemma 18 to prove there is a global satisfying assignment.

When $q > 1$, we categorize discussions based on whether the left multi-assignment u satisfies a certain property. Either we can still extract consistent assignments for the smaller (a', b') -bipartite direct product 2CSP instance while decreasing q and leaving r unchanged, or we can build multi-assignments that satisfy requirements in Lemma 18, and therefore directly invoke that lemma and stop the induction.

Define $k = (2r)^r$. The parameters a', b' with respect to $q - 1$ would be $a' = (2r)^{r+2q-2}$ and $b' = (2r)^{r+2q}$. In our case-analysis we will use the following inequalities, which we first prove here.

27:10 Baby PIH: Parameterized Inapproximability of Min CSP

■ $b \geq r \cdot b' + a$. Indeed

$$\begin{aligned} r \cdot b' + a &= r \cdot (2r)^{r+2q} + (2r)^{r+2q} \\ &\leq 2 \cdot (2r)^{r+2q+1} \\ &\leq (2r)^{r+2q+2} = b. \end{aligned}$$

■ $a \geq (r+1) \cdot (a' + k)$. Indeed

$$\begin{aligned} (r+1) \cdot (a' + k) &= (r+1) \cdot ((2r)^{r+2q-2} + (2r)^r) \\ &\leq (2r) \cdot (2 \cdot (2r)^{r+2q-2}) \\ &\leq (2r)^{r+2q} = a. \end{aligned}$$

Now we consider the following two cases based on u . The criterion is whether u satisfies a property which is reminiscent of the result of Lemma 17, except in Lemma 17 the property is for v while here we check it for u .

1. Suppose there exists $A \in \binom{X}{k}$ such that for every assignment f_A on A and every $S' \in \binom{X}{a'}$, there exists $S \in \binom{X}{a}$ satisfying $S \supseteq S' \cup A$ and $u(S)|_A \not\equiv f_A$.

Given $a \geq k$ and $b \geq r \cdot b' + a$, we can apply Lemma 17 to know that there is an assignment f_A on A , such that for every $T' \in \binom{X}{b'}$, there is $T \in \binom{X}{b}$ satisfying $T \supseteq T' \cup A$ and $v(T)|_A \equiv f_A$.

We therefore build multi-assignments u', v' for the smaller (a', b') -bipartite 2CSP instance as follows.

For every $S' \in \binom{X}{a'}$, we pick the set S guaranteed by the assumption of this case, and define $u'(S') = u(S)|_{S'}$. Note that u' still has size at most r .

For every $T' \in \binom{X}{b'}$, we pick the set T guaranteed by Lemma 17, and define $v'(T') = \{\sigma \in v(T)|_{\sigma|_A \neq f_A}\}_{T'}$. By the assumption, there exists $S \in \binom{X}{a}$ satisfying $S \supseteq A$ and $u(S)|_A \not\equiv f_A$. Thus to be consistent with $u(S)|_A$, we can conclude for every $T \in \binom{X}{b}$ containing A , $v(T)|_A$ should also contain some value other than f_A . Therefore, our constructed v' is non-empty and has size at most $q-1$.

It's also easy to see u', v' still satisfy list consistency, since in v' we only discard the assignments whose restriction on A equals to f_A , which are not consistent with any u' .

2. Suppose for every $A \in \binom{X}{k}$, there is an assignment f_A for A and a set $S' \in \binom{X}{a'}$, such that for every $S \in \binom{X}{a}$ satisfying $S \supseteq S' \cup A$, we have $u(S)|_A \equiv f_A$.

In this case we can construct r -sized multi-assignment u' and 1-sized assignment v' for left and right part of the $(r, k = (2r)^r)$ -bipartite direct product 2CSP of Π respectively, that meets the requirements of Lemma 18. Furthermore, u', v' are built purely based on u .

For every $B \in \binom{X}{r}$, we define

$$u'(B) = \bigcup_{A \in \binom{X}{k}, A \supseteq B} (f_A)|_B$$

For every $A \in \binom{X}{k}$, we define $v'(A) = f_A$.

We first claim the size of u' is at most r . Suppose it is not, there are $r+1$ sets $A_1, \dots, A_{r+1} \in \binom{X}{k}$ with f_{A_i} all different. Let $S'_1, \dots, S'_{r+1} \in \binom{X}{a'}$ be the corresponding sets guaranteed in the assumption of this case. Consider a $S \in \binom{X}{a}$ which contains $\bigcup_{i=1}^{r+1} (S'_i \cup A_i)$. Such S exists since $a \geq (r+1) \cdot (a' + k)$. Thus, $u(S)|_A$ contains $(r+1)$ different values, contradicting the fact that u is of size r .

It's easy to see u', v' meets the requirement of Lemma 18 by the definition of u' . Thus using Lemma 18, there is a satisfying assignment to the original instance Π . ◀

Our main result, Theorem 14, now follows immediately from Lemma 19.

Proof of Theorem 14. Given an integer r , we pick $t = (2r)^{r+2r+2}$, and prove that if a 2CSP instance $\Pi = (X, \Sigma, \Phi)$ is satisfiable, then so is $\Pi^{\odot t}$; if Π is not satisfiable, then $\Pi^{\odot t}$ is not r -list satisfiable.

The completeness case is easy: let σ be a satisfying assignment for Π , then we can assign each set $S \in \binom{X}{t}$ the function that maps any $x \in S$ to $\sigma(x)$.

For soundness case, suppose $\Pi^{\odot t}$ is r -list satisfiable by an assignment σ , we take $a = (2r)^{r+2q}$, $b = (2r)^{r+2q+2}$ and build multi-assignments u, v for the left and right part of the (a, b) -bipartite direct product 2CSP instance $\Pi^{\odot(a,b)}$. For each set $S' \in \binom{X}{a}$, we pick an arbitrary $S \in \binom{X}{t}$ with $S \supseteq S'$, and define $u(S') = \sigma(S)|_{S'}$. Similarly for each $T' \in \binom{X}{b}$, we pick an arbitrary $T \in \binom{X}{t}$ with $T \supseteq T'$, and define $v(T') = \sigma(T)|_{T'}$. It's easy to see u and v are r -list consistent. Thus by Lemma 19, Π is satisfiable. ◀

4 Average Baby PIH

Let us recall the average Baby PIH conjecture.

► **Hypothesis 20** (Average Baby PIH). *Given a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, we say a multi-assignment σ r -average-list satisfies Π if $\sum_{x \in X} |\sigma(x)| \leq r \cdot |X|$, and for every constraint ϕ_i , there exists $u \in \sigma(w_{i,1})$ and $v \in \sigma(w_{i,2})$, such that $(u, v) \in R_i$.*

Average Baby PIH states that for any constant $r > 1$ and any computable function f , no algorithm can given as input a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, distinguish between the following two cases in $f(|X|) \cdot |\Sigma|^{O(1)}$ time:

- (Completeness) Π is satisfiable.
- (Soundness) Π is not r -average-list satisfiable.

4.1 A Counter Example for Direct Product Construction

We use the following counter example to show that the Direct Product construction does not give us Average Baby PIH. Specifically, for any $t > 0$ and $\varepsilon > 0$, there exists an 2CSP instance which is not satisfiable but its t -wise direct product is $(1 + \varepsilon)$ -average-list satisfiable.

► **Example 21.** The 2CSP instance $\Pi = (X, \Sigma, \Phi)$ is defined as follows.

- $X = \{x_1, \dots, x_n\}$.
- $\Sigma_{x_1} = \{2, \dots, n\}$, and for every $i \in \{2, \dots, n\}$, $\Sigma_{x_i} = \{1\}$.
- For every $i \in \{2, \dots, n\}$, there is a constraint $\phi_i = \langle w_i, R_i \rangle \in \Phi$, where
 - $w_i = (x_1, x_i)$.
 - $R_i = \{(j, 1) | j \neq i\}$.

Π is not satisfiable since no value for x_1 can satisfy all constraints. Specifically, for $i \in \{2, \dots, n\}$, $x_1 = i$ would violate constraint ϕ_i .

However, for every integer $t > 0$, $\Pi^{\odot t}$ can be list satisfied by the following multi-assignment σ . For every $S \in \binom{X}{t}$,

- if $x_1 \notin S$, define $\sigma(S)$ to be the single assignment that maps every $x_i \in S$ to 1;
- if $x_1 \in S$, then for every $2 \leq j \leq 2t$ with $x_j \notin S$, let $\sigma(S)$ contain an assignment that maps x_1 to j , and maps everything else to 1.

It's easy to see σ satisfies all constraints induced by S . It remains to show that σ is consistent on every different $S, T \in \binom{X}{t}$.

27:12 Baby PIH: Parameterized Inapproximability of Min CSP

If $x_1 \notin S$ or $x_1 \notin T$, $\sigma(S)$ and $\sigma(T)$ are trivially consistent since every variable in $S \cap T$ is always assigned 1. Now suppose $x_1 \in S \cap T$, we have $|S \cup T| \leq 2t - 1$. By Pigeonhole Principle, there is a variable x_j with $j \leq 2t$, which is neither in S nor in T . Thus the assignment that maps x_1 to j and everything else to 1 appears in both $\sigma(S)$ and $\sigma(T)$, proving that σ list-satisfies $\Pi^{\circ t}$. The total size of the lists is

$$\begin{aligned} \frac{1}{\binom{|X|}{t}} \sum_{x \in X} |\sigma(x)| &\leq \frac{\binom{|X|-1}{t}}{\binom{|X|}{t}} \cdot 1 + \frac{\binom{|X|-1}{t-1}}{\binom{|X|}{t}} \cdot 2t \\ &= \left(1 - \frac{t}{|X|}\right) \cdot 1 + \frac{t}{|X|} \cdot 2t \\ &\leq 1 + \frac{2t^2}{|X|}, \end{aligned}$$

which is smaller than $1 + \varepsilon$ when $|X|$ goes to infinity.

4.2 Towards the Inapproximability of k -ExactCover

In this subsection, we show that a slightly strengthened version of Average Baby PIH, namely, Average Baby PIH with rectangular relations, implies constant inapproximability of k -EXACTCOVER problem. The latter, to our best knowledge, is currently not known under $W[1] \neq FPT$. Formally, we have the following theorem:

► **Theorem 22.** *Suppose Average Baby PIH holds even when the 2CSP instance has rectangular relations (see Definition 6), then for any constant $r \geq 1$ and any function f , no algorithm can approximate k -EXACTCOVER problem within factor r in $f(k) \cdot n^{O(1)}$ time. Specifically, no algorithm can given a k -SETCOVER instance $\Pi = (\mathcal{S}, U)$ with size n , distinguish between the following two cases in $f(k) \cdot n^{O(1)}$ time:*

- *There exists k sets in \mathcal{S} which form a partition of U .*
- *The union of any $r \cdot k$ sets in \mathcal{S} is not U .*

Proof. We reduce a 2CSP instance $\Pi = (X, \Sigma, \Phi)$ with rectangular relations to a k -SETCOVER instance $\Pi' = (\mathcal{S}, U)$ with $k = |X|$ in the following way. For every $(x, v) \in X \times \Sigma$, we build a set $S_{x,v}$. For each constraint $\phi_i = \langle (w_{i,1}, w_{i,2}), R_i \rangle$ in Φ , let Γ_i be the underlying set in the rectangular relation R_i and let $\pi_i, \sigma_i : \Sigma \rightarrow \Gamma_i$ be the underlying mappings. We build a $(r \cdot k, |\Gamma_i|)$ -set gadget $(\mathcal{M}^{(i)}, C_1^{(i)}, \dots, C_m^{(i)})$. For every $(a, b) \in R_i$, we add the set $C_{\pi_i(a)}^{(i)}$ to $S_{w_{i,1},a}$, and add the set $\overline{C_{\sigma_i(b)}^{(i)}}$ to $S_{w_{i,2},b}$. Let the final universe U be the union of every $\mathcal{M}^{(i)}$.

In the completeness case, let $\sigma : X \rightarrow \Sigma$ be a satisfying assignment of Π . It is easy to see the k sets $\{S_{x,\sigma(x)}\}_{x \in X}$ cover each element of the universe exactly once.

In the soundness case, let $\mathcal{S}' \subseteq \mathcal{S}$ be a collection of sets that covers U . Assuming $|\mathcal{S}'| \leq r \cdot k$, we claim the multi-assignment σ , which maps $x \in X$ to $\{v \in \Sigma \mid S_{x,v} \in \mathcal{S}'\}$, is a list satisfying assignment of Π . For every constraint $\phi_i = \langle (w_{i,1}, w_{i,2}), R_i \rangle$ with Γ_i being the image of the rectangular mapping, by the property of $(r \cdot k, |\Gamma_i|)$ -set gadget and the fact that $|\mathcal{S}'| \leq r \cdot k$, \mathcal{S}' must include $C_j^{(i)}$ and $\overline{C_j^{(i)}}$ for some $1 \leq j \leq |\Gamma_i|$, which implies that \mathcal{S}' includes $S_{w_{i,1},a}$ and $S_{w_{i,2},b}$ for some $(a, b) \in R_i$, and thus ϕ_i is list satisfied.

From Lemma 13, a $(r \cdot k, |\Gamma_i|)$ -set gadget can be constructed in time $\text{poly}(|\Gamma_i|, 2^{r \cdot k})$. The whole reduction runs in FPT time while preserving $k = |X|$. Thus, an $f(k) \cdot n^{O(1)}$ time algorithm for r -approximating k -EXACTCOVER would give an $f(k) \cdot |\Sigma|^{O(1)}$ algorithm for the r -average-list satisfiability of 2CSP with rectangular relations, contradicting the strengthened version of Average Baby PIH. ◀

We remark that the direct product construction does have rectangular relations, although it does not directly give us Average Baby PIH, in view of Example 21.

5 Discussion and Open Problems

In this concluding section, we speculate on some possible avenues to attack Average Baby PIH or even PIH itself.

5.1 Average Baby PIH from Clique Hardness?

In [31], the author proved that even constant approximating k -CLIQUE is $W[1]$ -hard. In [24, 8], the inapproximability ratio was improved to $k^{o(1)}$. Using CSP words, their results can be described as the following theorem:

► **Theorem 23** ([24, 8]). *Assuming $W[1] \neq \text{FPT}$, no algorithm can, given a 2CSP instance $\Pi = (X, \Sigma, \Phi)$, distinguish between the following two cases in $f(|X|) \cdot |\Sigma|^{O(1)}$ time, for any computable function f :*

- Π is satisfiable.
- The constraints induced by any $|X|^{1-o(1)}$ variables are not simultaneously satisfiable.

Thus, it is natural to ask whether we can get Average Baby PIH by applying direct product construction to a 2CSP instance with the above “clique-like” soundness? More formally, we have the following open question:

► **Open Question 1.** *Is it true that for any integer $r > 1$, there is an integer $t > 0$, such that for any 2CSP instance $\Pi = (X, \Sigma, \Phi)$, if $\Pi^{\otimes t}$ is r -average-list satisfiable, then there are $|X|^{1-o(1)}$ variables in Π such all constraints amongst them are simultaneously satisfiable?*

Note that the counterexample for proving average baby PIH using the direct product construction (Example 21) does not apply here, since there are $|X| - 1$ variables in Π that are simultaneously satisfiable.

5.2 PIH via Direct Product Testing Theorems?

Our constructed instance in proving Baby PIH is reminiscent of the direct product testing, which has been studied in a recent line of work [18, 15, 14]. These results have shown that if the t -sized subsets of variables have good local consistency, then there is a global function which agrees with most of the subsets. Formally, we have the following theorem from [14].

► **Theorem 24.** *There is an absolute constant $C > 1$, such that for any $\alpha, \beta \in (0, 1)$ with $\alpha + \beta \leq 1$, there exists a constant $Q(\alpha, \beta)$, such that given any k, t, m with $k \geq C \cdot t$ and $\alpha t \leq m \leq (1 - \beta)t$ and any finite alphabet Σ , we have the following.*

Let $\mathcal{F} = \{f_S : S \rightarrow \Sigma \mid S \in \binom{[k]}{t}\}$ be an ensemble of functions, one for every size- t subset of $[k]$. Let $\mathcal{D}(m)$ be the distribution as follows:

- Choose $I \in \binom{[k]}{m}$ uniformly at random.
- Choose A, B from the set $\{X \mid X \in \binom{[k]}{t}, X \supseteq I\}$ uniformly at random.

Suppose the following holds:

$$\Pr_{A, B \sim \mathcal{D}(m)} [f_A|_{A \cap B} = f_B|_{A \cap B}] \geq 1 - \varepsilon,$$

then there exists a global function $g : [k] \rightarrow \Sigma$ such that

$$\Pr_{A \sim \binom{[k]}{t}} [f_A = g|_A] \geq 1 - Q(\alpha, \beta) \cdot \varepsilon.$$

By setting the alphabet of each f_S to be the set of all partial satisfying assignments for S and adding the consistency checks according to the above theorem, one may wonder whether we can “extract” a large clique from the $1 - Q(\alpha, \beta) \cdot \varepsilon$ fraction of subsets that are globally consistent. Formally, let $\mathcal{S} = \{A \mid A \in \binom{[k]}{t}, f_A = g_{|A}\}$ as in Theorem 24, can we prove there exists a subset $T \subseteq [k]$ of size $\geq k^{1-o(1)}$, such that the following holds?

- For every $(i, j) \in \binom{[T]}{2}$, there exists $A \in \mathcal{S}$ with $(i, j) \subseteq A$.

However, this might not be true if we only use the size bound on \mathcal{S} and not the additional structures. Consider the following counter-example:

► **Example 25.** Mark each pair $(i, j) \in \binom{[k]}{2}$ as 1 independently with probability $1 - \gamma$ with γ to be determined. Take \mathcal{S}' to be the collection of t -sized subsets of $[k]$, with all pairs in it marked as 1: $\mathcal{S}' := \{A \mid A \in \binom{[k]}{t}, \forall (i, j) \subseteq A, (i, j) \text{ is marked as } 1\}$.

The probability that a t -sized subset belongs to \mathcal{S}' is $(1 - \gamma)^{\binom{t}{2}}$, which can be made arbitrarily close to 1 when we set $\gamma = \gamma(t)$ to be some small enough constant depending on the constant t .

However, it was known that (see e.g. [21]) the maximum clique size in this Erdős-Rényi graph is only $2 \log k / \log(1/(1 - \gamma))$, far smaller than $k^{1-o(1)}$.

This example suggests that, in order to potentially prove PIH from direct product testing theorems, one may need to analyze the structure of the collection \mathcal{S} . We leave this as an interesting future direction:

► **Open Question 2.** *Can we prove PIH under $W[1] \neq \text{FPT}$ using some appropriate form of direct product testing theorems?*

References


- 1 Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM J. Comput.*, 49(4), 2020. doi:10.1137/18M1171321.
- 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. doi:10.1006/jcss.1997.1472.
- 3 Sanjeev Arora and Carsten Lund. *Hardness of Approximations*, pages 399–446. In Dorit S. Hochbaum, editor, *Approximation algorithms for NP-hard problems*. PWS Publishing, 1996.
- 4 Libor Barto and Marcin Kozik. Combinatorial gap theorem and reductions between promise csp. 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 1204–1220. SIAM, 2022. doi:10.1137/1.9781611977073.50.
- 5 Jaroslaw Byrka, Thomas W. Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k -median and positive correlation in budgeted optimization. *ACM Trans. Algorithms*, 13(2):23:1–23:31, 2017. doi:10.1145/2981561.
- 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 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.
- 7 Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *J. Comput. Syst. Sci.*, 65(1):129–149, 2002. doi:10.1006/jcss.2002.1882.

- 8 Yijia Chen, Yi Feng, Bundit Laekhanukit, and Yanlin Liu. Simple combinatorial construction of the $k^{O(1)}$ -lower bound for approximating the parameterized k-clique. *CoRR*, abs/2304.07516, 2023. doi:10.48550/arXiv.2304.07516.
- 9 Yijia Chen and Martin Grohe. An isomorphism between subexponential and parameterized complexity theory. *SIAM Journal on Computing*, 37(4):1228–1258, 2007.
- 10 Yijia Chen and Bingkai Lin. The constant inapproximability of the parameterized dominating set problem. *SIAM J. Comput.*, 48(2):513–533, 2019. doi:10.1137/17M1127211.
- 11 Rajesh Chitnis, Andreas Emil Feldmann, and Pasin Manurangsi. Parameterized approximation algorithms for directed steiner network problems. *CoRR*, abs/1707.06499, 2017. arXiv:1707.06499.
- 12 Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight FPT approximations for k-median and k-means. 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 42:1–42:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICs.ICALP.2019.42.
- 13 Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken-ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 637–646. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.14.
- 14 Irit Dinur, Yuval Filmus, and Prahladh Harsha. Analyzing boolean functions on the biased hypercube via higher-dimensional agreement tests: [extended abstract]. 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 2124–2133. SIAM, 2019. doi:10.1137/1.9781611975482.128.
- 15 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 974–985. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.94.
- 16 Irit Dinur and Pasin Manurangsi. ETH-hardness of approximating 2-csp and directed steiner network. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPICs*, pages 36:1–36:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.ITCS.2018.36.
- 17 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.
- 18 Irit Dinur and David Steurer. Direct product testing. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 188–196. IEEE Computer Society, 2014. doi:10.1109/CCC.2014.27.
- 19 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.
- 20 Michael R Fellows. Blow-ups, win/win’s, and crown rules: Some new directions in fpt. In *Graph-Theoretic Concepts in Computer Science: 29th International Workshop, WG 2003, Elspeet, The Netherlands, June 19-21, 2003. Revised Papers 29*, pages 1–12. Springer, 2003.
- 21 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.
- 22 Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, Yican Sun, and Kewen Wu. Parameterized inapproximability hypothesis under eth, 2023. arXiv:2311.16587.

- 23 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Comput. Geom.*, 28(2-3):89–112, 2004. doi:10.1016/j.comgeo.2004.03.003.
- 24 Karthik C. S. and Subhash Khot. Almost polynomial factor inapproximability for parameterized k-clique. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPICs*, pages 6:1–6:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.CCC.2022.6.
- 25 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.
- 26 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.
- 27 Euiwoong Lee. Partitioning a graph into small pieces with applications to path transversal. *Math. Program.*, 177(1-2):1–19, 2019. doi:10.1007/s10107-018-1255-7.
- 28 Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. *SIAM J. Comput.*, 45(2):530–547, 2016. doi:10.1137/130938645.
- 29 Bingkai Lin. The parameterized complexity of the k-biclique problem. *J. ACM*, 65(5):34:1–34:23, 2018. doi:10.1145/3212622.
- 30 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.
- 31 Bingkai Lin. Constant approximating k-clique is w[1]-hard. 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 1749–1756. ACM, 2021. doi:10.1145/3406325.3451016.
- 32 Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. On lower bounds of approximating parameterized k-clique. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPICs*, pages 90:1–90:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.ICALP.2022.90.
- 33 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.
- 34 Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. Improved hardness of approximating k-clique under ETH. *CoRR*, abs/2304.02943, 2023. doi:10.48550/arXiv.2304.02943.
- 35 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 224–237. ACM, 2017. doi:10.1145/3055399.3055456.
- 36 Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Parameterized complexity and approximability of directed odd cycle transversal. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2181–2200. SIAM, 2020. doi:10.1137/1.9781611975994.134.
- 37 Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, 1994. doi:10.1145/185675.306789.

- 38 Pasin Manurangsi. A note on max k -vertex cover: Faster fpt-as, smaller approximate kernel and improved approximation. 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 15:1–15:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/OASIcs.SOSA.2019.15.
- 39 Pasin Manurangsi. Tight running time lower bounds for strong inapproximability of maximum k -coverage, unique set cover and related problems (via t -wise agreement testing theorem). In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 62–81. SIAM, 2020. doi:10.1137/1.9781611975994.5.
- 40 Dániel Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008. doi:10.1093/comjnl/bxm048.
- 41 Anup Rao. Parallel repetition in projection games and a concentration bound. *SIAM J. Comput.*, 40(6):1871–1891, 2011. doi:10.1137/080734042.
- 42 Piotr Skowron and Piotr Faliszewski. Chamberlin-courant rule with approval ballots: Approximating the maxcover problem with bounded frequencies in FPT time. *J. Artif. Intell. Res.*, 60:687–716, 2017. doi:10.1613/jair.5628.
- 43 Andreas Wiese. Fixed-parameter approximation schemes for weighted flowtime. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA*, volume 116 of *LIPIcs*, pages 28:1–28:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.28.

Finding Missing Items Requires Strong Forms of Randomness

Amit Chakrabarti 

Department of Computer Science, Dartmouth College, Hanover, NH, USA

Manuel Stoeckl 

Department of Computer Science, Dartmouth College, Hanover, NH, USA

Abstract

Adversarially robust streaming algorithms are required to process a stream of elements and produce correct outputs, even when each stream element can be chosen as a function of earlier algorithm outputs. As with classic streaming algorithms, which must only be correct for the worst-case fixed stream, adversarially robust algorithms with access to randomness can use significantly less space than deterministic algorithms. We prove that for the Missing Item Finding problem in streaming, the space complexity also significantly depends on how adversarially robust algorithms are permitted to use randomness. (In contrast, the space complexity of classic streaming algorithms does not depend as strongly on the way randomness is used.)

For Missing Item Finding on streams of length ℓ with elements in $\{1, \dots, n\}$, and $\leq 1/\text{poly}(\ell)$ error, we show that when $\ell = O(2^{\sqrt{\log n}})$, “random seed” adversarially robust algorithms, which only use randomness at initialization, require $\ell^{\Omega(1)}$ bits of space, while “random tape” adversarially robust algorithms, which may make random decisions at any time, may use $O(\text{polylog}(\ell))$ random bits. When ℓ is between $n^{\Omega(1)}$ and $O(\sqrt{n})$, “random tape” adversarially robust algorithms need $\ell^{\Omega(1)}$ space, while “random oracle” adversarially robust algorithms, which can read from a long random string for free, may use $O(\text{polylog}(\ell))$ space. The space lower bound for the “random seed” case follows, by a reduction given in prior work, from a lower bound for pseudo-deterministic streaming algorithms given in this paper.

2012 ACM Subject Classification Theory of computation \rightarrow Sketching and sampling; Theory of computation \rightarrow Lower bounds and information complexity; Theory of computation \rightarrow Pseudorandomness and derandomization

Keywords and phrases Data streaming, lower bounds, space complexity, adversarial robustness, derandomization, sketching, sampling

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.28

Related Version *Full Version:* <https://arxiv.org/abs/2310.03634> [9]

Extended in Chapter 3 of: <https://digitalcommons.dartmouth.edu/dissertations/229/> [25]

Funding Supported in part by the National Science Foundation under Award 2006589.

1 Introduction

Randomized streaming algorithms can achieve exponentially better space bounds than corresponding deterministic ones: this is a basic, well-known, easily proved fact that applies to a host of problems of practical interest. A prominent class of randomized streaming algorithms uses randomness in a very specific way, namely to sketch the input stream by applying a random linear transformation – given by a sketch matrix S – to the input frequency vector. The primary goal of a streaming algorithm is to achieve sublinear space, so it is infeasible to store S explicitly. In some well-known cases, the most natural presentation of the algorithm is to explicitly describe the distribution of S , a classic case in point being frequency moment estimation [16]. This leads to an algorithm that is very space-efficient *provided one doesn't charge the algorithm any space cost for storing S* . Algorithms that



© Amit Chakrabarti and Manuel Stoeckl;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 28; pp. 28:1–28:20
Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



work this way can be thought of as accessing a “random oracle”: despite their impracticality, they have theoretical value, because the standard ways of proving space *lower* bounds for randomized streaming algorithms in fact work in this model. For the specific frequency-moment algorithms mentioned earlier, [16] goes on to design variants of his algorithms that use only a small (sublinear) number of random bits and apply a pseudorandom generator to suitably mimic the behavior of his random-oracle algorithms. Thus, at least in this case, a random *oracle* isn’t necessary to achieve sublinear complexity. This raises a natural question: from a space complexity viewpoint, does it ever help to use a random oracle, as opposed to “ordinary” random bits that must be stored (and thus paid for) if they are to be reused?

For most classic streaming problems, the answer is “No,” but for unsatisfactory reasons: Newman’s Theorem [21] allows one to replace a long oracle-provided random string by a much shorter one (that is cheap to store), though the resulting algorithm is non-constructive. This brings us to the recent and ongoing line of work on *adversarially robust* streaming algorithms where we shall find that the answer to our question is a very interesting “Yes.” For the basic and natural MISSINGITEMFINDING problem, defined below, we shall show that three different approaches to randomization result in distinct space-complexity behaviors. To explain this better, let us review adversarial robustness briefly.

Some recent works have studied streaming algorithms in a setting where the input to the algorithm can be adaptively (and adversarially) chosen based on its past outputs. Existing (“classic”) randomized streaming algorithms may fail in this *adversarial setting* when the input-generating adversary learns enough about the past random choices of the algorithm to identify future inputs on which the algorithm will likely fail. There are, heuristically, two ways for algorithm designers to protect against this: (a) prevent the adversary from learning the past random choices of the algorithm (in the extreme, by making a pseudo-deterministic algorithm), or (b) prevent the adversary from exploiting knowledge of past random decisions, by having the algorithm’s future behavior depend on randomness that it has not yet revealed. Concretely, algorithms in this setting use techniques such as independent re-sampling [6], sketch switching using independent sub-instances of an underlying classic algorithm [5], rounding outputs to limit the number of computation paths [5], and differential privacy to safely aggregate classic algorithm sub-instances [15]. Mostly, these algorithms use at most as many random bits as their space bounds allow. However, some recently published adversarially robust streaming algorithms for vertex-coloring a graph (given by an edge stream) [8, 2], and one for the MISSINGITEMFINDING problem [24], assume access to a large amount of oracle randomness: they prevent the adversary from exploiting the random bits it learns by making each output depend on an unrevealed part of the oracle random string. It is still open whether these last two problems have efficient solutions that do not use this oracle randomness hammer. This suggests the following question:

Are there problems for which space-efficient adversarially robust streaming algorithms provably require access to oracle randomness?

In this paper, we prove that for certain parameter regimes, MISSINGITEMFINDING (henceforth, MIF) is such a problem. In the problem $\text{MIF}(n, \ell)$, the input is a stream $\langle e_1, \dots, e_\ell \rangle$ of ℓ integers, not necessarily distinct, with each $e_i \in \{1, \dots, n\}$, where $1 \leq \ell \leq n$. The goal is as follows: having received the i th integer, output a number v in $\{1, \dots, n\} \setminus \{e_1, \dots, e_i\}$. We will be mostly interested in the setting $\ell = o(n)$, so the “trivial” upper bound on the space complexity of $\text{MIF}(n, \ell)$ is $O(\ell \log n)$, achieved by the deterministic algorithm that simply stores the input stream as is.

1.1 Groundwork for Our Results

To state our results about MIF, we need to introduce some key terminology. Notice that MIF is a *tracking problem*: an output is required after reading each input. Thus, we view streaming algorithms as generalizations of finite state (Moore-type) machines. An algorithm \mathcal{A} has a finite set of states Σ (leading to a space cost of $\log_2 |\Sigma|$), a finite input set \mathcal{I} , and a finite output set \mathcal{O} . It has a transition function $T: \Sigma \times \mathcal{I} \times \mathcal{R} \rightarrow \Sigma$ indicating the state to switch to after receiving an input, plus an output function $\gamma: \Sigma \times \mathcal{R} \rightarrow \mathcal{O}$ indicating the output produced upon reaching a state. How the final parameter (in \mathcal{R}) of T and γ is used depends on the type of randomness. We consider four cases, leading to four different models of streaming computation.

- *Deterministic.* The initial state of the algorithm is a fixed element of Σ , and T and γ are deterministic (they do not depend on the parameter in \mathcal{R}).
- *Random seed.* The initial state is drawn from a distribution \mathcal{D} over Σ , and T and γ are deterministic. This models the situation that all random bits used count towards the algorithm's space cost.
- *Random tape.* The initial state is drawn from a distribution \mathcal{D} over Σ . The space \mathcal{R} is a sample space; when the algorithm receives an input $e \in \mathcal{I}$ and is at state $\sigma \in \Sigma$, it chooses a random $\rho \in \mathcal{R}$ independent of all previous choices and moves to state $T(e, \sigma, \rho)$. However, γ is deterministic. This models the situation that the algorithm can make random decisions at any time, but it cannot remember past random decisions without recording them (which would add to its space cost).
- *Random oracle.* The initial state is fixed; \mathcal{R} is a sample space. A specific $R \in \mathcal{R}$ is drawn at the start of the algorithm and stays the same over its lifetime. When the algorithm is at state σ and receives input e , its next state is $T(e, \sigma, R)$. The output given at state σ is $\gamma(\sigma, R)$. This models the situation that random bits are essentially “free” to the algorithm; it can read from a long random string which doesn't count toward its space cost and which remains consistent over its lifetime. A random oracle algorithm can be interpreted as choosing a random deterministic algorithm, indexed by R , from some family.

These models form a rough hierarchy; they have been presented in (almost) increasing order of power. Every z -bit (2^z -state) deterministic algorithm can be implemented in any of the random models using z bits of space; the same holds for any z -bit random seed algorithm. Every z -bit random tape algorithm has a corresponding $(z + \log \ell)$ -bit random oracle algorithm – the added space cost is because for a random oracle algorithm to emulate a random tape algorithm, it must have a way to get “fresh” randomness on each turn.¹

Streaming algorithms are also classified by the kind of correctness guarantee they provide. Recall that we focus on “tracking” algorithms [5]; they present an output after reading each input item and this *entire sequence* of outputs must be correct. Here are three possible meanings of the statement “algorithm \mathcal{A} is δ -error” (we assume that \mathcal{A} handles streams of length ℓ with elements in \mathcal{I} and has outputs in \mathcal{O}):

- *Static setting.* For all inputs $\tau \in \mathcal{I}^\ell$, running \mathcal{A} on τ produces incorrect output with probability $\leq \delta$.

¹ An alternative, which lets one express z -bit random tape algorithms using a z -bit random oracle variant, is to assume the random oracle algorithm has access to a clock or knows the position in the stream for free; both are reasonable assumptions in practice.

- *Adversarial setting.* For all (computationally unbounded) adaptive adversaries α (i.e., for all functions $\alpha: \mathcal{O}^* \rightarrow \mathcal{I}$),² running \mathcal{A} against α will produce incorrect output with probability $\leq \delta$.
- *Pseudo-deterministic setting.* There exists a canonical output function $f: \mathcal{I}^* \rightarrow \mathcal{O}$ producing all correct outputs so that, for each $\tau \in \mathcal{I}^\ell$, $\mathcal{A}(\tau)$ fails to output $f(\tau)$ with probability $\leq \delta$.

Algorithms for the static setting are called “classic” streaming algorithms; ones for the adversarial setting are called “adversarially robust” streaming algorithms. All pseudo-deterministic algorithms are adversarially robust, and all adversarially robust algorithms are also classic.

As a consequence of Newman’s theorem [21], any random oracle or random tape algorithm in the static setting with error δ can be emulated using a random seed algorithm with only ε increase in error and an additional $O(\log \ell + \log \log |\mathcal{I}| + \log \frac{1}{\varepsilon \delta})$ bits of space. However, the resulting algorithm is non-constructive.

1.2 Our Results

As context for our results, we remind the reader that it’s trivial to solve $\text{MIF}(n, \ell)$ in $O(\ell \log n)$ space deterministically (somewhat better deterministic bounds were obtained in [24]). Moving to randomized algorithms, [24] gave a space bound of $O(\log^2 n)$ for $\ell \leq n/2$ in the static setting, and a bound of $\tilde{O}(\ell^2/n + 1)$ ³ in the adversarial setting, using a random *oracle*. The immediate takeaway is that, given access to a deep pool of randomness (i.e., an oracle), MIF becomes easy in the static setting for essentially the full range of stream lengths ℓ and remains easy even against an adversary for lengths $\ell \leq \sqrt{n}$.

The main results of this paper consist of three new lower bounds and one new upper bound on the space complexity of $\text{MIF}(n, \ell)$. Stating the bounds in their strongest forms leads to complicated expressions; therefore, we first present some easier-to-read takeaways from these bounds that carry important conceptual messages. In the lower bounds below, the error level should be thought of as $\delta = 1/n^2$.

► **Result 1.** *At $\ell = \sqrt{n}$, adversarially robust random tape algorithms for $\text{MIF}(n, \ell)$ require $\Omega(\ell^{1/4})$ bits of space. More generally, for every constant $\alpha \in (0, 1)$, there is a constant $\beta \in (0, 1)$ such that at $\ell = \Omega(n^\alpha)$, the space requirement is $\Omega(\ell^\beta)$, in the adversarially robust random tape setting.*

This shows that MIF remains hard, even for modest values of ℓ , if we must be robust while using only a random *tape*, i.e., if there is a cost to storing random bits we want to reuse – a very reasonable requirement for a practical algorithm. The above result is an exponential separation between the random tape and random oracle models.

The random *seed* model places an even greater restriction on an algorithm: besides counting towards storage cost, random bits are available only at initialization and not on the fly. Many actual randomized algorithms, including streaming ones, are structured this way, making it a natural model to study. We obtain the following result.

► **Result 2.** *Adversarially robust random seed algorithms for $\text{MIF}(n, \ell)$ require $\tilde{\Omega}(\sqrt{\ell})$ bits of space.*

² By the minimax theorem, it suffices to consider deterministic adversaries.

³ The notations $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ hide factors polylogarithmic in n and ℓ .

■ **Table 1** Bounds for the space complexity of $\text{MIF}(n, \ell)$, from this and prior work. To keep expressions simple, these bounds are evaluated at error level $\delta = 1/n^2$, when applicable. † indicates that the precise results are stronger.

Setting	Type	Bound	Reference
Static	Random seed	$O((\log n)^2)$ if $\ell \leq n/2$	[24]
Adversarial	Random oracle	$O((\frac{\ell^2}{n} + \log n) \log n)$	[24]
		$\Omega(\frac{\ell^2}{n})$	[24]
Adversarial	Random tape	$O(\ell^{\log_n \ell} (\log \ell)^2 + \log \ell \cdot \log n)$ †	Theorem 1
		$\Omega(\frac{\log \ell}{\log n} \ell^{\frac{15}{32} \log_n \ell})$ †	Theorem 8
Adversarial	Random seed	$O((\frac{\ell^2}{n} + \sqrt{\ell} + \log n) \log n)$	[24] ^{a)}
		$\Omega(\frac{\ell^2}{n} + \sqrt{\frac{\ell}{(\log n)^3}} + \ell^{1/5})$	Theorem 10
Pseudo-deterministic	Random oracle	$\Omega(\frac{\ell}{(\log(2n/\ell))^2} + (\ell \log n)^{1/4})$	Theorem 16
Static	Deterministic	$\Omega(\frac{\ell}{\log(2n/\ell)} + \sqrt{\ell})$	[24]
		$O(\frac{\ell \log \ell}{\log n} + \sqrt{\ell \log \ell})$	[24]

a) The random seed algorithm for the adversarial setting is given in the arXiv version of [24].

Consider the two results above as ℓ decreases from \sqrt{n} to $\Theta(1)$. The bound in Result 2 stays interesting even when $\ell = n^{o(1)}$, so long as $\ell \geq (\log n)^C$ for a suitable constant C (in fact, the full version of the result is good for even smaller ℓ). In contrast, the bound in Result 1 peters out at much larger values of ℓ . There is a very good reason: MIF starts to become “easy,” even under a random-tape restriction, once ℓ decreases to sub-polynomial in n . Specifically, we obtain the following *upper* bound.

► **Result 3.** *There is an adversarially robust random tape algorithm for $\text{MIF}(n, \ell)$ that, in the regime $\ell = O(2^{\sqrt{\log n}})$, uses $O(\log \ell \cdot \log n)$ bits of space.*

Notice that at $\ell = \Theta(2^{(\log n)^{1/C}})$, where $C \geq 2$ is a constant, the bound in Result 3 is polylogarithmic in ℓ . Combined with the lower bound in Result 2, we have another exponential separation, between the random seed and random tape models.

The proof of Result 2 uses a reduction, given in prior work [24], that converts a space lower bound in the pseudo-deterministic setting to a related bound in the random-seed setting. A pseudo-deterministic algorithm is allowed to use randomness (which, due to Newman’s theorem, might as well be of the oracle kind) but must, with high probability, map each input to a *fixed* output, just as a deterministic algorithm would. This strong property makes the algorithm adversarially robust, because the adversary has nothing to learn from observing its outputs. Thanks to the [24] reduction, the main action in the proof of Result 2 is the following new lower bound we give.

► **Result 4.** *Pseudo-deterministic random oracle algorithms for $\text{MIF}(n, \ell)$ require $\tilde{\Omega}(\ell)$ bits of space.*

These separations rule out the possibility of a way to convert an adversarially robust random oracle algorithm to use only a random *seed* or even a random *tape*, with only minor (e.g., a $\text{polylog}(\ell, n)$ factor) overhead. In contrast, as we noted earlier, such a conversion is routine in the static setting, due to Newman’s theorem [21]. The separation between random

oracle and random tape settings shows that MISSINGITEMFINDING is a problem for which much lower space usage is possible if one’s adversaries are computationally bounded (in which case a pseudo-random generator can emulate a random oracle.)

Table 1 shows more detailed versions of the above results as well as salient results from earlier work, summarizing the state of the art for the space complexity of MIF(n, ℓ). The fully detailed versions of our results, showing the dependence of the bounds on the error probability, appear in later sections of the paper, as indicated in the table.

1.3 Related Work

We briefly survey related work. An influential early work [14] considered adaptive adversaries for *linear* sketches. The adversarial setting was formally introduced by [5], who provided general methods (like sketch-switching) for designing adversarially robust algorithms given classic streaming algorithms, especially in cases where the problem is to approximate a real-valued quantity. For some tasks, like F_0 -estimation, they obtained slightly better upper bounds by using a random oracle, although later work [26] removed this need. [6] observed that in sampling-based streaming algorithms, increasing the sample size is often all that is needed to make an algorithm adversarially robust. [15] described how to use differential privacy techniques as a more efficient alternative to sketch-switching, and [4] used this as part of a more efficient adversarially robust algorithm for turnstile F_2 -estimation.

Most of these papers focus on providing algorithms and general techniques, but there has been some work on proving adversarially robust lower bounds. [18] described a problem (of approximating a certain real-valued function) that requires exponentially more space in the adversarial setting than in the static setting. [8], in a brief comment, observed a similar separation for a simple problem along the lines of MIF. They also proved lower bounds for adversarially robust coloring algorithms for graph edge-insertion streams. [24] considered the MIF problem as defined here and, among upper and lower bounds in a number of models, described an adversarially robust algorithm for MIF that requires a random oracle; they asked whether a random oracle is *necessary* for space-efficient algorithms.

The *white-box* adversarial setting [1] is similar to the adversarial setting we study, with the adversary having the additional power of seeing the internal state of the algorithm, including (if used) the random oracle. [24] proved an $\Omega(\ell/\text{polylog}(n))$ lower bound for MIF(n, ℓ) for random tape algorithms in this setting, suggesting that any more efficient algorithm for MIF must conceal some part of its internal state. Pseudo-deterministic streaming algorithms were introduced by [12], who gave lower bounds for a few problems. [7, 13] gave lower bounds for pseudo-deterministic algorithms that approximately count the number of stream elements. The latter shows they require $\Omega(\log m)$ space, where m is the stream length; in contrast, in the static setting, Morris’s counter algorithm⁴ uses only $O(\log \log m)$ space.

While it is not posed as a streaming task, the *mirror game* introduced by [11] is another problem with conjectured separation between the space needed for different types of randomness. In the mirror game, two players (Alice and Bob) alternately state numbers in the set $\{1, \dots, n\}$, where n is even, without repeating any number, until one player mistakenly states a number said before (loss) or the set is completed (tie). [11] showed that if Alice has $o(n)$ bits of memory and plays a deterministic strategy, Bob can always win. Later, [10, 20] showed that if Alice has access to a random oracle, she can tie-or-win w.h.p. using only

⁴ Morris’s is a “random tape” algorithm; “random seed” algorithms for counting aren’t better than deterministic ones.

$O(\text{polylog}(n))$ space. A major open question here is how much space Alice needs when she does not have a random oracle. [19] did not resolve this, but showed that if Alice is “open-book” (equivalently, that Bob is a white-box adversary and can see her state), then Alice needs $\Omega(n)$ bits of state to tie-or-win.

Assuming access to a random oracle is a reasonable temporary measure when designing streaming algorithms in the static setting. As noted at the beginning of Section 1, [16] designed L_p -estimation algorithms using random linear sketch matrices, without regard to the amount of randomness used, and then described a way to apply Nisan’s PRG [22] to partially derandomize these algorithms and obtain efficient (random seed) streaming algorithms. In general, the use of PRGs for linear sketches has some space overhead, which later work (see [17] as a recent example) has been working to eliminate.

It is important to distinguish the “random oracle” type of streaming algorithm from the “random oracle model” in cryptography [3], in which one assumes that *all* agents have access to the random oracle. [1], when defining white-box adversaries, also assumed that they can see the same random oracle as the algorithm; and, for one task, obtained a more efficient algorithm against a computationally bounded white-box adversary, when both have access to a random oracle, than when neither do. Tight lower bounds are known in neither case.

The power of different types of access to randomness has been studied in computational complexity. [23] showed that logspace Turing machines with a multiple-access random tape can (with zero error) decide languages that logspace Turing machines with a read-once random tape decide only with bounded two-sided error. This type of separation does not hold for *time* complexity classes.

For a more detailed history and survey of problems related to MISSINGITEMFINDING, we direct the reader to [24].

2 Organization of This Extended Abstract

What follows is an extended abstract of our paper, which omits formal proofs of our results. Instead, we give a technical overview of each result, followed by selected details of its proof. The full paper contains all remaining details and formal proofs.

2.1 Notation

Throughout this paper, $\log x = \log_2 x$, while $\ln x = \log_e x$. The set \mathbb{N} consists of all positive integers; $[k] := \{1, 2, \dots, k\}$; and $[a, b)$ is a half open interval of real numbers. For a condition or event E , the symbol $\mathbb{1}_E$ takes the value 1 if E occurs and 0 otherwise. The sequence (stream) obtained by concatenating sequences a and b , in that order, is denoted $a \circ b$. For a set S of elements in a totally ordered universe, $\text{SORT}(S)$ denotes the sequence of elements of S in increasing order; $\binom{S}{k}$ is the set of k -element subsets of S ; and $\text{SEQS}(S, k) = \{\text{SORT}(Y) : Y \in \binom{S}{k}\}$. We sometimes extend set-theoretic notation to vectors and sequences; e.g., for $y \in [n]^t$, write $y \subseteq S$ to mean that $\forall i \in [t] : y_i \in S$. For a set X , $\Delta[X]$ denotes the set of probability distributions over X , while $A \sim X$ indicates that A is chosen uniformly at random from X .

2.2 Preliminary Remarks

The proofs of Results 1, 3, and 4 are all significant generalizations of existing proofs from [24] which handled different (and more tractable) models. The proof of Result 2 consists of applying a reduction from [24] to the lower bound given by Result 4. As we explain our techniques, we will summarize the relevant “basic” proofs from [24], which will clarify the enhancements needed to obtain our results.

Space complexity lower bounds in streaming models are often proved via communication complexity. This meta-technique is unavailable to us, because the setup of communication complexity blurs the distinctions between random seed, random tape, and random oracle models and our results are all about these distinctions. Instead, to prove Result 1, we design a suitable strategy for the stream-generating adversary that exploits the algorithm’s random-tape limitation by learning enough about its internal state. Our adversary uses a nontrivially recursive construction. To properly appreciate it, it is important to understand what streaming-algorithmic techniques the adversary must contend with. Therefore, we shall discuss our *upper* bound result first.

3 Random Tape Upper Bound (Result 3)

The adversarially robust random tape algorithm for $\text{MIF}(n, \ell)$ can be seen as a generalization of the random oracle and random seed algorithms.

The random oracle algorithm and its adversaries. The random oracle algorithm for $\text{MIF}(n, \ell)$ from [24] has the following structure. It interprets its oracle random string as a uniformly random sequence L containing $\ell + 1$ distinct elements in $[n]$. As it reads its input, it keeps track of which elements in L were in the input stream so far (were “covered”). It reports as its output the first uncovered element of L . Because L comes from the oracle, the space cost of the algorithm is just the cost of keeping track of the set J of covered positions in L . We will explain why that can be done using only $O((\ell^2/n + 1) \log \ell)$ space, in expectation.

An adversary for the algorithm only has two reasonable strategies for choosing the next input. It can “echo” back the current algorithm output to be the next input to the algorithm. It can also choose the next input to be a value from the set U of values that are neither an earlier input nor the current output – but because L is chosen uniformly at random, one can show that the adversary can do no better than picking the next input uniformly at random from U . (The third strategy, of choosing an old input, has no effect on the algorithm.) When the algorithm is run against an adversary that chooses inputs using a mixture of the echo and random strategies, the set J will be structured as the union of a contiguous interval starting at 1 (corresponding to the positions in L covered by the echo strategy) and a sparse random set of expected size $O(\ell^2/n)$ (corresponding to positions in L covered by the random strategy). Together, these parts of J can be encoded using $O((\ell^2/n + 1) \log \ell)$ bits, in expectation.

Delaying the echo strategy. If we implemented the above random oracle algorithm as a random seed algorithm, we would need $\Omega(\ell)$ bits of space, just to store the random list L . But why does L need to have length $\ell + 1$? This length is needed for the algorithm to be resilient to the echo strategy, which covers one new element of L on every step; if L were shorter, the echo strategy could entirely cover it, making the algorithm run out of possible values to output. The random seed algorithm for $\text{MIF}(n, \ell)$ works by making the echo strategy less effective, ensuring that multiple inputs are needed for it to cover another element of L . It does this by partitioning $[n]$ into $\Theta(\ell)$ disjoint subsets (“blocks”) of size $\Theta(n/\ell)$, and then taking L to be a random list of blocks (rather than a random list of elements of $[n]$). We will now say that a block is “covered” if *any* element of that block was an input. Instead of outputting the first uncovered element in L , the algorithm will run a deterministic algorithm for MIF *inside* the block corresponding to the first uncovered block of L , and report outputs from that; and will only move on to the next uncovered block when the nested algorithm

stops. See Algorithm 1 for the details of this design. Because the analogue of the echo strategy now requires many more inputs to cover a block, we can make the list L shorter. This change will not make the random strategy much more effective.⁵ The minimum length of L is constrained by the $O(n/\ell)$ block sizes, which limit the number of outputs that the nested algorithm can make; as a result, one must have $L = \Omega(\ell^2/n)$. In the end, after balancing the length of the list with the cost of the nested algorithm, the optimal list length for the random seed algorithm will be $O(\ell^2/n + \sqrt{\ell})$.

■ **Algorithm 1** Example: recursive construction for a random tape $\text{MIF}(n, \ell)$ algorithm.

Parameter: $t \in [\Omega(\ell^2/n), \ell]$ is the number of parts into which the input stream is split

Initialization:

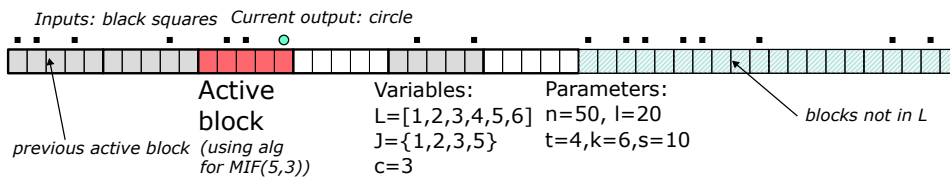
- 1: Let $k = O(t)$, $s = O(\ell)$, and B_1, \dots, B_s be a partition of $[n]$ into s equal “blocks”
- 2: $L \leftarrow$ uniformly randomly chosen sequence of k distinct elements of $[s]$
- 3: $J \leftarrow \emptyset$, is a subset of $[k]$ \triangleright a set marking which blocks of L have been covered
- 4: $c \leftarrow 1$ \triangleright the current active block
- 5: $A \leftarrow$ instance of algorithm \mathcal{A} solving $\text{MIF}(n/s, \lceil \ell/t \rceil)$

Update($a \in [n]$):

- 6: Let h be the block containing a , and x the rank of a in B_h
- 7: **if** $h \in L$ **then**
- 8: Add j to J , where $L_j = h$ \triangleright Mark list element containing h as used
- 9: **if** $h = L_c$ **then**
- 10: $A.\text{UPDATE}(x)$
- 11: **if** A is out of space **then** \triangleright This requires that $A.\text{UPDATE}()$ be called $\geq \lceil \ell/t \rceil$ times
- 12: $c \leftarrow$ least integer which is $> c$ and not in J \triangleright This line may abort if $J = [k]$
- 13: $A \leftarrow$ new instance of algorithm \mathcal{A} \triangleright Using new random bits, if \mathcal{A} is randomized

Output $\rightarrow [n]$:

- 14: Let $x \in [n/s]$ be the output of A
- 15: **return** x th entry of block B_c



■ **Figure 1** A diagram illustrating the state of an instance of Algorithm 1 on an example input. Positions on the horizontal axis correspond to integers in $[n]$; the set of values in the input stream ($\{1, 2, 4, 9, 12, 13, \dots\}$) is marked with black squares; the current output value (15) with a circle. Outside this example, L need not be contiguous or in sorted order.

The recursive random tape algorithm. The random seed algorithm for $\text{MIF}(n, \ell)$ used the construction of Algorithm 1 to build on top of an “inner” deterministic algorithm.⁶ To get an efficient random tape algorithm, we can recursively apply the construction of Algorithm 1

⁵ The fact that $[n]$ is split into $\Omega(\ell)$ blocks is enough to mitigate the random strategy; with ℓ guesses, the adversary is unlikely to guess more than a constant fraction of the elements in L .

⁶ The construction uses randomness in two places: when initializing the random sequence L , and (possibly) each time the inner algorithm is initialized. For the random seed model, every “inner” initialization

28:10 Finding Missing Items Requires Strong Forms of Randomness

$d - 1$ times, for $d = O(\min(\log \ell, \log n / \log \ell))$; at the end of this recursion, we can use a simple deterministic algorithm for MIF. The optimal lengths of the random lists used at each level of the recursion are determined by balancing the costs of the different recursion levels. We end up choosing list lengths that all bounded by a quantity which lies between $O(\ell^{1/d})$ and $O(\ell^{1/(d-1)})$.

In the extreme case where $d = \Theta(\log \ell)$ and the required error level δ is constant, our recursive algorithm may have a stack of random lists, each of length 2, and every time a level of the algorithm completes (i.e., all blocks of a list have been used), it will make a new instance of that level. That is, some large uncovered block will be split into many smaller blocks, and the algorithm will randomly pick two of them for the new instance's list. Because the lists are all short, the algorithm will not need to remember many random bits at a given time; in exchange, for this regime it needs a very large ($n = \ell^{\Omega(d)}$) number of possible outputs and will frequently need to sample new random lists.

The final version of our algorithm is given in the full version of the paper. It looks somewhat different from the recursive construction in Algorithm 1, because we have unraveled the recursive framing to allow for a simpler error analysis that must only bound the probability of a single “bad event.” The resulting space bound is:

► **Theorem 1.** *There is a family of adversarially robust random tape algorithms, where for $\text{MIF}(n, \ell)$ the corresponding algorithm has $\leq \delta$ error and uses*

$$O\left(\left\lceil \frac{(4\ell)^{\frac{2}{d-1}}}{(n/4)^{\frac{2}{d(d-1)}}} \right\rceil (\log \ell)^2 + \min(\ell, \log \frac{1}{\delta}) \log \ell\right)$$

bits of space, where $d = \max\left(2, \min\left(\lceil \log \ell \rceil, \left\lfloor 2 \frac{\log(n/4)}{\log(16\ell)} \right\rfloor\right)\right)$. At $\delta = 1/\text{poly}(n)$ this space bound is $O(\ell^{\log_n \ell} (\log \ell)^2 + \log \ell \log n)$.

4 Random Tape Lower Bound (Result 1)

The AVOID problem. At the core of many of the MIF lower bounds is the SUBSET AVOIDANCE communication problem, introduced in [8]. Here we have two players, Alice and Bob, and a known universe $[m]$: Alice has a set $A \subseteq [m]$ of size a , and should send a message (as short as possible) to Bob, who should use the message to output a set $B \subseteq [m]$ of size b which is disjoint from A . Henceforth, we'll call this problem $\text{AVOID}(m, a, b)$. [8] showed that both deterministic and constant-error randomized one-way protocols for this problem require $\Omega(ab/m)$ bits of communication. An adversarially robust z -space algorithm for $\text{MIF}(m, a + b)$ can be used as a subroutine to implement a z -bit one-way protocol for $\text{AVOID}(m, a, b)$, thereby proving $z = \Omega(ab/m)$. This immediately gives us an $\Omega(\ell^2/n)$ space lower bound for $\text{MIF}(n, \ell)$, which, as we have seen, is near-optimal in the robust, random oracle setting.

► **Lemma 2** (Adversarially robust random oracle lower bound, from [24]). *Any random oracle (or random seed) algorithm which solves $\text{MIF}(n, \ell)$ in the adversarial setting with total error $\leq \delta$ requires $\Omega(\ell^2/n + \log(1 - \delta))$ bits of space.*

The random tape lower bound. To prove stronger lower bounds that exploit the random tape limitation of the algorithm, we need a more sophisticated use of AVOID. Fix an adversarially robust, random tape, z -space algorithm \mathcal{A} for $\text{MIF}(n, \ell)$. Roughly speaking,

would require a corresponding set of random bits, which are counted toward the space cost of the algorithm. Using a deterministic inner algorithm avoids this cost.

while the random oracle argument used \mathcal{A} to produce an AVOID protocol at the particular scale $a = b = \ell$, for the fixed universe $[n]$, our random tape argument will “probe” \mathcal{A} in a recursive fashion – reminiscent of the recursion in our random tape upper bound – to identify a suitable scale and sub-universe at which an AVOID protocol can be produced. This probing will itself invoke the AVOID lower bound to say that if an $\text{AVOID}(m, a, b)$ protocol is built out of a z -space streaming algorithm where $z \ll a$, then B must be small, with size $b = O((z/a)m)$.

We will focus on the regime where $\delta = O(1/n)$. This error level requires a measure of structure from the algorithm: it cannot just pick a random output each step, because that would risk colliding with an earlier input with $\geq 1/n$ probability. Our recursive argument works by writing z , the space usage of \mathcal{A} , as a function of a space lower bound for $\text{MIF}(w, t)$, where $w = \Theta(zn/\ell)$ and $t = \Theta(n/z)$. For small enough z , $t^2/w \gg \ell^2/n$, so by repeating this reduction step a few times we can increase the ratio of the stream length to the input domain size until we can apply the simple $\Omega(\hat{\ell}^2/\hat{n})$ lower bound for $\text{MIF}(\hat{n}, \hat{\ell})$. With the right number of reduction steps, one obtains the lower bound formula of Theorem 8, of which Result 1 is a special case.

The reduction. The reduction step argues that the $\text{MIF}(n, \ell)$ algorithm \mathcal{A} “contains” a z -space algorithm for $\text{MIF}(w, t)$, which, on being given any $t = O(\ell/z)$ items in a certain sub-universe $W \subseteq [n]$ of size $w = O(zn/\ell)$, will repeatedly produce missing items *from that sub-universe*. That such a set W *exists* can be seen as a consequence of the lower bound for AVOID: if \mathcal{A} receives a random sorted subset S of $\ell/2$ elements in $[n]$, then because there are $\binom{n}{\ell/2}$ possible subsets, most of the 2^z states of \mathcal{A} will need to be “good” for $\Omega(2^{-z} \binom{n}{\ell/2})$ different subsets. In particular, upon reaching a given state σ , for \mathcal{A} to solve MIF with error probability $O(1/n)$, its outputs henceforth – for the next $\ell/2$ items in the stream – must avoid most of the sets of inputs that could have led it to σ . We will prove by a counting argument that after the random sequence S is sent, each state σ has an associated set H_σ of possible “safe” outputs which are unlikely to collide with the inputs from S , and that $|H_\sigma|$ is typically $O(zn/\ell)$. Thus, for a typical state σ , starting \mathcal{A} from σ causes its next $\ell/2$ outputs to be inside H_σ , w.h.p.; in other words, \mathcal{A} contains a “sub-algorithm” solving $\text{MIF}(O(zn/\ell), \ell/2)$ on the set $W = H_\sigma$.

However, even though there exists a set W on which \mathcal{A} will concentrate its outputs, it may not be possible for an adversary to find it. In particular, had \mathcal{A} been a random *oracle* algorithm, each setting of the random string might lead to a different value for W , making W practically unguessable. But \mathcal{A} is in fact a random *tape* algorithm, so we can execute the following strategy.

In our core lemma, Lemma 5, we design an adversary (Adversary 2) that can with $\Omega(1)$ probability identify a set W of size $\Theta(zn/\ell)$ for which the next $\Theta(\ell/z)$ outputs of \mathcal{A} will be contained in W , with $\Omega(1)$ probability, no matter what inputs the adversary sends next. In other words, our adversary will identify a part of the stream and a sub-universe of $[n]$ where the algorithm solves $\text{MIF}(\Theta(zn/\ell), \Theta(\ell/z))$. The general strategy is to use an iterative search based on a win-win argument. First, the adversary will send a stream comprising a random subset S of size $\ell/2$ to \mathcal{A} , to ensure that henceforth its outputs are contained in some (unknown) set H_ρ , where ρ is the (unknown) state reached by \mathcal{A} just after processing S . Because \mathcal{A} has $\leq 2^z$ states, from the adversary’s perspective there are $\leq 2^z$ possible candidates for H_ρ . Then, the adversary conceptually divides the rest of the stream to be fed to \mathcal{A} into $O(z)$ phases, each consisting of $t = O(\ell/z)$ stream items. In each phase, one of the following things happens.

1. There exists a “sub-adversary” (function to choose the t items constituting the phase, one by one) which will probably make \mathcal{A} output an item that rules out a constant fraction of the candidate values for H_ρ (output i rules out set J if $i \notin J$). The adversary then runs this sub-adversary.
2. No matter how the adversary picks the t inputs for this phase, there will be a set W (roughly, an “average” of the remaining candidate sets) that probably contains the corresponding t outputs of \mathcal{A} .

As the set of candidate sets can only shrink by a constant fraction $O(z)$ times, the first case can only happen $O(z)$ times, with high probability. Thus, eventually, the adversary will identify the set W that it seeks. Once it has done so, it will run the optimal adversary for $\text{MIF}(\Theta(zn/\ell), \Theta(\ell/z))$. This essentially reduces the lower bound for $\text{MIF}(\ell, n)$ to that for $\text{MIF}(\Theta(zn/\ell), \Theta(\ell/z))$.

4.1 Technical Details

Types of error. One subtlety is that we will need to carefully account for the probability that \mathcal{A} , over the next $\Theta(\ell/z)$ stream items, produces outputs outside W . This will require us to distinguish between two types of “errors” for the algorithm over those next $\Theta(\ell/z)$ items: an $O(1)$ chance of producing an output outside W , and a smaller chance of making a mistake per the definition of MIF, i.e., outputting an item that was not missing (cf. Definition 3).

► **Definition 3.** *An algorithm \mathcal{A} for $\text{MIF}(n, \ell)$ can fail in either of two ways. It may make an incorrect output, or mistake, if outputs an element that was already in the input stream. It may also abort, by outputting a special value \perp (or some other value which is not a possible input for MIF).*

This distinction is useful because, if we take an algorithm for $\text{MIF}(n, \ell)$, conditioned on producing some initial transcript of outputs in response to an input sequence, we may obtain an algorithm for $\text{MIF}(|W|, t)$ for some $t \leq \ell$ and $W \subseteq [n]$; the probability that the algorithm “aborts” (produces an output outside of W) can be much larger than the probability that the algorithm makes an incorrect output (output in W that collides with an earlier input). In the following proofs the algorithm aborting will be bad for the adversary, and making a mistake will be good.

For integers n, ℓ, z with $1 \leq \ell < n$, and $\gamma \in [0, 1]$, let $\text{Algs}(n, \ell, z, \gamma)$ be the set of all z -bit *random tape* algorithms for $\text{MIF}(n, \ell)$ which on *any* adversary abort with probability $\leq \gamma$. Define $\Delta(n, \ell, \gamma, z) := \min\{\delta(\mathcal{A}, n, \ell) : \mathcal{A} \in \text{Algs}(n, \ell, z, \gamma)\}$, where $\delta(\mathcal{A}, n, \ell)$ is the maximum probability, over all possible adversaries, that \mathcal{A} makes an incorrect output. As a consequence of the definition, $\Delta(n, \ell, \gamma, z)$ is non-increasing in γ and z .

Using this new notation, the $\Omega(\ell^2/n)$ lower bound for adversarially robust streaming algorithms from [24] (cf. Lemma 2) tells us:

► **Lemma 4.** *Random tape algorithms for $\text{MIF}(n, \ell)$ that do not abort often have high error if they use too little space: concretely,*

$$\Delta(n, \ell, \gamma, z) \geq \frac{1}{4} \mathbb{1}_{z \leq \ell^2 / (16n \ln 2)} \mathbb{1}_{\gamma \leq 1/2}. \quad (1)$$

Induction lemma. Our proof of Result 1 is inductive, with the above lemma being the base case. The induction step consists of a reduction, using an adaptive adversary described in Adversary 2 to prove a lower bound on the mistake probability. The next lemma formalizes the induction step.

► **Lemma 5.** Let $1 \leq \ell < n$ and z be integers, and $\gamma \in [0, \frac{1}{2}]$. Let k be an integer parameter for which $z \geq 2 \log(32k)$. Define, matching definitions in Adversary 2,

$$w = 2 \left\lfloor 32 \frac{zn}{\ell} \right\rfloor \quad \text{and} \quad t = \left\lfloor \frac{\ell}{64zk} \right\rfloor.$$

If $t < w$, then there is a distribution $\mu \in \Delta[0, 1]$ for which $\mathbb{E}_{G \sim \mu} G \leq \gamma + \frac{1}{4k}$ and

$$\Delta(n, \ell, \gamma, z) \geq \min \left(\frac{\ell}{2^7 nk}, \left(\frac{1}{2} - \frac{1}{4k} \right) \mathbb{E}_{G \sim \mu} \Delta(w, t, G, z) \right). \quad (2)$$

The adversary. The adversary (Adversary 2) used for Lemma 5 is rather complicated, and requires some additional definitions.

► **Definition 6.** Say \mathcal{A} is a random tape algorithm whose states are given by the set Σ , and Q is a subset of Σ , where each state in Q has an associated set H_σ . A sequence y in $[n]^t$ is said to be *divisive* for Q if $|\{\sigma \in Q : y \subseteq H_\sigma\}| \leq \frac{1}{2}|Q|$.

Say Υ is a t -length deterministic adversary. (That is, a function which maps sequences in $[n]^*$ of length between 0 and $t - 1$, inclusive, to values in $[n]$.) For any state $\sigma \in \Sigma$ of \mathcal{A} , let $G(\sigma, \Upsilon)$ be the random variable in $[n]^t$ which gives the output if we run \mathcal{A} , starting at state σ , against the adversary Υ . (If after processing a few inputs, the algorithm has output sequence $v \in [n]^*$, its next input will be $\Upsilon(v)$.) We define an adversary to be α -splitting for Q against a distribution $\mathcal{D} \in \Delta[\Sigma]$ if, when we choose a random state S from \mathcal{D} ,

$$\Pr[G(S, \Upsilon) \text{ is divisive for } Q] \geq \alpha.$$

When we run Adversary 2 against an algorithm \mathcal{A} , let ρ be the state of \mathcal{A} after v is sent. The proof of Lemma 5 is long and requires that we consider the probabilities of the following events:

- B_{REPEAT} occurs if \mathcal{A} produces an output in $[n] \setminus H_\rho$
- B_{BIG} occurs if the state ρ has $|H_\rho| > \frac{1}{2}w$
- $B_{\text{INCOMPLETE}}$ occurs if the adversary aborts without executing Line 18
- B_{ABORT} occurs if \mathcal{A} aborts *before* the adversary reaches Line 18
- R_{ABORT} occurs if \mathcal{A} “aborts” (either for real, or by making an output outside W') *while* the adversary is executing Line 18
- R_{ERROR} occurs if \mathcal{A} produces an incorrect output *while* the adversary is executing Line 18

Calculations. By repeatedly applying Lemma 5, we obtain the following:

► **Lemma 7.** Let $1 \leq \ell < n$. For any integer $k \geq 1$, say that z is an integer satisfying $z \leq \frac{1}{64k} \ell^{1/k}$. Then:

$$\Delta(n, \ell, 0, z) > \min \left(\frac{\ell}{2^{10} nk}, \frac{1}{2^{k+5}} \mathbb{1}_{z \leq L} \right) \quad \text{where} \quad L = \frac{1}{64k} \left(\frac{\ell^{k+1}}{n} \right)^{\frac{2}{k^2+3k-2}}. \quad (3)$$

Consequently, algorithms for MIF with $\leq \min(\frac{\ell}{2^{10} nk}, 2^{-(k+5)})$ error require $> L$ bits of space.

Lemma 7 implies a lower bound on z for z -bit algorithms with $< \frac{\ell}{2^{10} nk}$ error probability. Choosing the value of k which maximizes the lower bound on z , and doing some additional calculations, gives the following theorem:

⁶ For any sequence $v \in \text{SEQS}([n], \lceil \ell/2 \rceil)$, $P(v)(\sigma) = \Pr[\text{the state of } \mathcal{A} \text{ just after receiving } v \text{ is } \sigma]$

28:14 Finding Missing Items Requires Strong Forms of Randomness

■ **Adversary 2** An adversary for a random tape MIF(n, ℓ) algorithm, with parameter k .

Let: $w = 2 \lfloor 32 \frac{zn}{\ell} \rfloor$, $h_{\max} = 32zk$, and $t = \lfloor \frac{\ell}{2h_{\max}} \rfloor$

ADVERSARY

- 1: $v \leftarrow$ a uniformly random sequence in $\text{SEQS}([n], \lceil \ell/2 \rceil)$.
- 2: **send** v to the algorithm
- 3: Let \mathcal{G} be a distribution over functions of type $\text{SEQS}([n], \lceil \ell/2 \rceil) \rightarrow \Sigma$, so that when $F \sim \mathcal{G}$, the distribution of $F(v)$ equals the distribution of current algorithm states
- 4: Compute, for all $\sigma \in \Sigma$,

$$H_\sigma := \left\{ i \in [n] : \Pr_{X \sim \text{SEQS}([n], \lceil \ell/2 \rceil), F \sim \mathcal{G}} [i \in X \mid F(X) = \sigma] \leq \frac{\lceil \ell/2 \rceil}{4n} \right\}$$

- 5: Let $Q_0 = \{\sigma \in \Sigma : |H_\sigma| \leq \frac{1}{2}w\}$ ▷ *Have a $\geq 1 - 1/16k$ chance current alg. state is in Q_0*
- 6: **for** h in $1, \dots, h_{\max}$ **do**
- 7: Let \mathcal{D} be the distribution over alg. states conditioned on the transcript so far
- 8: **if** \exists a $1/(8k)$ -splitting t -length deterministic adversary Υ for Q_{h-1} given \mathcal{D} **then**
- 9: **run** Υ against the algorithm, and let $y \in [n]^t$ be the output
- 10: $Q_h \leftarrow \{\sigma \in Q_{h-1} : y \subseteq H_\sigma\}$ ▷ *Have a $\geq 1/(8k)$ chance that $|Q_h| \leq \frac{1}{2}|Q_{h-1}|$*
- 11: **if** $Q_h = \emptyset$ **then abort**
- 12: **else**
- 13: $W \leftarrow \{i \in [n] : |\{\sigma \in Q_{h-1} : i \in H_\sigma\}| \geq \frac{1}{2}|Q_{h-1}|\}$.
- 14: Let $W' \leftarrow W$ plus $w - |W|$ padding elements
- 15: Define sub-algorithm \mathcal{B} to behave like the given algorithm, conditioned on the exact transcript of inputs and outputs made so far
- 16: Let Ξ be an adversary maximizing the probability that \mathcal{B} makes an incorrect output. (This can be computed using brute-force search.)
- 17: ▷ *If \mathcal{B} produces an output outside of W' , we interpret this as \mathcal{B} having aborted, not as having made a mistake*
- 18: **run** adversary Ξ , sending t inputs in W'
- 19: **return**
- 20: **abort**

► **Theorem 8.** *Random tape δ -error adversarially robust algorithms for MIF(n, ℓ) require*

$$\Omega \left(\max_{k \in \mathbb{N}} \frac{1}{k} \left(\frac{\ell^{k+1}}{n} \right)^{\frac{2}{k^2+3k-2}} \right) = \Omega \left(\frac{\log \ell}{\log n} \ell^{\frac{15}{32} \log_n \ell} \right)$$

bits of space, for $\delta \leq \frac{1}{2^{10n}}$.

► **Remark.** The adversary of Adversary 2 runs in doubly exponential time, and requires knowledge of the algorithm. The former condition cannot be improved by too much: if one-way functions exist, one could implement the random oracle algorithm for MIF(n, ℓ) from [24] using a pseudo-random generator that fools all polynomial-time adversaries. One can also prove by minimax theorem that universal adversaries for (random tape or otherwise) MIF(n, ℓ) algorithms can not be used to prove any stronger lower bounds than the one for random oracle algorithms.

5 Random Seed Lower Bound (Result 2)

The adversary constructed above for our random tape lower bound can be seen as a significant generalization of the adversary used by [24] to prove a random seed lower bound conditioned on a (then conjectured) pseudo-deterministic lower bound. Indeed, [24]’s adversary against a z -space algorithm \mathcal{A} also proceeds in a number of phases, each of length $t = \Theta(\ell/z)$. In each step, either (1) it can learn some new information about the initial state of \mathcal{A} (the “random seed”), by sending \mathcal{A} a specific stream of inputs in $[n]^t$, looking at the resulting output, and ruling out the seed values that could not have produced the output; or (2) it cannot learn much information, because for any possible input stream in $[n]^t$, \mathcal{A} has an output that it produces with constant probability. Each time the adversary follows the case (1), a constant fraction of the $\leq 2^z$ seed values are ruled out. Therefore, either within $O(z)$ steps the adversary will exactly learn the seed, at which point it can perfectly predict \mathcal{A} ’s behavior, which lands us in case (2); or \mathcal{A} will not reveal much information about the seed in a given phase, which also puts us in case (2). Because case (2) means that \mathcal{A} behaves pseudo-deterministically, \mathcal{A} must use enough space to pseudo-deterministically solve $\text{MIF}(n, t)$.

► **Theorem 9** (from [24]). *Let $S_{1/3}^{PD}(n, \ell)$ give a space lower bound for a pseudo-deterministic algorithm for $\text{MIF}(n, \ell)$ with error $\leq 1/3$. Then an adversarially robust random seed algorithm with error $\delta \leq \frac{1}{6}$, if it uses z bits of space, must have $z \geq S_{1/3}^{PD}(n, \lfloor \frac{\ell}{2z+2} \rfloor)$.*

Thus, Result 2 follows as a corollary of Result 4, which we discuss next. More specifically, Theorem 10 follows by combining Theorem 9 with the pseudo-deterministic lower bound, and also applying Lemma 2, which is stronger in the regime $\ell \geq n^{2/3}$.

► **Theorem 10.** *Adversarially robust random seed algorithms for $\text{MIF}(n, \ell)$ with error $\leq \frac{1}{6}$ require space:*

$$\Omega\left(\frac{\ell^2}{n} + \sqrt{\ell/(\log n)^3} + \ell^{1/5}\right).$$

6 Pseudo-Deterministic Lower Bound (Result 4)

This proof generalizes [24]’s space lower bound for *deterministic* $\text{MIF}(n, \ell)$ algorithms, which we briefly explain. Fix a deterministic $\text{MIF}(n, \ell)$ algorithm \mathcal{A} that uses z bits of space. For each stream τ with length $|\tau| \leq \ell$, define F_τ to be the set of *all possible outputs* of \mathcal{A} corresponding to length- ℓ streams that have τ as a prefix. Let ρ be a stream such that $|\tau| + |\rho| \leq \ell$. Then, by definition, $F_{\tau \circ \rho} \subseteq F_\tau$; whereas, by the correctness of \mathcal{A} , $F_{\tau \circ \rho} \cap \rho = \emptyset$. Now consider the AVOID problem over the universe F_τ , for a fixed τ : if Alice gets $\rho \subseteq F_\tau$ as an input, she could send Bob the state σ of \mathcal{A} upon processing $\tau \circ \rho$, whereupon Bob could determine $F_{\tau \circ \rho}$ (by repeatedly running \mathcal{A} ’s state machine starting at σ), which would be a valid output.

Let us restrict this scenario to suffixes ρ of some fixed length t ; we’ll soon determine a useful value for t . By the above observations, were it the case that

$$\exists \tau \in [n]^{\leq \ell-t} \forall \rho \in [n]^t: |F_{\tau \circ \rho}| \geq \frac{1}{2}|F_\tau|, \quad (4)$$

we would have a z -bit protocol for AVOID($|F_\tau|, t, \frac{1}{2}|F_\tau|$). By [8]’s lower bound for AVOID, we would have $z \geq Ct$ for a universal constant C . On the other hand, if the opposite were true, i.e.,

28:16 Finding Missing Items Requires Strong Forms of Randomness

$$\forall \tau \in [n]^{\leq \ell-t} \exists \rho \in [n]^t: |F_{\tau \circ \rho}| < \frac{1}{2}|F_{\tau}|, \quad (5)$$

then, starting from the empty stream ϵ , we could add a sequence of length- ℓ suffixes ρ_1, \dots, ρ_d (where $d \leq \lfloor \ell/t \rfloor$) such that $|F_{\rho_1 \circ \dots \circ \rho_s}| < 2^{-d}|F_{\epsilon}| \leq 2^{-d}n$. Since \mathcal{A} must produce *some* output at time ℓ , this would be a contradiction for $d \geq \log n$. Thus, for a setting of $t = \Theta(\ell/\log n)$, situation (4) must occur, implying a lower bound of $z = \Omega(\ell/\log n)$.

Relaxing “all outputs” to “common outputs”. Examining the above argument closely shows where it fails for pseudo-deterministic algorithms. In constructing an AVOID protocol above, we needed the key property that F_{τ} can be determined from just the *state* of \mathcal{A} upon processing τ . For pseudo-deterministic algorithms, if we simply define F'_{τ} to be “the set of all *canonical* outputs at time ℓ for continuations of τ ,” we cannot carry out the above proof plan because this F'_{τ} cannot be computed reliably from a single state: given a random state σ associated to τ , on average a δ fraction of the outputs might be incorrect and have arbitrary values; even a single bad output could corrupt the union calculation!

To work around this issue, we replace F_{τ} with a more elaborate recursive procedure FINDCOMMONOUTPUTS, (or FCO for short) that computes the “most common outputs” at time ℓ for a certain distribution over continuations of τ . To explain this, let us imagine positions 1 through ℓ in the input stream as being divided into d contiguous “time intervals.” In the deterministic proof, these intervals were of length t each. Given a stream τ that occupies the first $d - k$ of these intervals, F_{τ} can be thought of as the output of a procedure FINDALLOUTPUTS (or FAO for short) where $\text{FAO}(\mathcal{A}, \tau, k)$ operates as follows: for each setting ρ of the $(d - k + 1)$ th time interval, call $\text{FAO}(\mathcal{A}, \tau \circ \rho, k - 1)$ and return the union of the sets so obtained. In the base case, $\text{FAO}(\mathcal{A}, \tau, 0)$ takes a stream $\tau \in [n]^{\ell}$ and returns the singleton set $\{\mathcal{A}(\tau)\}$. The deterministic argument amounts to showing that, with interval lengths $t = \Theta(z)$, the set $\text{FAO}(\mathcal{A}, \tau, k)$ has cardinality $\geq 2^k$; since $\text{FAO}(\mathcal{A}, \epsilon, d)$ has cardinality $\leq n$, this bounds $d \leq \log n$, which lower-bounds z .

For our pseudo-deterministic setting, we use time intervals as above and we design an analogous procedure $\text{FCO}(B, C, \tau, k)$ that operates on a function $B: [n]^{\ell} \rightarrow [n]$ (roughly corresponding to a MIF algorithm), a matrix C of random thresholds,⁷ and a stream τ of length $\leq \ell$ that occupies the first $d - k$ time intervals. The recursive structure of $\text{FCO}(B, C, \tau, k)$ is similar to FAO, but crucially, the sets computed by the recursive calls $\text{FCO}(B, C, \tau \circ \rho, k - 1)$ are used differently. Instead of simply returning their union, we use these sets to collect statistics about the outputs in $[n]$ and return only those that are sufficiently common. The thresholds in C control the meaning of “sufficiently common.”

The function B provided to FCO can be either the canonical output function Π of the given pseudo-deterministic algorithm \mathcal{B} or a deterministic algorithm $A \sim \mathcal{B}$ obtained by fixing the random coins of \mathcal{B} . We will show that:

- With high probability over C and the randomness of \mathcal{B} , FCO will produce the same outputs on Π and \mathcal{B} . In other words, FCO is robust to noise (i.e., to algorithm errors).
- When applied to the canonical algorithm, the cardinalities of the sets returned by FCO will grow exponentially with k . Equivalently, similar to $|F_{\tau}|$ from the deterministic proof, the cardinality of $\text{FCO}(B, C, \tau, k)$ will shrink exponentially as the length $|\tau|$ grows. Ultimately, this is proven by implementing AVOID using FCO on the actual algorithm as a subroutine. Critically, this implementation uses the fact that the recursive calls to FCO w.h.p. produce the same output on Π and \mathcal{B} .

⁷ The use of random thresholds is a standard trick for robustly computing quantities in the presence of noise.

- The argument can be carried out with all but one of the d time intervals being of length $\approx \Theta(z)$. If z were too small, d would be large enough that for the empty stream prefix we would have $|\text{FCO}(B, C, \epsilon, d)| > n$, which contradicts $\text{FCO}(\dots) \subseteq [n]$; this lets us derive a lower bound on z .

Error amplification and the case $n \gg \ell$. One technical issue that arises is that the correctness of FCO requires \mathcal{B} 's error probability to be as small as $1/n^{\Omega(\log n)}$. Fortunately, even if the original error probability was $1/3$, we can reduce it to the required level since pseudo-deterministic algorithms allow efficient error reduction by independent repetition. A second technical point is that a z -space pseudo-deterministic algorithm can be shown to have only $O(2^z)$ possible outputs; so if $n \gg \ell$, we can sometimes obtain a stronger lower bound by pretending that n is actually $O(2^z)$. This is formalized by a simple encoding argument.

6.1 Technical Details

Pseudocode. Pseudocode for FCO is given in Procedure 3. The procedure is parameterized by the interval lengths t_d, \dots, t_1 , the set S of all possible *canonical* outputs, and a series of output sizes w_d, \dots, w_1 , where $w_i = 2^{i-1}(t_1 + 1)$.

■ **Procedure 3** The procedure to compute a set for Lemma 11.

Let $t_1, \dots, t_d, w_1, \dots, w_d$ be integer parameters, and S the set of valid outputs

```

FINDCOMMONOUTPUTS( $B, C, x, k$ ) ▷ abbreviated as  $\text{FCO}(B, C, x, k)$ 
1: ▷ Inputs: function  $B: [n]^\ell \rightarrow [n]$ , matrix  $C \in [1, 2]^{d \times N}$ , stream prefix  $x \in [n]^{t_k + \dots + t_1}$ 
2: ▷ Output: a subset of  $S$  of size  $w_k$ 
3: if  $k = 1$  then
4:    $e_0 \leftarrow B(x \circ \langle 1, 1, \dots, 1 \rangle)$ 
5:   for  $i$  in  $1, \dots, t_1$  do
6:      $e_i \leftarrow B(x \circ \langle e_0, \dots, e_{i-1}, 1, \dots, 1 \rangle)$ 
7:   if  $e_0, \dots, e_{t_1}$  are all distinct then
8:     return  $\{e_0, e_1, \dots, e_{t_1}\}$  ▷ identify  $w_1$  distinct possible outputs
9:   return arbitrary subset of  $S$  of size  $w_1$  (failure)
10: else
11:   for each  $y \in \text{SEQS}([n], t_k)$  do
12:      $T_y \leftarrow \text{FINDCOMMONOUTPUTS}(B, C, x \circ y, k - 1)$  ▷ note  $|T_y| = w_{k-1}$ 
13:    $Q_0 \leftarrow T_{\langle 1, 2, \dots, t_k \rangle}$ 
14:   for  $h$  in  $1, 2, 3, 4$  do
15:     ▷ gather statistics and find common elements among the sets  $T_y$ 
16:     for each  $j \in S$  do
17:        $f_j^{(h)} \leftarrow |\{y \in \text{SEQS}(Q_{h-1}, t_k) : j \in T_y\}|$  ▷ count frequencies
18:      $\theta \leftarrow C_{k,h} w_{k-1} / (16|S|)$  ▷ set random threshold
19:      $P_h \leftarrow \{j \in S : f_j^{(h)} \geq \theta \binom{|Q_{h-1}|}{t_k}\}$  ▷ identify "sufficiently common" elements
20:      $Q_h \leftarrow Q_{h-1} \cup P_h$ 
21:     if  $|Q_h| \geq w_k$  then
22:       return the  $w_k$  smallest elements in  $Q_h$ 
23:   return arbitrary subset of  $S$  of size  $w_k$  (failure)

```

Central lemma. For a series of error probabilities with $1 > \varepsilon_d \gg \varepsilon_{d-1} \dots \gg \varepsilon_1 \approx 1/n^{\Omega(d)}$, we prove, by induction, the following lemma. It asserts that the set of common outputs is likely the same for the canonical function Π as it is for a random draw $A \sim \mathcal{B}$. It also asserts two other key properties of FCO. The lemma can be thought of as a “proof of correctness” of FCO.

► **Lemma 11.** *Let $k \in [d]$ and $x \in [n]^{t_d + \dots + t_{k+1}}$. Then FCO satisfies the following properties.*

1. $\Pr_{A \sim \mathcal{B}, C \sim [1,2]^{d \times N}}[\text{FCO}(A, C, x, k) = \text{FCO}(\Pi, C, x, k)] \geq 1 - \varepsilon_k$.
2. For all $C \in [1, 2]^d$, the set $\text{FCO}(\Pi, C, x, k)$ is disjoint from x and a subset of S .
3. For all $A: [n]^\ell \rightarrow [n]$ and $C \in [1, 2]^d$, $\text{FCO}(A, C, x, k)$ outputs a set of size w_k .

Its proof is split over a number of helper lemmas:

► **Lemma 12.** *Lemma 11 holds for $k = 1$.*

► **Lemma 13.** *Let $x \in [n]^{t_d + \dots + t_{k+1}}$. When computing $\text{FCO}(\Pi, C, x, k)$, in the h th loop iteration, if $|Q_{h-1}| < w_k$, then $|P_h \setminus Q_{h-1}| \geq \lceil \frac{1}{4} w_{k-1} \rceil$. Consequently, the procedure will return using Line 22, not Line 23.*

► **Lemma 14.** *For $k > 1$, $x \in [n]^{t_d + \dots + t_{k+1}}$, $\text{FCO}(\Pi, C, x, k)$ is disjoint from x and a subset of S ; and for all B, C, x, k , $\text{FCO}(B, C, x, k)$ outputs a set of size w_k .*

► **Lemma 15.** *For $k > 1$, and all $x \in [n]^{t_d + \dots + t_{k+1}}$,*

$$\Pr_{A \sim \mathcal{B}, C}[\text{FCO}(A, C, x, k) \neq \text{FCO}(\Pi, C, x, k)] \leq \varepsilon_k.$$

Using the central lemma. A consequence of Lemma 11 is that $\text{FCO}(\Pi, C, d)$ will output a set of size w_d ; this gives a lower bound on n . Solving for a lower bound on z gives:

► **Theorem 16.** *Pseudo-deterministic δ -error random oracle algorithms for MIF(n, ℓ) require*

$$\Omega \left(\min \left(\frac{\ell}{\log \frac{2n}{\ell}} + \sqrt{\ell}, \frac{\ell \log \frac{1}{2\delta}}{(\log \frac{2n}{\ell})^2 \log n} + \left(\ell \log \frac{1}{2\delta} \right)^{1/4} \right) \right)$$

bits of space when $\delta \leq \frac{1}{3}$. In particular, when $\delta = 1/\text{poly}(n)$ and $\ell = \Omega(\log n)$, this is:

$$\Omega \left(\frac{\ell}{(\log \frac{2n}{\ell})^2} + (\ell \log n)^{1/4} \right).$$

► **Remark.** For $\delta \leq 2^{-\ell}$, Theorem 16 reproduces the deterministic algorithm space lower bound for MIF(n, ℓ) from [24] within a constant factor.

References

- 1 Miklós Ajtai, Vladimir Braverman, T.S. Jayram, Sandeep Silwal, Alec Sun, David P. Woodruff, and Samson Zhou. The white-box adversarial data stream model. In *Proc. 41st ACM Symposium on Principles of Database Systems*, pages 15–27, 2022. doi:10.1145/3517804.3526228.
- 2 Sepehr Assadi, Amit Chakrabarti, Prantar Ghosh, and Manuel Stoeckl. Coloring in graph streams via deterministic and adversarially robust algorithms. In *Proc. 42nd ACM Symposium on Principles of Database Systems*, pages 141–153, 2023. doi:10.1145/3584372.3588681.
- 3 Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993. doi:10.1145/168588.168596.

- 4 Omri Ben-Eliezer, Talya Eden, and Krzysztof Onak. Adversarially robust streaming via dense-sparse trade-offs. In *Symposium on Simplicity in Algorithms (SOSA)*, pages 214–227, 2022. doi:10.1137/1.9781611977066.15.
- 5 Omri Ben-Eliezer, Rajesh Jayaram, David P. Woodruff, and Eylon Yogev. A framework for adversarially robust streaming algorithms. In *Proc. 39th ACM Symposium on Principles of Database Systems*, pages 63–80, 2020. doi:10.1145/3375395.3387658.
- 6 Omri Ben-Eliezer and Eylon Yogev. The adversarial robustness of sampling. In *Proc. 39th ACM Symposium on Principles of Database Systems*, pages 49–62. ACM, 2020. doi:10.1145/3375395.3387643.
- 7 Vladimir Braverman, Robert Krauthgamer, Aditya Krishnan, and Shay Sapir. Lower bounds for pseudo-deterministic counting in a stream. *arXiv preprint arXiv:2303.16287*, 2023. doi:10.48550/arXiv.2303.16287.
- 8 Amit Chakrabarti, Prantar Ghosh, and Manuel Stoeckl. Adversarially robust coloring for graph streams. In *Proc. 13th Conference on Innovations in Theoretical Computer Science*, pages 37:1–37:23, 2022. doi:10.4230/LIPIcs.ITCS.2022.37.
- 9 Amit Chakrabarti and Manuel Stoeckl. Finding missing items requires strong forms of randomness. *arXiv preprint*, 2024. arXiv:2310.03634.
- 10 Uriel Feige. A randomized strategy in the mirror game. *arXiv preprint*, 2019. doi:10.48550/arXiv.1901.07809.
- 11 Sumegha Garg and Jon Schneider. The Space Complexity of Mirror Games. In *Proc. 10th Conference on Innovations in Theoretical Computer Science*, pages 36:1–36:14, 2018. doi:10.4230/LIPIcs.ITCS.2019.36.
- 12 Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-Deterministic Streaming. In *Proc. 20th Conference on Innovations in Theoretical Computer Science*, volume 151, pages 79:1–79:25, 2020. doi:10.4230/LIPIcs.ITCS.2020.79.
- 13 Ofer Grossman, Meghal Gupta, and Mark Sellke. Tight space lower bound for pseudo-deterministic approximate counting. *arXiv preprint*, 2023. doi:10.48550/arXiv.2304.01438.
- 14 Moritz Hardt and David P. Woodruff. How robust are linear sketches to adaptive inputs? In *Proc. 45th Annual ACM Symposium on the Theory of Computing*, pages 121–130, 2013. doi:10.1145/2488608.2488624.
- 15 Avinatan Hassidim, Haim Kaplan, Yishay Mansour, Yossi Matias, and Uri Stemmer. Adversarially robust streaming algorithms via differential privacy. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL: <https://proceedings.neurips.cc/paper/2020/hash/0172d289da48c48de8c5ebf3de9f7ee1-Abstract.html>.
- 16 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006. doi:10.1145/1147954.1147955.
- 17 Rajesh Jayaram and David P Woodruff. Towards optimal moment estimation in streaming and distributed models. *ACM Trans. Alg.*, 19(3):1–35, 2023. doi:10.1145/3596494.
- 18 Haim Kaplan, Yishay Mansour, Kobbi Nissim, and Uri Stemmer. Separating adaptive streaming from oblivious streaming using the bounded storage model. In *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 94–121. Springer, 2021. doi:10.1007/978-3-030-84252-9_4.
- 19 Roey Magen and Moni Naor. Mirror games against an open book player. In *11th International Conference on Fun with Algorithms (FUN 2022)*, volume 226, pages 20:1–20:12, 2022. doi:10.4230/LIPIcs.FUN.2022.20.
- 20 Boaz Menuhin and Moni Naor. Keep that card in mind: Card guessing with limited memory. In *Proc. 13th Conference on Innovations in Theoretical Computer Science*, pages 107:1–107:28, 2022. doi:10.4230/LIPIcs.ITCS.2022.107.
- 21 Ilan Newman. Private vs. common random bits in communication complexity. *Inform. Process. Lett.*, 39(2):67–71, 1991. doi:10.1016/0020-0190(91)90157-D.

28:20 Finding Missing Items Requires Strong Forms of Randomness

- 22 Noam Nisan. Pseudorandom generators for space-bounded computation. In *Proc. 22nd Annual ACM Symposium on the Theory of Computing*, pages 204–212, 1990. doi:10.1145/100216.100242.
- 23 Noam Nisan. On read once vs. multiple access to randomness in logspace. *Theoretical Computer Science*, 107(1):135–144, 1993. doi:10.1016/0304-3975(93)90258-U.
- 24 Manuel Stoeckl. Streaming algorithms for the missing item finding problem. In *Proc. 34th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 793–818, 2023. Full version at [arXiv:2211.05170v1](https://arxiv.org/abs/2211.05170v1). doi:10.1137/1.9781611977554.ch32.
- 25 Manuel Stoeckl. *On adaptivity and randomness for streaming algorithms*. PhD thesis, Dartmouth College, 2024. URL: <https://digitalcommons.dartmouth.edu/dissertations/229/>.
- 26 David P. Woodruff and Samson Zhou. Tight bounds for adversarially robust streams and sliding windows via difference estimators. In *Proc. 62nd Annual IEEE Symposium on Foundations of Computer Science*, pages 1183–1196, 2022. doi:10.1109/FOCS52979.2021.00116.

Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

Shuichi Hirahara  

National Institute of Informatics, Tokyo, Japan

Valentine Kabanets 

Simon Fraser University, Burnaby, Canada

Zhenjian Lu  

University of Warwick, UK

Igor C. Oliveira  

University of Warwick, UK

Abstract

A *search-to-decision* reduction is a procedure that allows one to find a solution to a problem from the mere ability to decide when a solution exists. The existence of a search-to-decision reduction for time-bounded Kolmogorov complexity, i.e., the problem of checking if a string x can be generated by a t -time bounded program of description length s , is a long-standing open problem that dates back to the 1960s.

In this work, we obtain new average-case and worst-case search-to-decision reductions for the complexity measure K^t and its randomized analogue rK^t :

1. (Conditional Errorless and Error-Prone Reductions for K^t) Under the assumption that E requires exponential size circuits, we design polynomial-time average-case search-to-decision reductions for K^t in both errorless and error-prone settings.

In fact, under the easiness of deciding K^t under the uniform distribution, we obtain a search algorithm for any given polynomial-time samplable distribution. In the error-prone reduction, the search algorithm works in the more general setting of conditional K^t complexity, i.e., it finds a minimum length t -time bound program for generating x given a string y .

2. (Unconditional Errorless Reduction for rK^t) We obtain an unconditional polynomial-time average-case search-to-decision reduction for rK^t in the errorless setting. Similarly to the results described above, we obtain a search algorithm for each polynomial-time samplable distribution, assuming the existence of a decision algorithm under the uniform distribution.

To our knowledge, this is the first unconditional sub-exponential time search-to-decision reduction among the measures K^t and rK^t that works with respect to any given polynomial-time samplable distribution.

3. (Worst-Case to Average-Case Reductions) Under the errorless average-case easiness of deciding rK^t , we design a worst-case search algorithm running in time $2^{O(n/\log n)}$ that produces a minimum length randomized t -time program for every input string $x \in \{0, 1\}^n$, with the caveat that it only succeeds on some explicitly computed sub-exponential time bound $t \leq 2^{n^\epsilon}$ that depends on x . A similar result holds for K^t , under the assumption that E requires exponential size circuits.

In these results, the corresponding search problem is solved *exactly*, i.e., a successful run of the search algorithm outputs a t -time bounded program for x of minimum length, as opposed to an approximately optimal program of slightly larger description length or running time.

2012 ACM Subject Classification Theory of computation \rightarrow Computational complexity and cryptography

Keywords and phrases average-case complexity, Kolmogorov complexity, search-to-decision reductions

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.29

Related Version *Full Version:* <https://ecc.weizmann.ac.il/report/2024/059>



© Shuichi Hirahara, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira;

licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 29; pp. 29:1–29:56

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



COMPUTATIONAL
COMPLEXITY
CONFERENCE

Funding This work received support from the Royal Society University Research Fellowship URF\R1\191059; the UKRI Frontier Research Guarantee Grant EP/Y007999/1; and the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick.

1 Introduction

The time-bounded Kolmogorov complexity $K^t(x)$ of an input binary string x is defined as the length of a shortest program that prints out x within t time steps. The corresponding *search* version of the problem would be to find such a shortest program that prints x within time t . Both problems have been studied since the 1960s, and are conjectured to require brute-force (trivial) algorithms to solve them [32]. The existence of an efficient search-to-decision reduction for K^t , i.e., an algorithm to solve the search version of $K^t(x)$ assuming an algorithm for the decision version of computing $K^t(x)$, is also an old open problem going back to the 1960s.¹ In fact, it is consistent with current knowledge that there might exist an algorithm that computes $K^t(x)$ in time linear in $n = |x|$, while any search algorithm for the problem requires time $2^{\Omega(n)}$.

In this work, we obtain new *average-case* and *worst-case* search-to-decision reductions for the measure K^t and its randomized analogue rK^t , which considers the length of the shortest randomized program that prints x with high probability within time t . Our search algorithms have two important features:

- they solve the search problem *exactly*: they find an optimally minimal-size program to print x within t steps (rather than an approximately optimal program of slightly larger size or running in slightly bigger than t time); and
- they succeed with high probability on any given *polynomial-time samplable* distribution (rather than being restricted to the uniform distribution).

It should be noted that such search-to-decision reductions are *necessary* for excluding Pessiland from Impagliazzo’s five worlds [18], that is, for basing the security of a one-way function on the average-case hardness of NP. By the result of Liu and Pass [22], the existence of a one-way function is characterized by the average-case hardness of computing time-bounded Kolmogorov complexity over the uniform distribution. If Pessiland is eliminated, it follows that the average-case easiness of time-bounded Kolmogorov complexity implies that every NP search problem is easy on any polynomial-time samplable distribution [19, 3], and in particular, the search problems of finding short programs are also easy. Thus, designing such reductions can be seen as a progress towards excluding Pessiland from Impagliazzo’s five worlds.

We describe our results in the next section. We compare them with the existing literature on exact and approximate search-to-decision reductions in Section 1.2.

1.1 Results

Informally, our main results give polynomial-time algorithms for solving the search versions of time-bounded Kolmogorov complexity measures K^t and rK^t , on *average* with respect to any given *polynomial-time samplable distribution*, under the assumption that the corresponding decision versions are easy on average with respect to the *uniform distribution*. For rK^t , such a

¹ One reason why such search-to-decision reductions may be possible to K^t is that the decision version of K^t is conjectured to be NP-complete, and efficient search-to-decision reductions for NP-complete problems are easy to get.

search-to-decision average-case reduction is unconditional, whereas for K^t we make a standard derandomization assumption. Our reduction for rK^t works in the *errorless* average-case setting. Our (conditional) reductions for K^t work in both *errorless* and *error-prone* settings.

We provide a more detailed description of our results in the following subsections.

1.1.1 Average-Case Search-to-Decision for K^t

Below we employ standard definitions of $K^t(x)$ and $rK^t(x)$, reviewed in Section 2.1. We let U denote the fixed universal Turing machine used in these definitions.

Let MINKT be the following *decision* problem: Given $(x, 1^s, 1^t)$, where $x \in \{0, 1\}^*$ and $s, t \in \mathbb{N}$, decide whether $K^t(x) \leq s$. Let Search-MINKT be the corresponding *search* problem: Given $(x, 1^t)$, where $x \in \{0, 1\}^*$ and $t \in \mathbb{N}$, find a K^t -witness of x , i.e., a program $M \in \{0, 1\}^*$ such that $|M| = K^t(x)$ and $U(M)$ outputs x within t steps.

In our average-case search-to-decision reductions for K^t we consider both *errorless* and *error-prone* settings, which correspond to the average-case complexity classes AvgBPP and HeurBPP, respectively (cf. [4]).

The Errorless Setting. We shall use “MINKT \in AvgBPP”, as an abbreviation for the statement that MINKT can be solved in polynomial time on average *without errors* over polynomial-time samplable distributions. Similarly, we shall use “Search-MINKT \in AvgBPP” to state that Search-MINKT can be solved in polynomial time on average *without errors* over polynomial-time samplable distributions. More formally, we have the following definitions.²

“MINKT \in AvgBPP”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a polynomial-time algorithm A such that the following holds for all $n, s, k \in \mathbb{N}$, and all $t \geq \rho(n)$.

1. For all $x \in \{0, 1\}^n$, $A(x, 1^s, 1^t, 1^k)$ outputs either MINKT($x, 1^s, 1^t$) or \perp , and
2. $\Pr_{x \sim \mathcal{D}_n} [A(x, 1^s, 1^t, 1^k) = \text{MINKT}(x, 1^s, 1^t)] \geq 1 - \frac{1}{k}$.

“Search-MINKT \in AvgBPP”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a polynomial-time algorithm A such that the following holds for all $n, k \in \mathbb{N}$, and all $t \geq \rho(n)$.

1. For all $x \in \{0, 1\}^n$, $A(x, 1^t, 1^k)$ outputs either a K^t -witness of x or \perp , and
2. $\Pr_{x \sim \mathcal{D}_n} [A(x, 1^t, 1^k) \text{ outputs a } K^t\text{-witness of } x] \geq 1 - \frac{1}{k}$.

Before stating our first result, we recall the following widely believed complexity-theoretic assumption. We use $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ to denote that there is a language $L \in E$ and $\varepsilon > 0$ such that L requires Boolean circuits of size at least $2^{\varepsilon n}$ on every large enough input length n .

► **Theorem 1** (Errorless Average-Case Search-to-Decision for K^t). *Assume $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. Then*

$$\text{“MINKT} \in \text{AvgBPP”} \implies \text{“Search-MINKT} \in \text{AvgBPP”}.$$

² In [4], AvgBPP denotes the class of all the pairs (L, \mathcal{D}) of problems L and distributions \mathcal{D} that admit randomized average-polynomial-time algorithms (or, equivalently, randomized errorless heuristic schemes). Our statement “MINKT \in AvgBPP” deviates from this standard notation in that (1) we abbreviate the input distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n \in \mathbb{N}}$, and (2) the lower bound $\rho(n)$ of the time parameter t depends on the distribution \mathcal{D} .

The Error-Prone Setting. Theorem 1 shows an average-case search-to-decision reduction for MINKT in the *errorless* setting, under the assumption that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. Can we also get a similar reduction in the *error-prone* setting? It turns out that such a search-to-decision reduction is implicit in a recent work by Liu and Pass [23]. However, it requires a stronger assumption saying that $E \not\subseteq \text{i.o.NSIZE}[2^{o(n)}]$, i.e., that there is a language in E that requires *non-deterministic* circuits of exponential size. We discuss this in more detail next.

We define “MINKT \in HeurBPP” and “Search-MINKT \in HeurBPP” to be the analogs of “MINKT \in AvgBPP” and “Search-MINKT \in AvgBPP”, respectively, but in the regime where errors are allowed.³

“MINKT \in HeurBPP”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, with each \mathcal{D}_n over $\{0, 1\}^n$, there is a polynomial ρ and a polynomial-time algorithm A such that for all $n, s, k \in \mathbb{N}$, and all $t \geq \rho(n, k)$, $\Pr_{x \sim \mathcal{D}_n} [A(x, 1^s, 1^t, 1^k) = \text{MINKT}(x, 1^s, 1^t)] \geq 1 - \frac{1}{k}$.

“Search-MINKT \in HeurBPP”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, with each \mathcal{D}_n over $\{0, 1\}^n$, there is a polynomial ρ and a polytime algorithm A such that for all $n, k \in \mathbb{N}$, and all $t \geq \rho(n, k)$, $\Pr_{x \sim \mathcal{D}_n} [A(x, 1^t, 1^k)$ outputs a K^t -witness of x] $\geq 1 - \frac{1}{k}$.

As noted above, [23] proved “MINKT \in HeurBPP” \implies “Search-MINKT \in HeurBPP”, assuming $E \not\subseteq \text{i.o.NSIZE}[2^{o(n)}]$. We strengthen their result by weakening their assumption to $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. Combined with Theorem 1, this yields average-case search-to-decision reductions for MINKT in both errorless and error-prone settings, under the assumption that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$.

In fact, we show an even stronger result where we solve the *conditional* variant of the search problem, Search-MINcKT, on average over polynomial-time samplable distributions, while using the same assumption on the decision problem. We describe this in more detail below.

For $x, y \in \{0, 1\}^*$, we say that a program Π is a $K^t(\cdot | y)$ -witness of x if $|\Pi| = K^t(x | y)$ and $U(\Pi, y)$ outputs x within t steps.

► **Theorem 2** (Error-Prone Average-Case Search-to-Decision for Conditional K^t). *Assume $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If “MINKT \in HeurBPP” holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_{\langle n, m \rangle}\}_{n, m \in \mathbb{N}}$ supported over $\{0, 1\}^n \times \{0, 1\}^m$, there exist a polynomial ρ and a polynomial-time algorithm A such that for all $n, m, k \in \mathbb{N}$, and all $t \geq \rho(n, m, k)$,*

$$\Pr_{(x, y) \sim \mathcal{D}_{\langle n, m \rangle}} [A(x, y, 1^t, 1^k) \text{ outputs a } K^t(\cdot | y)\text{-witness of } x] \geq 1 - \frac{1}{k}.$$

Note that Theorem 2 implies a search-to-decision reduction for K^t (without the conditional string) by considering the set of polynomial-time samplable distribution families $\{\mathcal{D}_{\langle n, m \rangle}\}_{n, m \in \mathbb{N}}$ restricted to $m = 0$.

While the search-to-decision reductions from Theorems 1 and 2 rely on the assumption $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, we remark that it is possible to obtain weaker *unconditional* variants of these results using a simple win-win argument. Indeed, if the assumption does not hold then we can solve the corresponding search problem on infinitely many input lengths using

³ For technical reasons, in the error-prone setting we let the function ρ depend on k in addition to n . (See Remark 31). Obtaining a reduction without this dependence (as in the errorless setting) is an interesting problem.

circuits of size $2^{o(n)}$ (see Appendix C for an implementation of this idea). Consequently, it follows that there are errorless and error-prone search-to-decision reductions for K^t computed by sub-exponential size Boolean circuits, on infinitely many input lengths.

1.1.2 Average-Case Search-to-Decision for rK^t

We use $rK_\lambda^t(x)$ to denote the minimum length of a randomized program that outputs x with probability at least λ within t steps (see Section 2.1). We often omit λ in informal discussions, tacitly assuming that $\lambda = 2/3$.

Analogously to MINKT, one can also consider the problem of deciding whether $rK^t(x) \leq s$ given $(x, 1^s, 1^t)$. However, this problem is not very “natural” in the sense that it can only be placed in the class $\exists \cdot \text{PP}$. This is because the precise computation of the acceptance probability of a given machine is a computationally hard counting problem.

Here, we consider a more robust variant, which we call MINrKT, that can be shown to be in (promise) MA. We will then focus on the search version of MINrKT.

Let MINrKT be the following promise problem (YES, NO):

$$\begin{aligned} \text{YES} &:= \{(x, \lambda, 1^s, 1^t, 1^\ell) \mid rK_\lambda^t(x) \leq s\}, \\ \text{NO} &:= \{(x, \lambda, 1^s, 1^t, 1^\ell) \mid rK_{\lambda-1/\ell}^t(x) > s\}. \end{aligned}$$

Next, we describe the search version of MINrKT. We first need some notation. For $x \in \{0, 1\}^n$, $t \in \mathbb{N}$ and $0 < \varepsilon, \lambda \leq 1$, we say that a program M is an ε - rK_λ^t -witness of x if

- $|M| \leq rK_\lambda^t(x)$, and
- $U(M, r)$ outputs x within t steps with probability at least $\lambda - \varepsilon$ over $r \sim \{0, 1\}^t$.

Let Search-MINrKT be the following search problem: Given $(x, \lambda, 1^t, 1^\ell)$, where $x \in \{0, 1\}^*$, $t, \ell \in \mathbb{N}$ and $\lambda \in [0, 1]$, find an $(1/\ell)$ - rK_λ^t -witness of x .

We introduce the statement “MINrKT \in AvgBPP”, which states that MINrKT can be solved in probabilistic polynomial time on average (without errors) over polynomial-time samplable distributions. We first need to specify what it means by solving a *promise* problem in the average-case setting. For an algorithm A , $x \in \{0, 1\}^*$, $\lambda \in [0, 1]$, and $\ell, t, s \in \mathbb{N}$, we say that A *decides* MINrKT on $(x, \lambda, 1^s, 1^t, 1^\ell)$ if the following holds:

$$A(x, \lambda, 1^s, 1^t, 1^\ell) = \begin{cases} 1 & \text{if } rK_\lambda^t(x) \leq s \\ 0 & \text{if } rK_{\lambda-1/\ell}^t(x) > s \\ \text{either 0 or 1} & \text{otherwise.} \end{cases}$$

For $\lambda \in \mathbb{R}$, we denote by $|\lambda|$ the bit complexity of λ .

“MINrKT \in AvgBPP”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a probabilistic polynomial-time algorithm A such that the following hold for all $\lambda \in (0, 1)$, all $n, s, \ell, k \in \mathbb{N}$, and all $t \geq \rho(n) \cdot \log(1/(1 - \lambda))$.

1. For all $x \in \{0, 1\}^n$,

$$\Pr_A[A \text{ decides MINrKT on } (x, \lambda, 1^s, 1^t, 1^\ell) \text{ OR } A(x, \lambda, 1^s, 1^t, 1^\ell, 1^k) = \perp] \geq \frac{2}{3}.$$

2. With probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\Pr_A[A \text{ decides MINrKT on } (x, \lambda, 1^s, 1^t, 1^\ell)] \geq \frac{2}{3}.$$

We also introduce the statement “SearchMINrKT \in AvgBPP”, which states that SearchMINrKT can be solved in probabilistic polynomial time on average (without errors) over polynomial-time samplable distributions (cf. the definition of AvgBPP from [4]).

“**SearchMINrKT \in AvgBPP**”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a probabilistic polynomial-time algorithm A such that the following hold for all $\lambda \in (0, 1)$, all $n, s, \ell, k \in \mathbb{N}$, and all $t \geq \rho(n) \cdot \log(1/(1 - \lambda))$.

1. For all $x \in \{0, 1\}^n$,

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) \text{ outputs either an } (1/\ell)\text{-rK}_\lambda^t\text{-witness of } x \text{ or } \perp] \geq 1 - \frac{1}{2^k}.$$

2. With probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) \text{ outputs an } (1/\ell)\text{-rK}_\lambda^t\text{-witness of } x] \geq 1 - \frac{1}{2^k}.$$

► **Theorem 3** (Errorless Average-Case Search-to-Decision for rK^t). *We have*

$$\text{“MINrKT} \in \text{AvgBPP”} \implies \text{“SearchMINrKT} \in \text{AvgBPP”}^4$$

In contrast to our main results for K^t (Theorems 1 and 2), the search-to-decision reduction for rK^t is unconditional in that it does not rely on a circuit lower bound assumption. To our knowledge, this is the first unconditional reduction for the measures K^t and rK^t that runs in less than exponential time and that works with respect to all polynomial-time samplable distributions.

1.1.3 Worst-Case to Average-Case Search-to-Decision

In our next results, we aim to obtain a *worst-case* search algorithm from the same *average-case* easiness assumptions considered before. Note that this is significantly more challenging than a typical (worst-case to worst-case) search-to-decision reduction.

► **Theorem 4** (Conditional Worst-Case to Average-Case Search-to-Decision for K^t). *Assume $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If “MINKT \in AvgBPP” holds, then for every $\varepsilon > 0$ and every polynomial β , there is an algorithm A such that for all $n \in \mathbb{N}$ and $x \in \{0, 1\}^n$, $A(x)$ runs in time $2^{O(n/\log n)}$ and outputs a program M and an integer t that satisfy the following:*

- $\beta(n) \leq t \leq 2^{n^\varepsilon}$, and
- M is a K^t -witness of x .

► **Theorem 5** (Worst-Case to Average-Case Search-to-Decision for rK^t). *If “MINrKT \in AvgBPP” holds, every polynomial β , there is a probabilistic algorithm A such that for all $n \in \mathbb{N}$, $x \in \{0, 1\}^n$, all $\ell \in \mathbb{N}$, and all $\lambda \in (0, 1)$ such that $\lambda \leq 1 - 1/2^{\text{poly}(n)}$, $A(x, \lambda, 1^\ell)$ runs in time $2^{O(n/\log n)} \cdot \text{poly}(|\lambda|, \ell)$ and, with probability at least $1 - 2^{-\ell}$, outputs a program M and an integer t that satisfy the following:*

- $\beta(n) \leq t \leq 2^{n^\varepsilon}$, and
- M is an $(1/\ell)\text{-rK}_\lambda^t$ -witness of x .

In both results, we obtain a sub-exponential time search algorithm that works on every input string x . A caveat is that we have no control over the value of t on which the search algorithm succeeds, while ideally we would like it to succeed on every choice of t presented as an extra input parameter. On the positive side, in both results we make only an *average-case* easiness assumption on the decision problem, i.e., we obtain an interesting worst-case conclusion from a significantly weaker computational assumption.

1.1.4 Weaker Assumptions on the Decision Problems

In fact, for the results stated above, a much weaker assumption on the decision problem suffices to get the same consequence on the search problem. This is a consequence of the nature of our techniques, which we discuss in Section 1.3 below. Consider the following statements.

(MINKT, \mathcal{U}) \in HeurBPP: There exist a polynomial ρ and a polynomial-time algorithm A such that for all $n, s, k \in \mathbb{N}$, and all $t \geq \rho(n, k)$, $\Pr_{x \sim \{0,1\}^n} [A(x, 1^s, 1^t, 1^k) = \text{MINKT}(x, 1^s, 1^t)] \geq 1 - \frac{1}{k}$.

(coMINKT, \mathcal{U}) \in Avg¹BPP: There exist a constant $c > 0$, a polynomial ρ and a probabilistic polynomial-time algorithm A such that the following hold for all sufficiently large n , all $t \geq \rho(n)$, and all $s \leq n - c \cdot \log \log t$.

1. For every $x \in \{0, 1\}^n$ with $K^t(x) \leq s$, we have $\Pr_A[A(x, 1^s, 1^t) = 1] \geq 2/3$.
2. With probability at least $1/n$ over $x \sim \{0, 1\}^n$, we have $\Pr_A[A(x, 1^s, 1^t) = 0] \geq 2/3$.

It turns out that, as shown in the body of the paper, these weaker assumptions (see Proposition 11) suffice in the following search-to-decision reductions:

- Theorems 1 and 4 still hold if replacing “MINKT \in AvgBPP” with $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$.
- Theorems 3 and 5 still hold if replacing “MINrKT \in AvgBPP” with $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$.
- Theorem 2 still holds if replacing “MINKT \in HeurBPP” with $(\text{MINKT}, \mathcal{U}) \in \text{HeurBPP}$.

Consequently, in our search-to-decision reductions the existence of a decision algorithm for the *uniform* distribution provides a search algorithm for any *polynomial-time samplable* distribution.

1.2 Related Work

We now compare our results with prior work on search-to-decision reductions for time-bounded Kolmogorov complexity.

Approximate Reductions. Many previous results on search-to-decision for time-bounded Kolmogorov complexity have focused on approximate reductions (also known as gap reductions), where there is a weaker guarantee on the output of the search algorithm. More precisely, for a string $x \in \{0, 1\}^n$ such that $K^t(x) = s$, the search algorithm is allowed to output a program with the running time $t' \approx t$ and the program size $s' \approx s$.

In a recent development, [30] obtained a worst-case approximate reduction that produces a program with $t' = \text{poly}(|x|, t, s)$, $s' \leq s + \log \text{poly}(|x|, t, s)$, and that runs in randomized time $2^{\varepsilon \cdot s} \cdot \text{poly}(|x|, t, s)$, for an arbitrarily small $\varepsilon > 0$. An advantage of the approximate reduction of [30] with respect to our exact reductions is that it invokes the decision algorithm in a black-box way, while our techniques require access to the code of the decision algorithm.

While approximate reductions are not the focus of this work, we note that some of our techniques can be used to obtain a *polynomial-time* reduction with similar parameters t' and s' , under the assumption that \mathbf{E} requires exponential size circuits. Although predicated on a hardness assumption, our search-to-decision reduction has essentially the best possible runtime. We refer to Appendix C for the details.

A statement related to the average-case to worst-case search-to-decision reduction for K^t (Theorem 4) appears in [13, Theorem 8.7]. Both results are conditional. However, in contrast to Theorem 4, where the search algorithm produces an *exact* solution, in [13, Theorem 8.7] the search algorithm outputs an *approximate* solution where $s' = s = K^t(x)$ but t' can be as large as $2^{n/\log n}$ for $t \leq 2^{n^{0.99}}$.

For the time-bounded Kolmogorov complexity measures Kt and rKt , [25, 27] designed efficient reductions with $s' = O(s)$ such that, on a given input string x , the search algorithm only queries the decision algorithm on x .

In these approximate reductions, it is often possible to relax the requirement on the decision algorithm, i.e., the reduction still works when the latter only approximates $K^t(x)$. Interestingly, this is also the case in our results as a consequence of the discussion in Section 1.1.4, though we obtain an exact solution to the search problem even under a relaxation of the decision algorithm.

Exact Average-Case Reductions. [22] (see also the alternate proof in [30]) established the first error-prone polynomial-time search-to-decision reduction for K^t over the uniform distribution. Another related result appears in [24], which showed that if polynomial-time symmetry of information holds for K^t (i.e., if $K^t(x, y) \approx K^{t^{O(1)}}(x) + K^{t^{O(1)}}(y | x)$), then **Search-MINKT** admits an error-prone polynomial-time algorithm over the uniform distribution. In contrast, here we obtain both error-prone and errorless reductions for K^t for every given *polynomial-time samplable* distribution, under the assumption that **E** requires exponential size circuits.

While reductions restricted to the uniform distribution are not the focus of this work, complementing the results of [22, 30], which provide *error-prone* search-to-decision reductions for K^t under the uniform distribution, we describe in Appendix D an *errorless* search-to-decision reduction for K^t under the uniform distribution.

As discussed in Section 1.1.1, [23] implicitly established an error-prone search-to-decision reduction for K^t under any polynomial-time samplable distribution, under the assumption $E \not\subseteq \text{i.o.NSIZE}[2^{o(n)}]$. Our error-prone search-to-decision reduction for K^t weakens this circuit complexity assumption, and provides a search algorithm for the more general case of *conditional* K^t complexity. We note that [23] also establishes a search-to-decision reduction for the probabilistic Kolmogorov complexity measure pK^t , which we do not consider in this work.

Additional Related Work. In two recent works, [29] and [15] obtain *non-uniform* algorithms solving the exact search problem for K^t . In more detail, in these results the size of the non-uniform circuit is of order $2^{4n/5}$ and the circuit neither needs access to, nor assumes the existence of an algorithm for the decision problem. It is not known how to extend these results to uniform algorithms.

In another recent paper, [28] describes a non-uniform polynomial-size search-to-decision reduction when the decision procedure solves **MINKT** with respect to any underlying universal Turing machine U , given black-box access to it (see their paper for details about this setting).

Search-to-decision reductions have also been investigated in the related setting of circuit complexity theory, where the goal is to compute the complexity of a given input function. [16] investigated this problem for Boolean formulas (corresponding to **MFSP**, the Minimum Formula Size Problem), and designed a worst-case search-to-decision reduction that runs in time $O(2^{0.67n})$ on an input function of description length n . Additionally, [16] obtained an improved running time of $2^{O(n/\log \log n)}$ when the search algorithm is only required to succeed with high probability over the uniform distribution.

Finally, we note that efficient randomized search-to-decision reductions are known for the complexity class DistNP . Every DistNP search problem can be reduced to some DistNP decision problem [3]. However, such reductions typically do not preserve the problem they reduce from (except for certain DistNP -complete problems like the Bounded Halting Problem), and so do not seem to apply to the case of search-to-decision reductions for MINKT and MINrKT studied in our work.

1.3 Techniques

From a technical perspective, our most interesting results are an unconditional errorless average-case search-to-decision reduction for rK^t (Theorem 3) and a conditional error-prone average-case search-to-decision reduction for K^t (Theorem 2). However, for illustration, we start with a more complete overview of the proof of the conditional errorless average-case search-to-decision reduction for K^t (Theorem 1), which is simpler yet captures some key ideas behind most of our proofs.

Average-Case Search-to-Decision for K^t . Our starting point is the aforementioned result from [24], which showed that if polynomial-time symmetry of information holds for K^t , then Search-MINKT can be solved over the *uniform* distribution. By inspecting the proof more carefully, we observe that if polynomial-time symmetry of information holds for K^t , then given t and x , one can find a shortest t -time program for x in time exponential in the $(t, p(t))$ -computational depth of x , i.e., $\text{cd}^{t, p(t)}(x) := \text{K}^t(x) - \text{K}^{p(t)}(x)$, for some polynomial p .

To show this, consider $x \in \{0, 1\}^n$ and any sufficiently large $t \in \mathbb{N}$. Let y_t be a shortest t -time program for generating x . By the assumed polynomial-time symmetry of information, we get that there is a polynomial p' such that the following holds:

$$\begin{aligned} \text{K}^{p'(2t)}(y_t \mid x) &\lesssim \text{K}^{2t}(x, y_t) - \text{K}^{p'(2t)}(x) && \text{(by polytime symmetry of information)} \\ &\lesssim |y_t| - \text{K}^{p'(2t)}(x) && \text{(since } x \text{ is determined by } y_t) \\ &= \text{K}^t(x) - \text{K}^{p'(2t)}(x) && \text{(since } |y_t| = \text{K}^t(x)) \\ &\leq \text{K}^t(x) - \text{K}^{p(t)}(x) && \text{(by monotonicity of } \text{K}^t \text{ with respect to } t) \\ &= \text{cd}^{t, p(t)}(x), && \text{(by definition of computational depth)} \end{aligned}$$

where $p > p'$ is a polynomial. The above essentially says that there is a program Π_{y_t} of size at most $\text{cd}^{t, p(t)}(x)$ such that $U(\Pi_{y_t}, x)$ outputs y_t within $p(t)$ steps.

Consider the following algorithm:

For an integer s , enumerate all programs $\Pi \in \{0, 1\}^{\leq s}$, and run $U(\Pi, x)$ for $p(t)$ steps to obtain a list of candidate K^t -witnesses y , which is guaranteed to include y_t if $\text{cd}^{t, p(t)}(x) \leq s$. For each such candidate y , check if y is indeed a t -time program for x , and output a valid one of the smallest length.

This algorithm runs in time $2^s \cdot \text{poly}(t)$, and finds a K^t -witness for every x with $\text{cd}^{t, p(t)}(x) \leq s$.

Using ideas from prior work on meta-complexity [7, 9, 5, 13], one can show that (assuming $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$), if MINKT is easy on average (in the errorless setting), then polynomial-time symmetry of information for K^t holds (see Lemma 22 below).

Since the $(t, p(t))$ -computational depth of x is small, i.e., $O(\log |x|)$, for a uniformly random x with high probability, the above yields a polynomial-time Search-MINKT algorithm over the uniform distribution. We want to extend to all *polynomial-time samplable* distributions.

To this end, we want to say that, for a polynomial-time samplable distribution \mathcal{D} , $\text{cd}^{t,p(t)}(x)$ is small for almost all x sampled from \mathcal{D} . It turns out that this is true if the *coding theorem* holds for K^t , i.e., if for every $x \in \{0,1\}^n$ in the support of \mathcal{D} , $\mathsf{K}^t(x) \leq -\log \mathcal{D}(x) + O(\log n)$, for any sufficiently large $t \geq \text{poly}(n)$. Combining the above with the well-known property of Kolmogorov complexity that for almost all x sampled from \mathcal{D} , $\mathsf{K}(x) \geq -\log \mathcal{D}(x) - O(\log n)$, we would get that $\text{cd}^{t,p(t)}(x) = \mathsf{K}^t(x) - \mathsf{K}^{p(t)}(x) \leq \mathsf{K}^t(x) - \mathsf{K}(x) \leq O(\log n)$ for almost all x sampled from \mathcal{D} .

Again, using ideas from meta-complexity in prior work, assuming $\mathsf{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ and average-case easiness of MINKT (in the errorless setting), we can obtain the requisite coding theorem for K^t (see Lemma 23).

Thus, by the coding theorem for K^t , we can already show that if MINKT is easy on average (and assuming $\mathsf{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$), one can efficiently solve Search-MINKT over polynomial-time samplable distributions. However, such an average-case algorithm can make errors for strings x whose $(t, p(t))$ -computational depth is *not* small. We would like to recognize such strings x , and output \perp on them. To this end, we will design a deterministic polynomial-time *computational depth certifying algorithm* A with the following two properties:

1. If $A(x)$ accepts, then indeed $\text{cd}^{t,p(t)}(x) \leq O(\log n)$, and
2. For almost all x sampled from \mathcal{D} , $A(x)$ accepts.

Given A , our final errorless average-case algorithm for solving Search-MINKT is as follows:

Given x and t , if the algorithm A accepts, which implies that $\text{cd}^{t,p(t)}(x)$ is small, then we are guaranteed that the previously-mentioned procedure can output a K^t witness of x . Otherwise if algorithm A rejects, which happens with only small probability over $x \sim \mathcal{D}$, we output \perp .

It remains to explain how to get the requisite algorithm A . By known results in meta-complexity, if MINKT is easy on average and if $\mathsf{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, then there is some polynomial q such that given x and t' , one can compute in deterministic polynomial time an integer s' such that $\mathsf{K}^{q(t')}(x) \lesssim s' \leq \mathsf{K}^{t'}(x)$. By running this algorithm on both $(x, 1^{q^{-1}(t)})$ and $(x, 1^{p(t)})$, we obtain an integer s such that

$$\mathsf{K}^t(x) - \mathsf{K}^{p(t)}(x) \leq s \lesssim \mathsf{K}^{q^{-1}(t)}(x) - \mathsf{K}^{q(p(t))}(x)$$

(see Lemma 24). Let A be the algorithm that computes a number s as above, accepting if $s \leq O(\log n)$, and rejecting otherwise. By definition, A satisfies property (1) above. Also, A satisfies property (2) above, since as discussed earlier, by the coding theorem, $\mathsf{K}^{q^{-1}(t)}(x) - \mathsf{K}^{q(p(t))}(x) \leq O(\log n)$ for almost all x sampled from \mathcal{D} , provided that t (hence $q^{-1}(t)$) is sufficiently large.

Worst-Case Search-to-Decision for K^t . As described above, assuming average-case tractability of MINKT, one can find a K^t -witness of x in time exponential to $\text{cd}^{t,p(t)}(x)$, where p is a polynomial. The observation is that for every x , there exists some good $t \leq 2^{n^\epsilon}$ such that $\text{cd}^{t,p(t)}(x)$ is at most $O(n/\log n)$. We show that using the above-described computational depth certifying algorithm, one can also find such a good t for a given x . Then for such a t , we can find a K^t -witness in time $2^{O(n/\log n)} \cdot \text{poly}(t)$.

Average-Case Search-to-Decision for rK^t . One can use ideas from prior work on meta-complexity, and a known generator with rK^t -style reconstruction, to obtain symmetry of information, coding theorem, and a worst-to-average reduction for rK^t , albeit with an

$O(\log^3 n)$ overhead (as opposed to $O(\log n)$ in the case for K^t), just assuming the average-case easiness of MINrKT (and no derandomization assumptions). By using these tools and following a similar approach as described above for the average-case search-to-decision reduction for K^t , we get an average-case search-to-decision reduction for rK^t with time roughly $2^{O(\log^3 n)} \cdot \text{poly}(t)$, which is quasi-polynomial; see Section B in the appendix for details.

A *polynomial-time* reduction, as stated in Theorem 3, is considerably more challenging to get, since we don't know the desired symmetry of information theorem and coding theorem for rK^t with an optimal $O(\log n)$ overhead. Our approach is to use the symmetry of information theorem (under an average-case easiness assumption for MINKT) and the coding theorem for pK^t with optimal $O(\log n)$ overheads. However, implementing this plan requires a delicate analysis. We consider two variants of computational depth defined as $rK^t(x) - pK^{\text{poly}(t)}(x)$ and $pK^t(x) - K(x)$, and argue that

1. rK^t -witnesses can be found in time exponential in the computational depth $rK^t(x) - pK^{\text{poly}(t)}(x)$ (Lemma 19),
2. the computational depth $rK^t(x) - pK^{\text{poly}(t)}(x)$ is upper-bounded by $O(pK^{t^{1/c}}(x) - K(x) + \log n)$, for some constant $c > 0$ (Theorem 18).

Finally, using the optimal coding theorems for K and pK^t , we conclude that the running time exponential in $O(pK^{t^{1/c}}(x) - K(x) + \log n)$ is actually average polynomial time for every given $t^{1/c}$ -time samplable distribution.

The proof of Theorem 18 requires a novel application of techniques from meta-complexity. The key idea is to combine the hitting-set generator $H_m: \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ of [8] and the disperser $G_m: \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$ of [31]. The generator H_m has an *efficient* albeit *sub-optimal* reconstruction: if there is a randomized polynomial-time algorithm D that *avoids* $H_m(x, -)$ (i.e., D outputs 0 on input $H_m(x, z)$ for every $z \in \{0, 1\}^d$, yet D outputs 1 on most inputs), then $rK^{\text{poly}(n)}(x) \leq O(m + \log n)$. The disperser G_m may be viewed as a hitting-set generator with an *inefficient* but *nearly optimal* reconstruction: if there is an algorithm D that avoids $G_m(x, -)$, then $K(x) \leq m + O(\log n)$. For $x \in \{0, 1\}^n$, we set $m \approx K(x)$ and $m' \approx pK^{t^{1/c}}(x) - K(x)$. We then argue that the concatenated generator $G_m(x, z) \circ H_{m'}(x, z')$ (for seeds z and z') has an efficient distinguisher, based on an algorithm that approximates the pK -complexity of its input. On the other hand, $G_m(x, z) \circ \mathcal{U}_{m'}$ is “indistinguishable” from the uniform distribution $\mathcal{U}_{m+m'}$ (by our choice of m). This implies that there is an efficient algorithm that takes m bits of advice and avoids $H_{m'}(x, -)$, which allows us to apply the reconstruction property of $H_{m'}$ to conclude the proof.

We should also point out another important difference between K^t and rK^t witness search. In the search-to-decision reduction for K^t , after generating a list of candidate K^t witnesses in the search algorithm, one can check whether each of them is a valid t -time program that outputs x . However, given a candidate randomized program y and λ , we cannot efficiently check whether y outputs x with probability at least λ or if this probability is less than λ , unless $\text{PP} = \text{BPP}$. However, we can distinguish the set of randomized programs that output x with probability at least λ and those that output x with probability less than $\lambda - (1/\ell)$, in time $\text{poly}(\ell)$. This allows us to find an $(1/\ell)$ - rK^t_λ -witness.

Error-Prone Average-Case Search-to-Decision for Conditional K^t . We first describe the proof ideas behind the (conditional) error-prone average-case search-to-decision reductions for K^t in [23] mentioned in Section 1.1.1.

First of all, it was shown in [22] that if MINKT is average-case easy (in the error-prone setting), then infinitely-often one-way functions do not exist. Also, implicit in [22, 23], if infinitely-often one-way functions do not exist, then there is an error-prone average-case

algorithm for solving Search-MINKT over the “universal t -time-bounded distribution” where each x is assigned the probability mass $2^{-K^t(x)}$. Thus, to get an average-case Search-MINKT algorithm over a *polynomial-time samplable distribution* \mathcal{D} , it suffices to argue that \mathcal{D} is *dominated*⁵ by the universal t -time-bounded distribution (for some polynomial t). The latter would follow from a coding theorem for K^{poly} .

While the coding theorem is not known to hold for K^{poly} , it does hold for $\text{p}K^{\text{poly}}$ [27]. Moreover, it is also known that K^{poly} and $\text{p}K^{\text{poly}}$ are essentially equivalent under the derandomization assumption that $E \not\subseteq \text{i.o. NSIZE}[2^{o(n)}]$ [6]. As a result, assuming $E \not\subseteq \text{i.o. NSIZE}[2^{o(n)}]$, one gets a coding theorem for K^{poly} . Using these observations, [23] showed that assuming $E \not\subseteq \text{i.o. NSIZE}[2^{o(n)}]$, the average-case algorithms for solving Search-MINKT over the class of universal poly-time-bounded distributions also work for the class of polynomial-time samplable distributions.

Our key observation is that assuming only $E \not\subseteq \text{i.o. SIZE}[2^{o(n)}]$, plus the non-existence of infinitely-often one-way functions, one can get an *average-case* coding theorem for K^{poly} ; this result is implicit in [17]. We then show that such an average-case coding theorem for K^{poly} implies that polynomial-time samplable distributions are dominated by universal poly-time-bounded distributions *on average*. In turn, this implies that the average-case Search-MINKT algorithms over universal poly-time-bounded distributions also work over polynomial-time samplable distributions.

Next, we explain how to generalize these ideas to get an average-case Search-MINcKT algorithm. For simplicity, consider a polynomial-time samplable distribution family $\{\mathcal{D}_n\}$ supported over $\{0, 1\}^n \times \{0, 1\}^n$. Also, let $\{\mathcal{C}_n\}$ be the family of marginal distributions of $\{\mathcal{D}_n\}$ on the second part. That is, to sample from \mathcal{C}_n , we sample (x, y) from \mathcal{D}_n and then output y . We observe the following equivalent way of sampling \mathcal{D}_n : First sample y from \mathcal{C}_n and then sample x from $\mathcal{D}_n(\cdot | y)$, where $\mathcal{D}_n(\cdot | y)$ is the conditional distribution of \mathcal{D}_n on the first part given that the second part is y .

First of all, by borrowing ideas from [22, 23], we show that non-existence of infinitely-often one-way functions implies that there is an (error-prone) average-case algorithm A such that, with high probability over $y \sim \mathcal{C}_n$, A outputs a $K^t(\cdot | y)$ -witness of x with high probability over the distribution \mathcal{E}_y^t assigning each x the probability mass $2^{-K^t(x|y)}$.

In [14], it was shown that if infinitely-often one-way functions do not exist, then one can get an average-case *conditional* coding theorem for $\text{p}K^{\text{poly}}$. By “derandomizing” the proof, we can show that assuming $E \not\subseteq \text{i.o. SIZE}[2^{o(n)}]$, plus the non-existence of infinitely-often one-way functions, one gets an average-case conditional coding theorem for K^{poly} which says that with high probability over $(x, y) \sim \mathcal{D}_n$,

$$K^{\text{poly}(n)}(x | y) \lesssim \frac{1}{\mathcal{D}_n(x | y)}. \quad (1)$$

Note that by an averaging argument, we get that with high probability over $y \sim \mathcal{C}_n$, Equation (1) holds with high probability over $x \sim \mathcal{D}_n(\cdot | y)$.

Now using this conditional coding theorem, we get that with high probability over $y \sim \mathcal{C}_n$, the distribution \mathcal{E}_y^t dominates $\mathcal{D}_n(\cdot | y)$, again, on average, for any sufficiently large $t \geq \text{poly}(n)$. By the same observation as discussed earlier, such “average-case domination” suffices for us to argue that the algorithm A , which works on average over \mathcal{E}_y^t , also works on average over the distribution $\mathcal{D}_n(\cdot | y)$. As a result, we get that with high probability over $y \sim \mathcal{C}_n$, A output a $K^t(\cdot | y)$ -witness of x with high probability over $x \sim \mathcal{D}_n(\cdot | y)$. This implies that A solves Search-MINcKT on average over $(x, y) \sim \mathcal{D}_n$.

⁵ Recall that a distribution \mathcal{D} dominates another distribution \mathcal{D}' if $\mathcal{D}(x) \geq \mathcal{D}'(x)/\text{poly}(n)$ for every x .

1.4 Concluding Remarks, Directions, and Open Problems

We have designed exact search-to-decision reductions for K^t and rK^t complexities in the average-case setting. The results for K^t hold under a widely believed hardness assumption, while the results for rK^t are unconditional. We have also made progress on worst-case to average-case search-to-decision reductions, where a worst-case search algorithm is obtained from an average-case easiness assumption on the decision problem. (As stated in Section 1.1.4, the assumptions on the decision problems in most results can be made considerably weaker, while maintaining the same conclusion.) A key contribution of our results is showing that search-to-decision reductions exist for any fixed polynomial-time samplable distribution. (We also describe new approximate reductions in Appendix C, and a new errorless reduction over the uniform distribution in Appendix D.) A summary of the existing average-case polynomial-time search-to-decision reductions for the measures K^t and rK^t appears in Table 1.

We would like to highlight the following problems and directions:

1. In the worst-case setting, it is currently possible that computing $K^t(x)$ admits a linear time algorithm, while finding a minimum t -time bounded program for x requires time $2^{\Omega(|x|)}$. Are there sub-exponential time (exact) worst-case to worst-case search-to-decision reductions for K^t and rK^t ?
2. Can we improve Theorems 4 and 5 so that the search algorithm works for every choice of the parameter t ? Note that this would provide a positive solution to the previous problem.
3. Design an unconditional polynomial-time error-prone search-to-decision reduction for rK^t for polynomial-time samplable distributions.
4. Our search-to-decision reductions are non-black-box, i.e., the search algorithm relies on the code of the decision algorithm. Is it possible to obtain black-box search-to-decision reductions for the settings considered in our work?
5. Is it possible to combine our techniques for exact search-to-decision with the techniques from [30] and Appendix C for approximate search-to-decision to obtain stronger results?

■ **Table 1** Summary of average-case polytime search-to-decision reductions for K^t and rK^t .

Assumption	Measure	Distribution	Errorless or Error-prone	Reference
None	K^t	Uniform	Error-prone	[22]
$E \not\subseteq \text{i.o. NSIZE}[2^{\sigma(n)}]$	K^t	P-Samplable	Error-prone	[23]
None	K^t	Uniform	Errorless	Appendix D
$E \not\subseteq \text{i.o. SIZE}[2^{\sigma(n)}]$	K^t	P-Samplable	Errorless	Theorem 1
$E \not\subseteq \text{i.o. SIZE}[2^{\sigma(n)}]$	K^t	P-Samplable	Error-prone	Theorem 2
None	rK^t	Uniform	Error-prone	[30] ⁶
None	rK^t	P-Samplable	Errorless	Theorem 3

⁶ The proof of [30, Theorem 1.3] via list recoverable codes extends to rK^t with simple modifications.

2 Preliminaries

2.1 Definitions and Notation

For a string $w \in \{0, 1\}^*$, we use $|w| \in \mathbb{N}$ to denote its length. The empty string is denoted by ϵ .

Time-Bounded Kolmogorov Complexity. Let U be a Turing machine. Given a positive integer t and a string $x \in \{0, 1\}^*$, we let

$$K_U^t(x) = \min_{p \in \{0, 1\}^*} \left\{ |p| \mid U(p, \epsilon) \text{ outputs } x \text{ in at most } t \text{ steps} \right\}.$$

We say that $K_U^t(x)$ is the *t-time-bounded Kolmogorov complexity of x* (with respect to U). As usual, we fix U to be a time-optimal machine [21], i.e., a universal machine that is almost as fast and length efficient as any other universal machine, and drop the index U when referring to time-bounded Kolmogorov complexity measures.

We also consider a randomized variant of K^t where instead of having a deterministic machine that prints x , we consider a randomized machine that generates x with high probability. Given a probability parameter $\lambda \in [0, 1]$ and a positive integer t , we let

$$rK_\lambda^t(x) = \min_{p \in \{0, 1\}^*} \left\{ |p| \mid \Pr_{r \sim \{0, 1\}^t} [U(p, r) \text{ outputs } x \text{ in at most } t \text{ steps}] \geq \lambda \right\}.$$

denote the *t-time-bounded randomized Kolmogorov complexity of x* . Note that we do not require that $U(p, r)$ stops in time at most t on every r .⁷ We assume that the random string r is given on a separate input tape.

Also, for $\lambda \in [0, 1]$ and a positive integer t , we let

$$pK_\lambda^t(x) = \min \left\{ k \mid \Pr_{r \sim \{0, 1\}^t} [\exists p \in \{0, 1\}^k, U(p, r) \text{ outputs } x \text{ in at most } t \text{ steps}] \geq \lambda \right\}.$$

denote the *t-time-bounded probabilistic Kolmogorov complexity of x* . For simplicity, in both definitions above, we omit λ when $\lambda = 2/3$.

For more information about different notions of randomized time-bounded Kolmogorov complexity and their applications, we refer to [26].

We use $K(x)$ to denote the (time-unbounded) Kolmogorov complexity of x .

These definitions are extended to *conditional* Kolmogorov complexity measures in the usual way. For instance, in $rK^t(x \mid y)$ the machine U is also given access to the string y as part of its input. We assume that the string y is given on a separate input tape.

Probability Distributions. We will consider distributions supported over pairs of strings. Let $\mathcal{D} = \{\mathcal{D}_{\langle n, m \rangle}\}_{n, m \in \mathbb{N}}$ be a family of polynomial-time samplable distributions⁸, where each $\mathcal{D}_{\langle n, m \rangle}$ is supported over $\{0, 1\}^n \times \{0, 1\}^m$. For $y \in \{0, 1\}^m$, we denote by $\mathcal{D}_{\langle n, m \rangle}(\cdot \mid y)$ the conditional distribution of $\mathcal{D}_{\langle n, m \rangle}$ on the first part given that the second part is y .

⁷ This condition would be computationally difficult to check for a given randomized program. However, in a setting where it might be relevant, it can be achieved with a clocked program by storing the value t using $\log t$ bits, or an approximation of t (e.g., the exponent of the smallest power of 2 not smaller than t) using just $\log \log t$ bits.

⁸ Recall that \mathcal{D} can be sampled in polynomial time if there is a polynomial-time algorithm Samp such that $\text{Samp}(1^{\langle n, m \rangle}, r)$ is distributed according to $\mathcal{D}_{\langle n, m \rangle}$ when r is a uniformly random string of length $\text{poly}(n, m)$.

We use $\mathcal{D}_{\langle n,m \rangle}(x, y)$ to denote the probability that the pair (x, y) is sampled from $\mathcal{D}_{\langle n,m \rangle}$. Similarly, $\mathcal{D}_{\langle n,m \rangle}(x \mid y)$ denotes the probability that x is sampled from the conditional distribution $\mathcal{D}_{\langle n,m \rangle}(\cdot \mid y)$.

2.2 Basic Results in Kolmogorov Complexity

We will need the following results.

► **Fact 6.** *For every $x \in \{0, 1\}^*$, time bound $t \in \mathbb{N}$, and $\lambda > 1/2$,*

$$K(x) \leq rK_{\lambda}^t(x).$$

Since we have not explicitly considered prefix-free encodings in our definitions, below we simply observe the following result, which is useful later.

► **Lemma 7** (“Kraft’s Inequality for K ”). *For all $n > 0$,*

$$\sum_{x \in \{0,1\}^n} 2^{-K(x)} \leq n^{O(1)}.$$

Proof. For every $x \in \{0, 1\}^n$, its Kolmogorov description of length $K(x)$ can be encoded using a *prefix-free* code (where no codeword is a prefix of another codeword) at the expense of extra $O(\log n)$ bits (roughly, by adding the encoding of the integer value $K(x) \leq n + O(1)$, using a simple prefix-free binary code where each bit of the message is repeated twice, and 10 is added at the end). Let $C(x)$ denote the length of this prefix-free encoding of x . Then we have

$$\begin{aligned} \sum_{x \in \{0,1\}^n} 2^{-K(x)} &\leq \sum_{x \in \{0,1\}^n} 2^{-C(x)+O(\log n)} \\ &\leq n^{O(1)} \cdot \sum_{x \in \{0,1\}^n} 2^{-C(x)} \\ &\leq n^{O(1)}, \end{aligned}$$

where the last step uses Kraft’s inequality (saying that for every prefix-free binary code with lengths $C(x)$, we have $\sum_x 2^{-C(x)} \leq 1$). ◀

► **Theorem 8** (Coding Theorem for pK^t [27]). *There is a constant $c > 0$, such that the following holds. For any distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, samplable in time $p(n)$, we have $pK^{p(n)^c}(x) \leq -\log \mathcal{D}_n(x) + O(\log p(n))$.*

► **Lemma 9** (See [14, Lemma 9]). *There exists a universal constant $b > 0$ such that for any distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, and $\gamma \in \mathbb{N}$,*

$$\Pr_{x \sim \mathcal{D}_n} \left[K(x) < \log \frac{1}{\mathcal{D}_n(x)} - \gamma \right] < \frac{n^b}{2^\gamma}.$$

► **Lemma 10** (Success Amplification for rK^t). *For any string $x \in \{0, 1\}^*$, time bound $t \in \mathbb{N}$, and $q \in \mathbb{N}$, we have*

$$rK_{1-1/q}^{t'}(x) \leq rK^t(x) + O(\log \log q),$$

where $t' := t \cdot O(\log q)$.

► **Proposition 11.** *The following hold.*

1. “MINKT \in HeurBPP” \implies (MINKT, \mathcal{U}) \in HeurBPP.
2. “MINKT \in AvgBPP” \implies (coMINKT, \mathcal{U}) \in Avg¹BPP.
3. “MINrKT \in AvgBPP” \implies (coMINKT, \mathcal{U}) \in Avg¹BPP.

Proof. The implication from “MINKT \in HeurBPP” to (MINKT, \mathcal{U}) \in HeurBPP (Item 1) is immediate. Next, we show that “MINrKT \in AvgBPP” implies (coMINKT, \mathcal{U}) \in Avg¹BPP (Item 2).

Suppose “MINrKT \in AvgBPP” holds. Then it follows that there exist a polynomial ρ and a probabilistic polynomial-time algorithm A' such that the following hold for all $n, s \in \mathbb{N}$, and all $t \geq \rho(n)$.

- For all $x \in \{0, 1\}^n$,

$$\Pr_{A'}[A' \text{ decides MINrKT on } (x, 2/3, 1^s, 1^t, 1^n) \text{ OR } A'(x, 2/3, 1^s, 1^t, 1^n) = \perp] \geq \frac{2}{3}. \quad (2)$$

- With probability at least $1 - 1/(2 \log t)$ over $x \sim \{0, 1\}^n$,

$$\Pr_{A'}[A' \text{ decides MINrKT on } (x, 2/3, 1^s, 1^t, 1^n)] \geq \frac{2}{3}. \quad (3)$$

Let A be the algorithm: On input $(x, 1^s, 1^t)$, A accepts if $A'(x, 2/3, 1^s, 1^t, 1^n)$ outputs 1 or \perp ; otherwise it rejects. We claim that the algorithm A satisfies the conditions stated for (coMINKT, \mathcal{U}) \in Avg¹BPP.

Let $t \geq \rho(n)$ and $s \leq n - 2 \log \log t$.

On the one hand, consider $x \in \{0, 1\}^n$ such that $K^t(x) \leq s$. Then we also have $rK^t(x) \leq s$. This means that $(x, 2/3, 1^s, 1^t, 1^n)$ is a YES instance of MINrKT. Then by Equation (2), $A'(x, 2/3, 1^s, 1^t, 1^n)$ outputs 1 or \perp with probability at least $2/3$, which implies that $A(x, 1^s, 1^t)$ accepts with probability at least $2/3$.

On the other hand, by a counting argument, we have that with probability at least $1 - 1/(2 \log t)$ over $x \sim \{0, 1\}^n$, $K(x) \geq n - \log(2 \log t) > s$. By Fact 6, we also get that

$$rK_{2/3-1/n}^t(x) > s.$$

In this case, $(x, 2/3, 1^s, 1^t, 1^n)$ is a NO instance of MINrKT. Combining this fact with Equation (3) and using a union bound, we get that with probability at least $1 - 1/(2 \log t) \geq 1/n$ over $x \sim \{0, 1\}^n$, $A'(x, 2/3, 1^s, 1^t, 1^n)$ rejects with probability at least $2/3$. Note that the above allows us to conclude that (coMINKT, \mathcal{U}) \in Avg¹BPP holds.

Item 3 can be shown in a similar way. We omit the details. ◀

We will also need the following lemma.

► **Lemma 12** (Computational Depth Upper Bound [11]). *For every $\varepsilon > 0$, every non-decreasing polynomials q_{dpt} and p_{dpt} , and every large enough $x \in \{0, 1\}^n$, there exists a time bound t^* such that $q_{\text{dpt}}(n) \leq t^* \leq 2^{n^\varepsilon}$ and*

$$K^{t^*}(x) - K^{p_{\text{dpt}}(t^*)}(x) \leq O\left(\frac{n}{\log n}\right).$$

Moreover, the same holds if we replace in the above $K^{t^*}(x) - K^{p_{\text{dpt}}(t^*)}(x)$ with $rK^{t^*}(x) - rK^{p_{\text{dpt}}(t^*)}(x)$.

Proof. We show the proof for randomized time-bound Kolmogorov complexity. The proof can be easily adapted to the deterministic case.

Given $x \in \{0, 1\}^n$ and polynomials q_{dpt} and p_{dpt} , define the polynomial $\tau := p_{\text{dpt}} \circ q_{\text{dpt}}$. For an integer $I \geq 1$, consider the following telescoping sum:

$$\begin{aligned} \text{rK}^{\tau(n)}(x) - \text{rK}^{\tau^{(I+1)}(n)}(x) &= \left(\text{rK}^{\tau(n)}(x) - \text{rK}^{\tau^{(2)}(n)}(x) \right) \\ &\quad + \left(\text{rK}^{\tau^{(2)}(n)}(x) - \text{rK}^{\tau^{(3)}(n)}(x) \right) + \cdots + \left(\text{rK}^{\tau^{(I)}(n)}(x) - \text{rK}^{\tau^{(I+1)}(n)}(x) \right), \end{aligned}$$

where $\tau^{(i)}$ denotes the composition of τ with itself i times. For any choice of x , q_{dpt} , and p_{dpt} as in the statement of the lemma, $\text{rK}^{\tau(n)}(x) \leq n + d$, for some universal constant $d \geq 0$; hence, the above sum is at most $n + d$. By averaging, there is some index $i_0 \in [I]$ such that

$$\text{rK}^{\tau^{(i_0)}(n)}(x) - \text{rK}^{\tau^{(i_0+1)}(n)}(x) \leq \frac{n + d}{I}. \quad (4)$$

For this i_0 , define $t^* := \tau^{(i_0)}(n)$. Note that $t^* \geq \tau(n) \geq (n)$, since $i_0 \geq 1$ and $p_{\text{dpt}}(\ell) \geq \ell$ for every input ℓ . Letting $c \in \mathbb{N}$ be such that $\tau(n) \leq n^c$ for sufficiently large n , define

$$I := \log_c \left(\frac{n^\varepsilon}{\log n} \right).$$

Then $t^* \leq n^{c^I} = 2^{n^\varepsilon}$. Moreover,

$$\begin{aligned} \text{rK}^{t^*}(x) - \text{rK}^{p_{\text{dpt}}(t^*)}(x) &\leq \text{rK}^{t^*}(x) - \text{rK}^{\tau^{(t^*)}}(x) \\ &\leq O \left(\frac{n}{\log n} \right), \end{aligned} \quad (\text{by Equation (4)})$$

where the constant behind the $O(-)$ can depend on ε and c (and hence q_{dpt} and p_{dpt}). ◀

3 Errorless Average-Case Search-to-Decision Reduction for rK^t

Here we prove Theorem 3, re-stated in its stronger form below (cf. Proposition 11).

► Theorem 13.

$$(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP} \implies \text{“SearchMINrKT} \in \text{AvgBPP”}.$$

3.1 Technical Tools

A *randomized oracle* $D: \{0, 1\}^m \rightarrow \{0, 1\}$ is a family $\{D_q\}_{q \in \{0, 1\}^m}$ of random variables D_q over $\{0, 1\}$. When a query $q \in \{0, 1\}^m$ is made to a randomized oracle D , a sample $a \sim D_q$ is returned independently.

We say that an algorithm $D: \{0, 1\}^m \rightarrow \{0, 1\}$ ε -*avoids* a generator $G: \{0, 1\}^d \rightarrow \{0, 1\}^m$ if D is 1 on at least ε fraction of its inputs, and yet $D(G(z)) = 0$ for all $z \in \{0, 1\}^d$. Similarly, a randomized oracle D ε -*avoids* a generator G if $\Pr_D[D(w) = 1] \geq \frac{2}{3}$ for at least $\varepsilon 2^m$ inputs $w \in \{0, 1\}^m$, and yet $\Pr_D[D(G(z)) = 0] \geq \frac{2}{3}$ for all $z \in \{0, 1\}^d$.

We will use the following two hitting-set generators with reconstruction properties.

► **Lemma 14** (Implicit in [8]). *There exists a polynomial-time-computable family*

$$H = \left\{ H_{n,m} : \{0,1\}^n \times \{0,1\}^{d(n,m)} \rightarrow \{0,1\}^m \right\}_{n,m \in \mathbb{N}}$$

of functions such that $d(n,m) = O(\log^3 m + \log n)$ and for any $x \in \{0,1\}^n$ and any randomized oracle $D: \{0,1\}^m \rightarrow \{0,1\}$ that ε -avoids $H_{n,m}(x, -)$, it holds that

$$\mathsf{K}^{t,D}(x) \leq 2m + O(\log^3 m + \log n)$$

for $t := \text{poly}(n, m)$.

Proof Sketch. For a deterministic oracle D , this is [8, Corollary 4.4]. By inspecting the proof, one can observe that the proof can be generalized to a randomized oracle D . ◀

We need the nearly optimal construction of a disperser obtained by [31]. We regard it as a hitting-set generator with an *inefficient* reconstruction property.

► **Lemma 15.** *There exists a polynomial-time-computable family*

$$G = \left\{ G_{n,m} : \{0,1\}^n \times \{0,1\}^{O(\log n)} \rightarrow \{0,1\}^m \right\}_{n,m \in \mathbb{N}}$$

of functions such that for any $x \in \{0,1\}^n$ and any oracle $D: \{0,1\}^m \rightarrow \{0,1\}$ that ε -avoids $G_{n,m}(x, -)$, it holds that

$$\mathsf{K}^D(x) \leq m + O(\log n).$$

Proof. We may assume without loss of generality that $m \leq 2n$ because otherwise the conclusion is obvious. It is shown in [31, Theorem 1.4] that for every n, k and constant $\varepsilon > 0$, there exists a strongly explicit bipartite graph (V, W, E) with left degree $2^d = n^{O(1)}$ such that $V = [2^n]$, $|W| = \Theta(2^{k+d-3\log n})$, and every subset $A \subseteq V$ of size at least 2^k has at least $(1 - \varepsilon/2)|W|$ distinct neighbours in W . We let $|W| = 2^m$, where $m = k + d - 3\log n \pm \Theta(1)$, and view the vertices in W as m -bit strings. We define $G_{n,m}(x, z)$ to be the z -th neighbour of $x \in \{0,1\}^n \equiv V$ for every $z \in [2^d] \equiv \{0,1\}^d$.

Let A be the set of n -bit strings $x \in \{0,1\}^n$ such that $D(G_{n,m}(x, z)) = 0$ for every $z \in \{0,1\}^d$. We claim that the size of A is at most 2^k . Assume, towards a contradiction, that $|A| \geq 2^k$. Let Γ denote the set of the neighbours of A . By the property of the disperser, $|\Gamma| \geq (1 - \varepsilon/2)|W|$. By the definition of A , for every $w \in \Gamma$, we have $D(w) = 0$. This contradicts the assumption that $D(w) = 1$ for at least an ε fraction of $w \in \{0,1\}^m$.

Observe that the elements of A can be enumerated given $n, m \in \mathbb{N}$ and oracle access to D . Thus, we obtain $\mathsf{K}^D(x) \leq \log |A| + O(\log nm) \leq k + O(\log n) \leq m + O(\log n)$ for every $x \in A$. ◀

► **Lemma 16** ([13, 5, 6]). *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$, then there exists a randomized polynomial-time algorithm M such that for every $x \in \{0,1\}^*$ and every $t \geq |x|$,*

$$\mathsf{pK}^{t^{O(1)}}(x) - O(\log n) \leq M(x, 1^t) \leq \mathsf{pK}^t(x)$$

with high probability over the internal randomness of M .

► **Lemma 17** (Symmetry of Information for pK^t ; implicit in [13, 6]). *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist polynomials p_{Sol} and p_0 such that for all sufficiently large $x, y \in \{0,1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$\mathsf{pK}^{p_{\text{Sol}}(t)}(y | x) \leq \mathsf{pK}^t(x, y) - \mathsf{pK}^{p_{\text{Sol}}(t)}(x) + \log p_{\text{Sol}}(|x| + |y|) + \log p_{\text{Sol}}(\log t).$$

The Symmetry of Information statement for pK^t as in Lemma 17 above was proved in [6] under the stronger assumption that distributional NP is easy on average for randomized polynomial-time algorithms in the errorless setting. It turns out that the weaker assumption on the average-case errorless easiness of MINKT (rather than all problems in NP) suffices to get the same result, with the proof similar to that in [6]. For completeness, we give the proof of Lemma 17 in Appendix A.

3.2 On Computational Depth

The following is the key result enabling us to argue that an algorithm that runs in time $2^{O(\mathsf{rK}^{\text{poly}(t)}(x) - \mathsf{K}(x) + \log n)}$ also runs in time $2^{O(\mathsf{pK}^{\text{poly}(t)}(x) - \mathsf{K}(x) + \log n)}$. The latter runtime can be shown to be average-polynomial-time over any t -time samplable distribution.

► **Theorem 18.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$, then for some polynomial p , for all $n \in \mathbb{N}$, all $t \geq n$, and all $x \in \{0, 1\}^n$, it holds that*

$$\mathsf{rK}^{p(t)}(x) - \mathsf{K}(x) \leq O(\mathsf{pK}^t(x) - \mathsf{K}(x) + \log n).$$

Moreover, for every polynomial q , there exists a randomized algorithm M such that, on input (x, t) , with probability at least $1 - o(1)$ over the internal randomness of M , outputs $v \in \mathbb{N}$ such that

$$\mathsf{rK}^{p(t)}(x) - \mathsf{pK}^{q(t)}(x) - O(\log n) \leq v \leq O(\mathsf{pK}^t(x) - \mathsf{K}(x) + \log n).$$

in time $2^{O(\mathsf{pK}^t(x) - \mathsf{K}(x) + \log n)}$.

Proof. Let G be the function of Lemma 15. Let H be the black-box hitting set generator construction of Lemma 14. The idea is to avoid $G_{n,m}(x, z) \circ H_{n,m'}(x, z')$ by measuring its Kolmogorov complexity for some m and m' . Let M be the algorithm of Lemma 16.

Define $m := \mathsf{K}(x) - c \log n$ and $m' = \mathsf{pK}^t(x) - \mathsf{K}(x) + \log^3 m' + c' \log n$ for sufficiently large constants c, c' . Observe that there exists a polynomial q such that

$$\begin{aligned} \mathsf{pK}^{q(t)}(G_{n,m}(x, z) \circ H_{n,m'}(x, z')) &\leq \mathsf{pK}^t(x) + |z'| + O(\log n) \\ &\leq m + m' - (c' - c - O(1)) \log n. \end{aligned}$$

Let D_0 be an algorithm that takes a string $w \in \{0, 1\}^{m+m'}$ and outputs 0 if and only if

$$M(w, 1^{q(t)}) \leq m + m' - (c' - c - O(1)) \log n.$$

Then, $D_0(G_{n,m}(x, z) \circ H_{n,m'}(x, z')) = 0$ because

$$\begin{aligned} M(G_{n,m}(x, z) \circ H_{n,m'}(x, z'), 1^{q(t)}) &\leq \mathsf{pK}^{q(t)}(G_{n,m}(x, z) \circ H_{n,m'}(x, z')) \\ &\leq m + m' - (c' - c - O(1)) \log n. \end{aligned}$$

On the other hand, for a uniformly random $w \in \{0, 1\}^{m+m'}$, we have $D_0(w) = 1$ with probability at least $1 - \varepsilon$ for a small $\varepsilon > 0$.

Let D' be an (inefficient) algorithm that takes $w \in \{0, 1\}^m$ and checks whether

$$\Pr_{w' \sim \{0,1\}^{m'}, D_0} [D_0(w \circ w') = 0] \leq 2\varepsilon.$$

29:20 Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

By Markov's inequality, with probability at least $\frac{1}{2}$ over $w \sim \{0, 1\}^m$, it holds that $D'(w) = 1$. If D' $\frac{1}{2}$ -avoids $G_{n,m}(x, -)$, by Lemma 15, we would obtain

$$\begin{aligned} \mathsf{K}(x) &\leq \mathsf{K}^{D'}(x) + O(1) \\ &\leq m + O(\log n) \\ &= \mathsf{K}(x) - (c - O(1)) \log n, \end{aligned}$$

which is a contradiction for a sufficiently large constant c . Thus, D' does not avoid $G_{n,m}(x, -)$ and so there exists $z \in \{0, 1\}^{O(\log n)}$ such that $D'(G_{n,m}(x, z)) = 1$. That is,

$$\Pr_{w' \sim \{0, 1\}^{m'}, D_0} [D_0(G_{n,m}(x, z) \circ w') = 0] \leq 2\varepsilon.$$

Next define a randomized oracle D as follows. On input $w' \in \{0, 1\}^{m'}$, $D(w') = 1$ if and only if $D_0(G_{n,m}(x, z) \circ w') = 1$. Note that D $(1 - 2\varepsilon)$ -avoids $H_{n,m'}(x, -)$, and so, by Lemma 14, we obtain

$$\begin{aligned} \mathsf{rK}^{p(t), D}(x) &\leq 2m' + O(\log^3 m' + \log n) \\ &\leq O(m' + \log n). \end{aligned}$$

Finally, observe that

$$\begin{aligned} \mathsf{rK}^{t^{O(1)}}(x) &\leq \mathsf{rK}^{p(t), D}(x) + m + O(\log m) \\ &\leq m + O(m' + \log n), \end{aligned}$$

because D can be computed by hard-wiring the fixed string $G_{n,m}(x, z) \in \{0, 1\}^m$. By the definitions of m and m' , we obtain that

$$\mathsf{rK}^{t^{O(1)}}(x) - \mathsf{K}(x) \leq O(\mathsf{pK}^t(x) - \mathsf{K}(x) + \log n).$$

This completes the proof of the first part.

To see the “moreover” part, we compute \tilde{m} such that

$$\mathsf{K}(x) - c \log n \leq \tilde{m} \leq \mathsf{pK}^{q(t)}(x) + O(\log n).$$

This can be done in randomized polynomial time by using the algorithm M . For every $m \leq \tilde{m}$, we define D_0 to be the algorithm that takes a string w of length $m + m'$ and outputs 1 if and only if $M(w, 1^{t^{O(1)}}) \leq m + m' - (c'/2) \log n$. We compute the maximum integer m such that there exists z such that $\Pr_{w'} [D_0(G_{n,m}(x, z) \circ w') = 0] \leq 2\varepsilon$. Note that m can be approximately computed in polynomial time by using random sampling. By the proof above, we have

$$\mathsf{K}(x) - c \log n \leq m \leq \tilde{m} \leq \mathsf{pK}^{q(t)}(x) + O(\log n).$$

Next, we compute the maximum integer m' such that $D_0(G_{n,m}(x, z) \circ H_{n,m'}(x, z')) = 1$ for all $z' \in \{0, 1\}^{O(\log^3 m' + \log n)}$. This can be computed in quasi-polynomial time in m' . By the proof above, we have $m' \leq \mathsf{pK}^t(x) - m + O(\log^3 m' + \log n)$. Finally, we define the output v to be m' . As in the proof above, we obtain

$$\mathsf{rK}^{t^{O(1)}}(x) \leq m + O(m' + \log n),$$

from which it follows that

$$\begin{aligned}
\mathbf{rK}^{t^{O(1)}}(x) - \mathbf{pK}^{q(t)}(x) - O(\log n) &\leq \mathbf{rK}^{t^{O(1)}}(x) - m \\
&\leq O(v + \log n) \\
&\leq O(\mathbf{pK}^t(x) - m + \log n) \\
&\leq O(\mathbf{pK}^t(x) - \mathbf{K}(x) + \log n),
\end{aligned}$$

as required. \blacktriangleleft

3.3 Finding \mathbf{rK}^t -Witnesses for Strings of Small Computational Depth

We call a $0\text{-}\mathbf{rK}_\lambda^t(x)$ -witness a $\mathbf{rK}_\lambda^t(x)$ -witness.

► **Lemma 19.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$, then for some polynomial p' , there exists a randomized polynomial-time algorithm A that, on input $(x, \lambda, 1^t, 1^k)$, outputs a list of strings that contains an \mathbf{rK}_λ^t -witness of x with probability at least $1 - o(1)$ over the internal randomness of A if*

$$\mathbf{rK}^{t/O(\log(1/(1-\lambda)))}(x) - \mathbf{pK}^{p'(t)}(x) + O(\log|x| + \log \log t + \log \log(1/(1-\lambda))) \leq \log k.$$

Proof. We assume without loss of generality that $\lambda \geq 2/3$. The proof can be easily adapted to the case where $\lambda \leq 2/3$.

The algorithm A operates as follows.

On input $(x, \lambda, 1^t, 1^k)$, repeat the following $k^{O(1)}$ times: Choose a uniformly random string r (of length t), run $U(z, r, x)$ for $\text{poly}(t)$ steps, for each string $z \in \{0, 1\}^{\leq \log k}$, and add its output to the list.

To prove the correctness, let y be the lexicographically first \mathbf{rK}_λ^t -witness of x . Note that $|y| = \mathbf{rK}_\lambda^t(x)$. By Lemma 17, we have

$$\mathbf{pK}^{\text{pSol}(2t)}(y | x) \leq \mathbf{pK}^{2t}(x, y) - \mathbf{pK}^{\text{pSol}(2t)}(x) + \log \text{pSol}(|x| + |y|) + \log \text{pSol}(\log t).$$

Observe that (x, y) can be described by y . Thus, we obtain

$$\begin{aligned}
\mathbf{pK}^{2t}(x, y) &\leq |y| + O(1) \\
&= \mathbf{rK}_\lambda^t(x) + O(1) && \text{(by the definition of } y\text{)} \\
&\leq \mathbf{rK}^{t/O(\log(1/(1-\lambda)))}(x) + O(\log \log(1/(1-\lambda))). && \text{(by Lemma 10)}
\end{aligned}$$

Combining these inequalities, we obtain

$$\begin{aligned}
\mathbf{pK}^{\text{pSol}(2t)}(y | x) &\leq \mathbf{rK}^{t/O(\log(1/(1-\lambda)))}(x) - \mathbf{pK}^{p'(t)}(x) \\
&\quad + O(\log|x| + \log \log t + \log \log(1/(1-\lambda))) \\
&\leq \log k,
\end{aligned}$$

which implies that A adds the witness y to its list with high probability. \blacktriangleleft

► **Lemma 20.** *Suppose $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$. Then for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_n$ supported over $\{0, 1\}^n$, there exist a polynomial ρ , a randomized algorithm A , and a time function T such that, for all $n \in \mathbb{N}$, $\lambda \in \mathbb{R}$, and*

$$t \geq \rho(n) \cdot \log(1/(1-\lambda)),$$

the following conditions hold:

29:22 Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

- For every $x \in \{0, 1\}^n$, with probability at least $2/3$ over its randomness, $A(x, \lambda, 1^t)$ stops within $T(x, \lambda, t)$ steps and outputs a list of strings that contains an rK_λ^t -witness of x .
- For some constant $\varepsilon > 0$,

$$\mathbf{E}_{x \sim \mathcal{D}_n} [T(x, \lambda, t)^\varepsilon] \leq \text{poly}(n, |\lambda|, t).$$

Proof. Throughout the proof, we will assume $t \geq \rho(n) \cdot \log(1/(1-\lambda))$, for some sufficiently large polynomial ρ to be specified later.

Let p the polynomial of Theorem 18. Also, let M be the algorithm from Theorem 18, instantiated with a sufficiently large polynomial q to be specified later. Let A and p' be the algorithm and polynomial of Lemma 19, respectively.

We define a new algorithm A' as follows.

On input $(x, \lambda, 1^t)$, let t_0 be the maximum integer t_0 such that

$$t \geq p(t_0) \cdot O(\log(1/(1-\lambda))).$$

Run M on input $(x, 1^{t_0})$ to obtain $v := M(x, 1^{t_0})$, and then simulate A on input $(x, \lambda, 1^t, 1^k)$ for

$$k := 2^{O(v + \log |x| + \log \log t + \log \log(1/(1-\lambda)))}$$

and output what A outputs.

By Theorem 18, we get that with probability at least $1 - o(1)$, the value v obtained in the algorithm satisfies

$$\text{rK}^{p(t_0)}(x) - \text{pK}^{q(t_0)}(x) - O(\log n) \leq v \leq O(\text{pK}^{t_0}(x) - \text{K}(x) + \log n). \quad (5)$$

Therefore, our algorithm will run in time

$$T(x, \lambda, t) := 2^{O(\text{pK}^{t_0}(x) - \text{K}(x) + \log n + \log t + \log |\lambda|)}.$$

Also, by letting q be a sufficiently large polynomial, we have

$$\text{rK}^{t/O(\log(1/(1-\lambda)))}(x) - \text{pK}^{p'(t)}(x) \leq \text{rK}^{p(t_0)}(x) - \text{pK}^{q(t_0)}(x) \leq O(v + \log n).$$

Thus, we have

$$\begin{aligned} & \text{rK}^{t/O(\log(1/(1-\lambda)))}(x) - \text{pK}^{p'(t)}(x) + O(\log |x| + \log \log t + \log(1/(1-\lambda))) \\ & \leq O(v + \log n) + O(\log |x| + \log \log t + \log \log(1/(1-\lambda))) \\ & \leq \log k, \end{aligned}$$

which means that the condition of Lemma 19 is satisfied.

As a result, we get that with probability at least $2/3$, the algorithm A' runs in time $T(x, \lambda, t)$ and outputs a list of strings that contains an rK_λ^t -witness of x .

We claim that for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}$, there exists a polynomial ρ such that for all large $n \in \mathbb{N}$, A' is an average-polynomial-time algorithm on input $(x, \lambda, 1^t)$ over $x \sim \mathcal{D}_n$ if $t \geq \rho(n) \cdot \log(1/(1-\lambda))$. Fix the parameters n, λ and t such that $t \geq \rho(n) \cdot \log(1/(1-\lambda))$. We have

$$\begin{aligned}
& \mathbf{E}_{x \sim \mathcal{D}_n} [T(x, \lambda, t)^\varepsilon] \\
& \leq \sum_x \mathcal{D}_n(x) \cdot 2^{\varepsilon \cdot O(\rho K^{t_0}(x) - K(x) + \log n + \log t + \log |\lambda|)} \\
& \leq n \cdot |\lambda| \cdot t \cdot \sum_x \mathcal{D}_n(x) \cdot 2^{\rho K^{t_0}(x) - K(x)} \quad (\text{for sufficiently small } \varepsilon > 0) \\
& \leq n \cdot |\lambda| \cdot t \cdot \sum_x 2^{-K(x)} \quad (\text{by Coding Theorem for } \rho K^t \text{ (Theorem 8)}) \\
& \leq n^{O(1)} \cdot |\lambda| \cdot t. \quad (\text{by "Kraft's Inequality for } K" \text{ (Lemma 7)})
\end{aligned}$$

Note that the penultimate inequality holds for ρ that is a sufficiently large polynomial. ◀

3.4 Proof of Theorem 3

By using Lemma 20, we obtain an errorless average-case polynomial-time algorithm for finding $(1/\ell)$ - rK^t -witnesses.

Proof of Theorem 13. Let $\{\mathcal{D}_n\}$ be a polynomial-time samplable distribution family.

Consider the algorithm A in Lemma 20. We first amplify the success probability of A , as follows. Given $(x, \lambda, 1^t, 1^k)$, we maintain $\text{poly}(k)$ executions of $A(x, \lambda, 1^t)$ *in parallel* (each with its own randomness). After half of the executions have stopped, we take the union of the outputs of these executions. By standard concentration bounds, we get an algorithm A' such that

$$\mathbf{E}_{x \sim \mathcal{D}_n} [T'(x, \lambda, t, k)^\varepsilon] \leq \text{poly}(n, |\lambda|, t, k),$$

where $\varepsilon > 0$ is a constant, and T' satisfies that for all x , with probability at least $1 - 2^{-k}/2$ over its randomness, $A'(x, \lambda, 1^t, 1^k)$ stops within $T'(x, \lambda, t, k)$ steps and outputs a list of strings that contains an rK_λ^t -witness of x .

By Markov's inequality, we get that for every k , with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, $A'(x, \lambda, 1^t, 1^k)$ runs in time $T_k := \text{poly}(n, |\lambda|, t, k)$ and outputs a list of strings that contains an rK_λ^t -witness of x , with probability at least $1 - 2^{-k}/2$ (over the internal randomness of A').

Consider the algorithm A'' that, on input $(x, \lambda, 1^t, 1^\ell, 1^k)$, simulates $A'(x, \lambda, 1^t, 1^{k+1})$. If it does not stop within T_k steps, we output \perp ; otherwise, we obtain a list of programs.

Note that for every x , we will either get \perp or obtain a list of programs that contains an rK_λ^t -witness of x , with probability at least $1 - 1/2^{-k}/2$ (over the internal randomness of A'').

Also, with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, we will obtain a list of programs that contains an rK_λ^t -witness of x , with probability at least $1 - 2^{-k}/2$ (over the internal randomness of A''). We aim to find an $(1/\ell)$ - rK_λ^t -witness of x in this case.

We need one more tool. Given $x \in \{0, 1\}^n$, a randomized program y and a time bound $t \in \mathbb{N}$, we will need to check whether y is a valid randomized program that outputs x with probability at least $\lambda - 1/\ell$.

▷ **Claim 21.** There is a polynomial-time algorithm `Valid` that takes as input $(x, y, \lambda, 1^t, 1^\ell, 1^{k'})$, where $x, y \in \{0, 1\}^*$, $\lambda \in (0, 1)$, and $t, \ell, k' \in \mathbb{N}$, and with probability at least $1 - 2^{-k'}$,

- accepts if y is a randomized program that outputs x within t steps with probability at least λ , and
- rejects if y is a randomized program that outputs x within t steps with probability less than $\lambda - 1/\ell$.

Proof Sketch of Claim 21. The algorithm repeatedly simulates the randomized program y for t steps, for $\text{poly}(\ell, k')$ simulations and counts the fraction of times that x is obtained. If this number is greater than $\lambda - 1/(2\ell)$, the algorithm accepts; otherwise it rejects. The correctness can be easily shown using Chernoff bounds. \triangleleft

Using the algorithm Valid in Claim 21, we can easily obtain, from a good list output by the algorithm A'' , an $(1/\ell)$ - rK_λ^t -witness of x , with probability at least $1 - 2^{-k}/2$, by outputting the first y in the list so that $\text{Valid}(x, y, \lambda, 1^\ell, 1^{k'})$ accepts, where k' is set appropriately.

It is easy to verify our final algorithm has polynomial running time. The correctness follows from a union bound. \blacktriangleleft

4 Errorless Average-Case Search-to-Decision Reduction for K^t

In this section we prove Theorem 1.

4.1 Technical Tools

The lemmas stated in this subsection are implicit in prior work, e.g., [7, 9, 5, 13]. The proof ideas are similar to those in Appendix B.1, but instead of using a generator with rK^t -style reconstruction, we use a generator with K^t reconstruction (assuming $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$). (See also Lemma 53.) We omit the details of the proofs since no new ideas are needed.

► **Lemma 22.** *Assume $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist polynomials p_{sol} and p_0 such that for all sufficiently large $x, y \in \{0, 1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$\text{K}^t(x, y) > \text{K}^{p_{\text{sol}}(t)}(x) + \text{K}^{p_{\text{sol}}(t)}(y | x) - \log p_{\text{sol}}(t).$$

► **Lemma 23.** *Assume $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_n$, there exists a polynomial p_{code} such that for every $n \in \mathbb{N}$ and $x \in \text{Support}(\mathcal{D}_n)$,*

$$\text{K}^{p_{\text{code}}(n)}(x) \leq \log \frac{1}{\mathcal{D}_n(x)} + \log p_{\text{code}}(n).$$

► **Lemma 24.** *Assume $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist a constant $c > 0$, a polynomial τ and an algorithm Approx-depth that, on input $(x, 1^{t_1}, 1^{t_2})$, where $x \in \{0, 1\}^n$, $t_1, t_2 \in \mathbb{N}$ with $t_1, t_2 \geq cn$, runs in time $\text{poly}(n, t_1, t_2)$ and outputs an integer s such that*

$$\text{K}^{\tau(t_1)}(x) - \text{K}^{t_2}(x) \leq s \leq \text{K}^{t_1}(x) - \text{K}^{\tau(t_2)}(x) + \log \tau(t_1) + \log \tau(t_2).$$

4.2 Proof of Theorem 1

The following implies Theorem 1 via Proposition 11.

► **Theorem 25.** *Assume $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a polynomial-time algorithm A such that the following holds for all $n, k \in \mathbb{N}$, and all $t \geq \rho(n)$.*

1. For all $x \in \{0, 1\}^n$, $A(x, 1^t, 1^k)$ outputs either a K^t -witness of x or \perp ,
2. and

$$\Pr_{x \sim \mathcal{D}_n} [A(x, 1^t, 1^k) = \perp] \leq \frac{1}{k}.$$

Proof. Throughout the proof, we will assume that $t \geq \rho(n)$ for some polynomial ρ , which will be specified later.

Let $t \in \mathbb{N}$ be such that $t \geq p_0(3n)$, where p_0 is the polynomial from Lemma 22. Consider any $x \in \{0, 1\}^n$, and let y_t be a K^t -witness of x . That is, y_t is the shortest t -time program that outputs x .

First of all, by symmetry of information (Lemma 22), there exists a polynomial p_{sol} ,

$$\begin{aligned} K^{p_{\text{sol}}(2t)}(y_t | x) &\leq K^{2t}(x, y_t) - K^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) \\ &\leq |y_t| - K^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + O(1) \\ &= K^t(x) - K^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + O(1) \end{aligned} \quad (6)$$

where the second inequality follows from the fact that given y_t , one can also output x within t steps.

Let $d > 0$ be some constant specified later, we say that $x \in \{0, 1\}^n$ is (t, k) -good if

$$K^t(x) - K^{p_{\text{sol}}(2t)}(x) \leq d \cdot \log t + \log k. \quad (7)$$

Consider any x, t, k such that x is (t, k) -good. Equation (6) implies that

$$\begin{aligned} K^{td}(y_t | x) &\leq K^{p_{\text{sol}}(2t)}(y_t | x) \\ &\leq K^t(x) - K^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + O(1) \\ &\leq 2d \log t + \log k, \end{aligned} \quad (8)$$

provided that d is a sufficiently large constant (which depends on p_{sol}).

Given Equation (8), we get that for some sufficiently large constant $c > d$, there is a program Π_{y_t} of length at most

$$s := c \cdot \log t + \log k \quad (9)$$

that, given x , outputs y_t within $T := t^c \cdot k^c$ steps. We aim to find such a y_t . Let A' be the following algorithm that, given $(x, 1^t)$ such that x is (t, k) -good, aims to output a K^t -witness of x .

■ **Algorithm 1** Search for K^t -Witnesses for Good x 's.

```

1: procedure  $A'(x, 1^t)$ 
2:    $n := |x|$ 
3:    $M := 0^{2n}$ 
4:    $s := c \cdot \log t + \log k$ , where  $c$  is the constant from Equation (9).
5:    $T := t^c \cdot k^c$ 
6:
7:   for  $\Pi \in \{0, 1\}^{\leq s}$  do
8:      $y :=$  the output of  $U(\Pi, x)$  after running  $T$  steps.
9:     if  $|y| < |M|$  and  $U(y)$  outputs  $x$  within  $t$  steps then
10:        $M := y$ 
11:   Output  $M$ 

```

It is easy to verify that $A'(x, 1^t)$ runs in time $\text{poly}(n, t, k)$. Next, we argue that if x is (t, k) -good, then the above algorithm outputs a K^t -witness of x .

Note that the algorithm A' always outputs some program M that can output x within t steps. Also, if x is (t, k) -good, then as described in previous paragraphs there is a program Π_{y_t} of length at most $s := c \cdot \log t + \log k$ such that $U(\Pi_{y_t}, x)$ outputs y_t within $T := t^c \cdot k^c$ steps. For such an x , we will have that $|M| \leq |y_t| = K^t(x)$ when $\Pi = \Pi_{y_t}$ in the for loop.

29:26 Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

We now describe our final algorithm A in the theorem. Let τ be the polynomial in Lemma 24, and let **Approx-depth** be the algorithm from Lemma 24. Our final algorithm A works as follows.

On input $(x, 1^t, 1^k)$, we first check if

$$\text{Approx-depth}\left(x, 1^{\lfloor \tau^{-1}(t) \rfloor}, 1^{p_{\text{Sol}}(2t)}\right) \leq d \cdot \log t + \log k,$$

where d is the constant in Equation (7). If yes, we output $A'(x, 1^t, 1^k)$. Otherwise, we output \perp .

We argue that the algorithm A above satisfies the two conditions stated in the theorem.

For the first condition, we consider two cases. Suppose x is not (t, k) -good, meaning that

$$\mathsf{K}^t(x) - \mathsf{K}^{p_{\text{Sol}}(2t)}(x) > d \cdot \log t + \log k.$$

Note that by Lemma 24, in this case **Approx-depth** $\left(x, 1^{\lfloor \tau^{-1}(t) \rfloor}, 1^{p_{\text{Sol}}(2t)}\right)$ outputs some s that satisfies

$$\begin{aligned} s &\geq \mathsf{K}^{\tau(\lfloor \tau^{-1}(t) \rfloor)}(x) - \mathsf{K}^{p_{\text{Sol}}(2t)}(x) \\ &\geq \mathsf{K}^t(x) - \mathsf{K}^{p_{\text{Sol}}(2t)}(x) \\ &> d \cdot \log t + \log k. \end{aligned}$$

Therefore, our algorithm will output \perp in this case. Now suppose x is (t, k) -good. As discussed above, for such x , $A'(x, 1^t, 1^k)$ will output a K^t -witness of x . Therefore, our algorithm will always output \perp or a K^t -witness of x .

For the second condition, we will show that in the above algorithm the criteria using **Approx-depth** will fail (hence output \perp) with probability at most $1/k$ over $x \sim \mathcal{D}_n$. To show this, we claim the following.

▷ **Claim 26.** For every $t, k \in \mathbb{N}$ such that $t \geq \rho(n)$, with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, we have

$$\text{Approx-depth}\left(x, 1^{\lfloor \tau^{-1}(t) \rfloor}, 1^{p_{\text{Sol}}(2t)}\right) \leq d \cdot \log t + \log k.$$

Proof of Claim 26. Recall the coding theorem for K^t (Lemma 23). By letting ρ be a sufficiently large polynomial so that for all $t \geq \rho(n)$, it is satisfied that $\lfloor \tau^{-1}(t) \rfloor \geq p_{\text{code}}(n)$, where p_{code} is the quasi-polynomial from Lemma 23, we get that for every $x \in \text{Support}(\mathcal{D}_n)$,

$$\mathsf{K}^{\lfloor \tau^{-1}(t) \rfloor}(x) \leq \mathsf{K}^{p_{\text{code}}(n)}(x) \leq \log \frac{1}{\mathcal{D}_n(x)} + \log p_{\text{code}}(n). \quad (10)$$

On the other hand, by Lemma 9, with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, we have

$$\mathsf{K}(x) \geq \log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k,$$

where $b > 0$ is a constant. In particular, this implies

$$\mathsf{K}^{\tau(p_{\text{Sol}}(2t))}(x) \geq \mathsf{K}(x) \geq \log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k. \quad (11)$$

Finally, we get that with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\begin{aligned}
& \text{Approx-depth}\left(x, 1^{\lfloor \tau^{-1}(t) \rfloor}, 1^{p_{\text{sol}}(2t)}\right) \\
& \leq \mathsf{K}^{\lfloor \tau^{-1}(t) \rfloor}(x) - \mathsf{K}^{\tau(p_{\text{sol}}(2t))}(x) + \log \tau(\lfloor \tau^{-1}(t) \rfloor) + \log \tau(p_{\text{sol}}(2t)) \quad (\text{by Lemma 24}) \\
& \leq \left(\log \frac{1}{\mathcal{D}_n(x)} + \log p_{\text{code}}(n) \right) - \left(\log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k \right) + \log t + \log \tau(p_{\text{sol}}(2t)) \\
& \quad (\text{by Equation (10) and Equation (11)}) \\
& = \log p_{\text{code}}(n) + b \log n + \log k + \log t + \log \tau(p_{\text{sol}}(2t)) \\
& \leq d \cdot \log t + \log k,
\end{aligned}$$

where the last inequality holds by letting d be a sufficiently large constant. \triangleleft

Claim 26 implies that for at least $1 - 1/k$ fraction of the x sampled from \mathcal{D}_n , our algorithm will output something other than \perp , as desired. \blacktriangleleft

5 Error-Prone Average-Case Search-to-Decision Reduction for Conditional K^t

In this section, we prove Theorem 2. We start with some technical tools.

5.1 Technical Tools

► **Lemma 27.** *Assume*

- $\mathsf{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, and
- *infinitely-often one-way functions do not exist.*

Then for every polynomial-time samplable distribution family $\{\mathcal{C}_{\langle n, m \rangle}\}$, where each $\mathcal{C}_{\langle n, m \rangle}$ is over $\{0, 1\}^m$, there exists a polynomial-time algorithm A such that for all $n, m, t, k \in \mathbb{N}$ with $t \geq n^{1.01}$, with probability at least $1 - 1/k$ over $y \sim \mathcal{C}_{\langle n, m \rangle}$,

$$\sum_{x \in \{0, 1\}^n} 2^{-\mathsf{K}^t(x|y)} \cdot \mathbb{1}_{[A(x, y, 1^t, 1^k) \notin \text{Search-MINKT}(x, y, 1^t)]} \leq \frac{\text{poly}(n)}{k}. \quad (12)$$

Proof. Let $c > 0$ be a constant so that $\mathsf{K}^t(x) \leq n + c$ for every $x \in \{0, 1\}^n$ and $t \geq n^{1.01}$. Let S be the sampler for $\{\mathcal{C}_{\langle n, m \rangle}\}$ that takes $u := \text{poly}(n, m)$ random bits.

Let f be a polynomial-time computable function defined as follows.

On input (ℓ, Π, r, r_1, r_2) , where $\ell \in \{0, 1\}^{\log(n+c)}$, $\Pi \in \{0, 1\}^{n+c}$, $r \in \{0, 1\}^u$, $r_1 \in \{0, 1\}^t$ and $r_2 \in \{0, 1\}^k$, we first obtain $y := \mathsf{S}(r)$. We then run $U(\Pi_{[\ell]}, y)$ for t steps and obtain a string x . If x is of length n , we output $(\ell, x, y, 1^t, 1^k)$; otherwise output $(\ell, 0^n, y, 1^t, 1^k)$.

Since we assume that $\mathsf{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ and that infinitely-often one-way functions do not exist (which implies infinitely-often weak one-way functions do not exist), there is a *deterministic* polynomial-time algorithm A' such that for all $n, m, t, k \in \mathbb{N}$, it holds that

$$\Pr[A'(\ell, x, y, 1^t, 1^k) \text{ succeeds}] \geq 1 - \frac{1}{k^2},$$

where $(\ell, x, y, 1^t, 1^k)$ is sampled according to f and “ $A'(\ell, x, y, 1^t, 1^k)$ succeeds” means $A'(\ell, x, y, 1^t, 1^k)$ outputs a pre-image of $(\ell, x, y, 1^t, 1^k)$. By an averaging argument, we

29:28 Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

get that with probability at least $1 - 1/k$ over $y \sim \mathcal{C}_{\langle n, m \rangle}$ (i.e., over $r \sim \{0, 1\}^u$), it holds that

$$\Pr[A'(\ell, x, y, 1^t, 1^k) \text{ succeeds}] \geq 1 - \frac{1}{k}, \quad (13)$$

where the above probability is only over ℓ and x . In what follows, fix a *good* y such that Equation (13) holds.

By a union bound, Equation (13) yields that for all $\ell \in \{0, 1\}^{\log(n+c)}$,

$$\Pr[A'(\ell, x, y, 1^t, 1^k) \text{ succeeds}] \geq 1 - \frac{n+c}{k}, \quad (14)$$

where now the probability is only over x .

Next, for any fixed ℓ , consider the following distribution $\mathcal{D}_{(y, \ell)}$:

1. Pick $\Pi \sim \{0, 1\}^{n+c}$.
2. Run $U(\Pi_{[\ell]}, y)$ for t steps and obtain a string x . If x is of length n , output x ; otherwise output 0^n .

Then Equation (14) implies that for all $\ell \in \{0, 1\}^{\log(n+c)}$,

$$\Pr_{(x) \sim \mathcal{D}_{(y, \ell)}} [A'(\ell, x, y, 1^t, 1^k) \text{ fails}] < \frac{n+c}{k}. \quad (15)$$

Now consider the following algorithm A :

On input $(x, y, 1^t, 1^k)$, we try $\ell = 1, 2, \dots, n+c$, and finds the smallest ℓ such that $A'(\ell, x, y, 1^t, 1^k)$ returns some (ℓ, Π, r, r_1, r_2) for which $y = S(r)$ and $U(\Pi_{[\ell]}, y)$ outputs x within t steps. Then we output $\Pi_{[\ell]}$.

We claim that the algorithm A satisfies the condition stated in Equation (12) for all good y . For the sake of contradiction, suppose there exists some good y such that

$$\sum_{x \in \{0, 1\}^n} 2^{-K^t(x|y)} \cdot \mathbf{1}_{[A(x, y, 1^t, 1^k) \notin \text{Search-MINcKT}(x, y, 1^t)]} > \frac{n^b}{k}, \quad (16)$$

where $b > 0$ is a constant specified later.

Note that for every fixed y and ℓ , the support of $\mathcal{D}_{(y, \ell)}$ consists of only strings whose $K^t(\cdot | y)$ -complexity is at most ℓ . Also, for every $x \in \{0, 1\}^n$ with $K^t(x | y) = \ell$, $\mathcal{D}_{(y, \ell)}$ outputs x with probability at least $2^{-K^t(x|y)}$. In other words, for every such x , we have

$$2^{-K^t(x|y)} \leq \mathcal{D}_{(y, \ell)}(x). \quad (17)$$

Also, for every $x \in \{0, 1\}^n$ with $K^t(x | y) = \ell$, if $A'(\ell, x, y, 1^t, 1^k)$ succeeds, then $A(x, y, 1^t, 1^k) \in \text{Search-MINcKT}(x, y, 1^t)$.

Then we have

$$\begin{aligned} \frac{n^b}{k} &\leq \sum_{\ell} \sum_{x: K^t(x|y)=\ell} 2^{-K^t(x|y)} \cdot \mathbf{1}_{[A(x, y, 1^t, 1^k) \notin \text{Search-MINcKT}(x, y, 1^t)]} && \text{(by Equation (16))} \\ &\leq \sum_{\ell} \sum_{x: K^t(x|y)=\ell} \mathcal{D}_{(y, \ell)}(x) \cdot \mathbf{1}_{[A(x, y, 1^t, 1^k) \notin \text{Search-MINcKT}(x, y, 1^t)]} && \text{(by Equation (17))} \\ &\leq \sum_{\ell} \sum_{x: K^t(x|y)=\ell} \mathcal{D}_{(y, \ell)}(x) \cdot \mathbf{1}_{[A'(\ell, x, y, 1^t, 1^k) \text{ fails}]}. \end{aligned}$$

By averaging, the above implies that there exists some ℓ such that

$$\sum_{x:K^\ell(x|y)=\ell} \mathcal{D}_{(y,\ell)}(x) \cdot \mathbb{1}_{[A'(\ell,x,y,1^\ell,1^k) \text{ fails}]} \geq \frac{n^b}{(n+c) \cdot k},$$

which contradicts Equation (15) by letting b be a sufficiently large constant. \blacktriangleleft

► **Lemma 28** (Implicit in [22]). *If $(\text{MINKT}, \mathcal{U}) \in \text{HeurBPP}$ holds, then infinitely-often one-way functions do not exist.*

► **Lemma 29** (Following [14]; see the proof of [14, Lemma 14]). *Assume*

- $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, and
- *infinitely-often one-way functions do not exist.*

Then for every polynomial-time samplable distribution family $\{\mathcal{D}_{(n,m)}\}$ supported over $\{0,1\}^n \times \{0,1\}^m$, there exists a polynomial p such that for all $n, m, k \in \mathbb{N}$,

$$\Pr_{(x,y) \sim \mathcal{D}_{(n,m)}} \left[\mathbb{K}^{p(n,m,k)}(x|y) \leq \log \frac{1}{\mathcal{D}_{(n,m)}(x|y)} + \log p(n,m,k) \right] \geq 1 - \frac{1}{k}.$$

Proof Sketch. First of all, [14, Lemma 14] gives that if infinitely-often one-way functions do not exist, then one can get average-case coding theorem for pK^{poly} . (See also [14, Section 1.3] for an exposition). The proof here is done by “derandomizing” that of [14, Lemma 14]. More specifically, it is not hard to adapt the proof of [14, Lemma 14] to show the following. If infinitely-often one-way functions do not exist, then for every polynomial-time samplable distribution family $\{\mathcal{D}_{(n,m)}\}$ supported over $\{0,1\}^n \times \{0,1\}^m$, there exists a *deterministic* polynomial-time algorithm Rec , such that for all $n, m, k \in \mathbb{N}$, with probability at least $1 - 1/k$ over $(x, y) \sim \mathcal{D}_{(n,m)}$,

$$\Pr_{\substack{w \sim \{0,1\}^{\text{poly}(n)} \\ r_{\text{Rec}} \sim \{0,1\}^{\text{poly}(n,m,k)}}} [\text{Rec}(H_w(x), y, w, 1^k; r_{\text{Rec}}) = x] \geq \frac{2}{3}, \quad (18)$$

Where H_w is a function from a pairwise independent hash family, mapping n bits to

$$s := \log \frac{1}{\mathcal{D}_{(n,m)}(x|y)} + O(\log n)$$

bits, and is indexed by the string w . Moreover, given w and x , $H_w(x)$ can be computed in time $\text{poly}(n)$.

Fix any (x, y) such that Equation (18) holds, we show that given y and an advice of length

$$\log \frac{1}{\mathcal{D}_{(n,m)}(x|y)} + O(\log nk),$$

we can output x in time $\text{poly}(n, m, k)$. This will conclude the proof of the lemma.

The idea is to derandomize Equation (18). Consider the circuit D that takes as input $w \in \{0,1\}^{\text{poly}(n)}$ and $r_{\text{Rec}} \in \{0,1\}^{\text{poly}(n,m,k)}$, and such that

$$D(w, r_{\text{Rec}}) = 1 \iff \text{Rec}(H_w(x), y, w, 1^k) = x.$$

Note that D can be implemented as a circuit of size $\text{poly}(n, m, k)$. Also, by Equation (18), we have

$$\Pr_{w, r_{\text{Rec}}} [D(w, r_{\text{Rec}}) = x] \geq \frac{2}{3}. \quad (19)$$

Assuming $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, there is a pseudorandom generator G of seed length $O(\log s)$ that can derandomize circuits of size at most s [20]. In particular,

$$\Pr_{z \sim \{0,1\}^{O(\log(nmk))}} [D(G(z)) = 1] - \Pr_{w, r_{\text{Rec}}} [D(w, r_{\text{Rec}}) = 1] \geq \frac{1}{10}.$$

Together with Equation (19), the above yields that there exists some $z \in \{0, 1\}^{O(\log(nmk))}$ such that for $(w, r_{\text{Rec}}) := G(z)$, we have

$$\text{Rec}(H_w(x), y, w, 1^k; r_{\text{Rec}}) = x.$$

Note that $|H_w(x)| = s$. As a result, given y, z and $H_w(x)$, we can recover x in time $\text{poly}(n, m, k)$, as desired. \blacktriangleleft

5.2 Proof of Theorem 2

We prove the following which implies Theorem 2.

► **Theorem 30.** *Assume $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{MINKT}, \mathcal{U}) \in \text{HeurBPP}$ holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_{\langle n, m \rangle}\}_{n, m \in \mathbb{N}}$ supported over $\{0, 1\}^n \times \{0, 1\}^m$, there exist a polynomial ρ and a polynomial-time algorithm A such that for all $n, m, k \in \mathbb{N}$, and all $t \geq \rho(n, m, k)$,*

$$\Pr_{(x, y) \sim \mathcal{D}_{\langle n, m \rangle}} [A(x, y, 1^t, 1^k) \text{ outputs a } \text{K}^t(\cdot | y)\text{-witness of } x] \geq 1 - \frac{1}{k}.$$

Proof. Let $\{\mathcal{D}_{\langle n, m \rangle}\}$ be a polynomial-time samplable distribution family. Let $\{\mathcal{C}_{\langle n, m \rangle}\}$ be the family of marginal distributions of $\{\mathcal{D}_{\langle n, m \rangle}\}$ on the second part. That is, to sample from $\mathcal{C}_{\langle n, m \rangle}$, we sample (x, y) from $\mathcal{D}_{\langle n, m \rangle}$ and then output y . Note that $\{\mathcal{C}_{\langle n, m \rangle}\}$ is polynomial-time samplable and is supported over $\{0, 1\}^m$. Also, let $n, m, k \in \mathbb{N}$, and all $t \geq \rho(n, m, k)$, where ρ is a polynomial specified later.

We show how to solve Search-MINcKT with probability at least $1 - 1/k$ over $\mathcal{D}_{\langle n, m \rangle}$.

First of all, since we assume that $(\text{MINKT}, \mathcal{U}) \in \text{HeurBPP}$ holds, by Lemma 28, we get that infinitely-often one-way functions do not exist. Let A' be the polynomial-time algorithm in Lemma 27. We have that with probability at least $1 - 1/(4k)$ over $y \sim \mathcal{C}_{\langle n, m \rangle}$,

$$\sum_{x \in \{0,1\}^n} 2^{-\text{K}^t(x|y)} \cdot \mathbb{1}_{[A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, y, 1^t)]} \leq \frac{1}{k^b \cdot (nm)^b}. \quad (20)$$

where $b > 0$ is a constant specified later.

Also, by Lemma 29 and an averaging argument, there exists a polynomial p such that, with probability at least $1 - 1/(4k)$ over $y \sim \mathcal{C}_{\langle n, m \rangle}$,

$$\Pr_{x \sim \mathcal{D}_{\langle n, m \rangle}(\cdot | y)} \left[\text{K}^{p(n, m, 16k^2)}(x | y) \leq \log \frac{1}{\mathcal{D}_{\langle n, m \rangle}(x | y)} + \log p(n, m, 16k^2) \right] \geq 1 - \frac{1}{4k}. \quad (21)$$

Fix any *good* y such that both Equation (20) and Equation (21) hold. Note that y is good with probability at least $1 - 1/(2k)$ when sampled from $\mathcal{C}_{\langle n, m \rangle}$. We claim that

$$\Pr_{x \sim \mathcal{D}_{\langle n, m \rangle}(\cdot | y)} \left[A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \text{ outputs a } \text{K}^t(\cdot | y)\text{-witness of } x \right] \geq 1 - \frac{1}{2k}. \quad (22)$$

Note that this suffices to show the theorem, since sampling $(x, y) \sim \mathcal{D}_{\langle n, m \rangle}$ is equivalent to first sampling $y \sim \mathcal{C}_{\langle n, m \rangle}$ and then sampling $x \sim \mathcal{D}_{\langle n, m \rangle}(\cdot | y)$.

Suppose, for the sake of contradiction, Equation (22) is not true. Then

$$\Pr_{x \sim \mathcal{D}_{\langle n, m \rangle}(\cdot | y)} \left[A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, y, 1^t) \right] > \frac{1}{2k}. \quad (23)$$

Let $\mathcal{E}(x)$ be the event that both the following hold.

- $A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, y, 1^t)$
- $\mathbb{K}^{p(n, m, 16k^2)}(x | y) \leq \log \frac{1}{\mathcal{D}_{\langle n, m \rangle}(x | y)} + \log p(n, m, 16k^2)$.

By Equation (23) and Equation (21), we get that

$$\sum_{x \in \{0, 1\}^n} \mathcal{D}_{\langle n, m \rangle}(x | y) \cdot \mathbf{1}_{\mathcal{E}(x)} \geq \frac{1}{4k}. \quad (24)$$

Note that whenever $\mathcal{E}(x)$ holds, we have

$$\mathcal{D}_{\langle n, m \rangle}(x | y) \leq \frac{p(n, m, 16k^2)}{2^{\mathbb{K}^{p(n, m, 16k^2)}(x | y)}}. \quad (25)$$

Now we have

$$\begin{aligned} \frac{1}{4k} &\leq \sum_{x \in \{0, 1\}^n} \mathcal{D}_{\langle n, m \rangle}(x | y) \cdot \mathbf{1}_{\mathcal{E}(x)} && \text{(by Equation (24))} \\ &\leq \sum_{x \in \{0, 1\}^n} \frac{p(n, m, 16k^2)}{2^{\mathbb{K}^{p(n, m, k)}(x | y)}} \cdot \mathbf{1}_{\mathcal{E}(x)} && \text{(by Equation (25))} \\ &\leq p(n, m, 16k^2) \cdot \sum_{x \in \{0, 1\}^n} 2^{-\mathbb{K}^{p(n, m, 16k^2)}(x | y)} \cdot \mathbf{1}_{\mathcal{E}(x)} \\ &\leq p(n, m, 16k^2) \cdot \sum_{x \in \{0, 1\}^n} 2^{-\mathbb{K}^{p(n, m, 16k^2)}(x | y)} \cdot \mathbf{1}_{[A'(x, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, 1^t)]} \\ &\leq p(n, m, 16k^2) \cdot \sum_{x \in \{0, 1\}^n} 2^{-\mathbb{K}^t(x | y)} \cdot \mathbf{1}_{[A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, y, 1^t)]} \end{aligned}$$

where the last inequality holds if $t \geq p(n, m, 16k^2)$. By rearranging, we get

$$\sum_{x \in \{0, 1\}^n} 2^{-\mathbb{K}^t(x | y)} \cdot \mathbf{1}_{[A'(x, y, 1^t, 1^{(nm)^b \cdot k^b}) \notin \text{Search-MINcKT}(x, y, 1^t)]} \geq \frac{1}{2k^2 \cdot p(n, m, 16k^2)}.$$

However, this contradicts Equation (20) by letting b be a sufficiently large constant. ◀

► **Remark 31.** In Theorem 30, our search algorithm only works for $t \geq \rho(n, m, k)$ instead of $t \geq \rho(n, m)$, where ρ is some polynomial (depending on the distribution family) and k is the parameter controlling the success probability of the algorithm. The reason for the dependency of k is that in the proof of Theorem 30, we need to apply the average-case conditional coding theorem (Lemma 29) with success probability at least $1 - 1/(4k)$ (see Equation (21)), and as a result, the time bound in the coding theorem is at least $\text{poly}(n, m, k)$. As shown at the end of the proof, we need t to be greater than this time bound.

6 Worst-Case to Average-Case Search-to-Decision Reductions

6.1 Worst-Case to Average-Case Search-to-Decision for $r\mathbb{K}^t$

In this subsection, we show the following which implies Theorem 5 via Proposition 11.

► **Theorem 32.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every $\varepsilon > 0$ and every polynomial β , there is a probabilistic algorithm A such that for all $n \in \mathbb{N}$, $x \in \{0, 1\}^n$, all $\ell \in \mathbb{N}$, and all $\lambda \in (0, 1)$ such that $\lambda \leq 1 - 1/2^{\text{poly}(n)}$, $A(x, \lambda, 1^\ell)$ runs in time $2^{O(n/\log n)} \cdot \text{poly}(|\lambda|, \ell)$ and, with probability at least $1 - 2^{-\ell}$, outputs a program M and an integer t that satisfy the following:*

- $\beta(n) \leq t \leq 2^{n^\varepsilon}$, and
- M is an $(1/\ell)$ - rK_λ^t -witness of x .

Proof. Without loss of generality, we assume $\lambda \geq 2/3$. The proof can be easily adapted to the case where $\lambda \leq 2/3$.

Let $0 < \varepsilon < 1$ and let β be a polynomial. Let $t \in \mathbb{N}$ be such that $t \geq p_0(3n) \cdot \log^2(1/(1-\lambda))$, where p_0 is the polynomial from Lemma 43. Consider any $x \in \{0, 1\}^n$, and let y_t be a rK_λ^t -witness of x . That is, y_t is a program such that $U(y_t, r)$ outputs x within t steps with probability at least λ over $r \sim \{0, 1\}^t$ and $|y_t| = \text{rK}_\lambda^t(x)$. Also, let $q := \lceil 1/(1-\lambda) \rceil$. Note that $\log(q) \leq O(|\lambda|)$.

By symmetry of information (Lemma 43), we have, for some polynomial p_{Sol} ,

$$\begin{aligned}
 & \text{rK}^{p_{\text{Sol}}(2t)}(y_t | x) \\
 & \leq \text{rK}^{2t}(x, y_t) - \text{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) \\
 & \leq |y_t| - \text{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) + O(1) \\
 & = \text{rK}_\lambda^t(x) - \text{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) + O(1) \quad (\text{by the definition of } y_t) \\
 & \leq \text{rK}_{1-1/q}^t(x) - \text{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) + O(1) \\
 & \leq \text{rK}^{t/O(\log q)}(x) - \text{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + O(\log \log q) \quad (\text{by Lemma 10}) \\
 & \leq \text{rK}^{\sqrt{t}}(x) - \text{rK}^{p_{\text{Sol}}(t^2)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) + O(\log \log q),
 \end{aligned}$$

where the second inequality follows from the fact that given y_t , one can also output x within t steps with probability at least $2/3$, and the last inequality uses that $t \geq p_0(3n) \cdot O(\log^2(1/(1-\lambda)))$. Then by the above, we have

$$\text{rK}^{p_{\text{Sol}}(2t)}(y_t | x) \leq \text{rK}^{\sqrt{t}}(x) - \text{rK}^{p_{\text{Sol}}(t^2)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(n) + O(\log |\lambda|). \quad (26)$$

We note that the above holds for all $t \geq p_0(3n) \cdot \log^2(1/(1-\lambda))$.

We claim the following.

▷ **Claim 33.** There is an algorithm B that, on input $x \in \{0, 1\}^n$ and $\ell \in \mathbb{N}$, runs in time $O(2^{n^\varepsilon}) \cdot \text{poly}(\ell)$ and with probability at least $1 - 2^{-\ell}$, outputs an integer t_{good} such that

- $\max\{p_0(3n) \cdot \log^2(1/(1-\lambda)), \beta(n)\} \leq t_{\text{good}} \leq 2^{n^\varepsilon}$, and
- $\text{rK}^{\sqrt{t_{\text{good}}}}(x) - \text{rK}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) \leq dn/\log n$, where $d \geq 1$ is a constant.

Proof of Claim 33. Let **Approx-depth** be the algorithm from Lemma 47, and let $\ell' := \ell + \lceil n^{\varepsilon/2} \rceil$. Also, let $d \geq 1$ be a constant specified later.

The algorithm B works as follows.

On input $x \in \{0, 1\}^n$, we enumerate all

$$t_0 \in \left[\max\{p_0(3n) \cdot \log^2(1/(1-\lambda)), \beta(n)\}, 2^{n^{\varepsilon/2}} \right]$$

and consider the first t_0 such that $\text{Approx-depth}(x, 1^{t_0}, 1^{\tau(p_{\text{Sol}}(t_0^4))}, 1^{\ell'}) \leq dn/\log n$. If such t_0 is found, we output $t_{\text{good}} := \tau(t_0^2)$, where τ is the polynomial from Lemma 47.

Otherwise, we output \perp .

It is easy to see that the running time of this algorithm is $O(2^{n^\varepsilon}) \cdot \text{poly}(\ell)$.

We now argue its correctness. First of all, by a union bound, we get that with probability except $2^{-\ell'} \cdot 2^{n^{\varepsilon/2}} \leq 2^{-\ell}$, $\text{Approx-depth}(x, 1^{t_0}, 1^{p_{\text{Sol}}(t_0^2)}, 1^{k'})$ will succeed (meaning that it outputs an answer that satisfies the condition stated in Lemma 47) on all $t_0 \leq 2^{n^{\varepsilon/2}}$. In what follows, we assume that this is the case.

Now consider Lemma 12 instantiated with the parameter $\varepsilon/2$, polynomials q_{dpt} such that $q_{\text{dpt}}(n) \geq \max\{p_0(3n) \cdot \log^2(1/(1-\lambda)), \beta(n)\}$, and p_{dpt} such that $p_{\text{dpt}}(z) \geq \tau^{(2)}(p_{\text{Sol}}(z^4))$. We have that there exists some t^* such that $q_{\text{dpt}}(n) \leq t^* \leq 2^{n^{\varepsilon/2}}$ and that

$$\text{rK}^{t^*}(x) - \text{rK}^{p_{\text{dpt}}(t^*)}(x) \leq \frac{d_0 \cdot n}{\log n}, \quad (27)$$

by choosing d_0 to be a large enough constant. For such t^* , $\text{Approx-depth}(x, 1^{t^*}, 1^{\tau(p_{\text{Sol}}((t^*)^4))}, 1^{k'})$ outputs some s that satisfies

$$\begin{aligned} s &\leq \text{rK}^{t^*}(x) - \text{rK}^{\tau(p_{\text{Sol}}((t^*)^4))}(x) + \log \tau(t^*) + \log \tau(p_{\text{Sol}}((t^*)^4)) + \log^3 \tau(n) \\ &\leq \text{rK}^{t^*}(x) - \text{rK}^{p_{\text{dpt}}(t^*)}(x) + \log \tau(t^*) + \log \tau(p_{\text{Sol}}((t^*)^4)) + \log^3 \tau(n) \\ &\leq \frac{2d_0 n}{\log n}. \end{aligned} \quad (\text{by Equation (27)})$$

In other words, if we let $d \geq 2d_0$, there is at least one t_0 (in particular, t^*) that can pass the test using Approx-depth . Also, by the property of Approx-depth , for any t_0 that passes the test, we have

$$\text{rK}^{\tau(t_0)}(x) - \text{rK}^{\tau(p_{\text{Sol}}((t_0)^4))}(x) \leq dn / \log n.$$

Recall that we will output $t_{\text{good}} := \tau(t_0^2)$. Then by the above, we have

$$\text{rK}^{\sqrt{t_{\text{good}}}}(x) - \text{rK}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) \leq dn / \log n,$$

as desired. \triangleleft

Suppose we run the above algorithm B on x and obtain an integer t_{good} that satisfies the condition stated in Claim 33. Now by Equation (26), where we let $t := t_{\text{good}}$, we get

$$\begin{aligned} &\text{rK}^{p_{\text{Sol}}(2t_{\text{good}})}(y_{t_{\text{good}}} | x) \\ &\leq \text{rK}^{\sqrt{t_{\text{good}}}}(x) - \text{rK}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) + \log p_{\text{Sol}}(2t_{\text{good}}) + \log^3 p_{\text{Sol}}(n) + O(1) \\ &\leq \frac{2dn}{\log n}, \end{aligned} \quad (28)$$

provided that d is a sufficiently large constant.

Given Equation (28) and using amplification techniques (Lemma 10), we get that for some large constant $c \geq 1$, there is a randomized program Π_{y_t} of length at most

$$s := cn / \log n + c \cdot \log \log \ell \quad (29)$$

that, given x , outputs y_t within $T := 2^{cn^\varepsilon} \cdot \ell^c$ steps with probability at least $1 - 2^{-\ell}/4$, where $t := t_{\text{good}}$ and y_t is a rK^t -witness of x . We aim to find such a y_t .

Let Valid be the algorithm from Claim 21. Consider the following algorithm A that, on input (x, λ, ℓ) , aims to output a program M and an integer t such that M is a $(1/\ell)$ - $\text{rK}_{1-\lambda}^t$ -witness of x .

■ **Algorithm 2** Search for rK^t -Witnesses.

```

1: procedure  $A(x, \lambda, 1^\ell)$ 
2:    $n := |x|$ 
3:    $M := 0^{2n}$ 
4:    $s := cn/\log n + c \log \log \ell$ , where  $c$  is the constant from Equation (29).
5:    $T := 2^{cn^\varepsilon} \cdot \ell^c$ 
6:
7:    $t := B(x, 1^{\ell+2})$ , where  $B$  is the algorithm in Claim 33.
8:
9:   for  $\Pi \in \{0, 1\}^{\leq s}$  do
10:     $r :=$  a uniformly random string in  $\{0, 1\}^T$ .
11:     $y :=$  the output of  $U(\Pi, x, r)$  after running  $T$  steps.
12:    if  $|y| < |M|$  and  $\text{Valid}(x, y, \lambda, 1^t, 1^\ell, 1^{\ell+s+3})$  then
13:       $M := y$ 
14:    Output  $M$  and  $t$ 

```

First of all, it is easy to verify that the above algorithm runs in time $2^{O(n/\log n)} \cdot \text{poly}(\ell)$. Next, we show its correctness.

Note that if the algorithm B succeeds (meaning that it returns an integer t such that there is a randomized program $\Pi_{y_t} \in \{0, 1\}^{\leq s}$ that outputs y_t within T steps with probability at least $1 - 2^{-\ell}/4$, where y_t is a rK^t -witness of x), which happens with probability at least $1 - 2^{-\ell}/4$, then our algorithm will succeed if both of the following are true.

1. The algorithm Valid succeeds in all of the $m := \sum_{i=1}^s 2^i \leq 2^{s+1}$ executions, which happens with probability at most $1 - 2^m \cdot 2^{-\ell-s-3} = 1 - 2^{-\ell}/4$.
2. For $\Pi = \Pi_{y_t}$, $U(\Pi, x, r)$ outputs y_t within T steps, which happens with probability at least $1 - 2^{-\ell}/4$ over $r \sim \{0, 1\}^T$.

To see this, if the first item is true, then the randomized program M output by the algorithm is a “valid” one that outputs x within t steps with probability at least $1 - 1/\ell$. If the second item is true, then $|M| \leq |y_t| = \text{rK}_\lambda^t(x)$, since $\text{Valid}(x, y_t, \lambda, 1^t, 1^\ell, 1^{\ell+s+3}) = 1$ (for a successful execution of Valid).

The correctness of the algorithm then follows by a union bound. ◀

6.2 Worst-Case to Average-Case Search-to-Decision for K^t

The following implies Theorem 4 via Proposition 11.

► **Theorem 34.** *Assume $\text{E} \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every $\varepsilon > 0$ and every polynomial β , there is an algorithm A such that for all $n \in \mathbb{N}$, $x \in \{0, 1\}^n$, $A(x)$ runs in time $2^{O(n/\log n)}$ and outputs a program M and an integer t that satisfy the following:*

- $\beta(n) \leq t \leq 2^{n^\varepsilon}$, and
- M is a K^t -witness of x .

Proof. The proof follows closely to that of Theorem 32.

Let $0 < \varepsilon < 1$ and let β be a polynomial.

Fix any $t \in \mathbb{N}$ such that $t \geq p_0(3n)$, where p_0 is the polynomial from Lemma 22. Consider any $x \in \{0, 1\}^n$, and let y_t be a K^t -witness of x . That is, y_t is a shortest program such that $U(y_t)$ outputs x within t steps.

By symmetry of information (Lemma 22), we have, for some polynomial p_{Sol} ,

$$\begin{aligned} \mathsf{K}^{p_{\text{Sol}}(2t)}(y_t \mid x) &\leq \mathsf{K}^{2t}(x, y_t) - r\mathsf{K}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) \\ &\leq |y_t| - \mathsf{K}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + O(1) \\ &= \mathsf{K}^t(x) - \mathsf{K}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + O(1) \quad (\text{by the definition of } y_t) \end{aligned}$$

where the second inequality follows from the fact that given y_t , one can also output x within t steps. Then by the above, we have

$$\mathsf{K}^{p_{\text{Sol}}(2t)}(y_t \mid x) \leq \mathsf{K}^t(x) - \mathsf{K}^{p_{\text{Sol}}(t^2)}(x) + \log p_{\text{Sol}}(2t) + O(1). \quad (30)$$

We show the following claim.

▷ **Claim 35.** There is an algorithm B that, on input $x \in \{0, 1\}^n$, runs in time $O(2^{n^\varepsilon})$ and outputs an integer t_{good} such that

- $\max\{p_0(3n), \beta(n)\} \leq t_{\text{good}} \leq 2^{n^\varepsilon}$, and
- $\mathsf{K}^{t_{\text{good}}}(x) - \mathsf{K}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) \leq dn/\log n$, where $d \geq 1$ is a constant.

Proof of Claim 35. Let **Approx-depth** be the algorithm from Lemma 24. Also, let $d \geq 1$ be a constant specified later.

The algorithm B works as follows.

On input $x \in \{0, 1\}^n$, we enumerate all

$$t_0 \in \left[\max\{p_0(3n), \beta(n)\}, 2^{n^{\varepsilon/2}} \right]$$

and consider the first t_0 such that $\mathbf{Approx\text{-}depth}(x, 1^{t_0}, 1^{\tau(p_{\text{Sol}}(t_0^2))}) \leq dn/\log n$. If such t_0 is found, we output $t_{\text{good}} := \tau(t_0)$, where τ is the polynomial from Lemma 47.

Otherwise, we output \perp .

It is easy to verify that the running time of this algorithm is $O(2^{n^\varepsilon})$. Next, we argue its correctness.

First of all, consider Lemma 12 instantiated with the parameter $\varepsilon/2$, polynomials q_{dpt} such that $q_{\text{dpt}}(n) \geq \max\{p_0(3n), \beta(n)\}$, and p_{dpt} such that $p_{\text{dpt}}(z) \geq \tau^{(2)}(p_{\text{Sol}}(z^2))$. We have that there exists some t^* such that $q_{\text{dpt}}(n) \leq t^* \leq 2^{n^{\varepsilon/2}}$ and that

$$\mathsf{K}^{t^*}(x) - \mathsf{K}^{p_{\text{dpt}}(t^*)}(x) \leq \frac{d_0 \cdot n}{\log n}, \quad (31)$$

by choosing d_0 to be a large enough constant. For such t^* , $\mathbf{Approx\text{-}depth}(x, 1^{t^*}, 1^{\tau(p_{\text{Sol}}((t^*)^2))})$ outputs some s that satisfies the following.

$$\begin{aligned} s &\leq \mathsf{K}^{t^*}(x) - \mathsf{K}^{\tau^{(2)}(p_{\text{Sol}}((t^*)^2))}(x) + \log \tau(t^*) + \log \tau(p_{\text{Sol}}((t^*)^2)) \\ &\leq \mathsf{K}^{t^*}(x) - \mathsf{K}^{p_{\text{dpt}}(t^*)}(x) + \log \tau(t^*) + \log \tau(p_{\text{Sol}}((t^*)^2)) \\ &\leq \frac{2d_0 n}{\log n}. \end{aligned} \quad (\text{by Equation (31)})$$

In other words, if we let $d \geq 2d_0$, there is at least one t_0 (in particular, t^*) that can pass the test using **Approx-depth**. Also, by the property of **Approx-depth**, for any t_0 that passes the test, we have

$$\mathsf{K}^{\tau(t_0)}(x) - \mathsf{K}^{\tau(p_{\text{Sol}}((t_0)^2))}(x) \leq dn/\log n.$$

29:36 Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity

Recall that we will output $t_{\text{good}} := \tau(t_0)$. Then by the above, we have

$$\mathsf{K}^{t_{\text{good}}}(x) - \mathsf{K}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) \leq dn / \log n,$$

as desired. \triangleleft

Suppose we run the above algorithm B on x and obtain an integer t_{good} that satisfies the condition stated in Claim 35. Now by Equation (30), where we let $t := t_{\text{good}}$, we get

$$\begin{aligned} \mathsf{K}^{p_{\text{Sol}}(2t_{\text{good}})}(y_{t_{\text{good}}} \mid x) &\leq \mathsf{K}^{t_{\text{good}}}(x) - \mathsf{K}^{p_{\text{Sol}}(t_{\text{good}}^2)}(x) + \log p_{\text{Sol}}(2t_{\text{good}}) + O(1) \\ &\leq \frac{2dn}{\log n}, \end{aligned} \quad (32)$$

provided that d is a sufficiently large constant.

Given Equation (32), we get that for some large constant $c \geq 1$, there is a program Π_{y_t} of length at most

$$s := cn / \log n \quad (33)$$

that, given x , outputs y_t within $T := 2^{cn^\varepsilon}$ steps, where $t := t_{\text{good}}$ and y_t is a K^t -witness of x . We aim to find such a y_t .

Consider the following algorithm A that, on input x , aims to output a program M and an integer t such that M is a K^t -witness of x .

■ **Algorithm 3** Search for K^t -Witnesses.

```

1: procedure  $A(x)$ 
2:    $n := |x|$ 
3:    $M := 0^{2n}$ 
4:    $s := cn / \log n$ , where  $c$  is the constant from Equation (33).
5:    $T := 2^{cn^\varepsilon}$ 
6:
7:    $t := B(x)$ , where  $B$  is the algorithm in Claim 35.
8:
9:   for  $\Pi \in \{0, 1\}^{\leq s}$  do
10:     $y :=$  the output of  $U(\Pi, x)$  after running  $T$  steps.
11:    if  $|y| < |M|$  and  $U(y)$  outputs  $x$  within  $t$  steps then
12:       $M := y$ 
13:   Output  $M$  and  $t$ 

```

First of all, it is easy to verify that the above algorithm runs in time $2^{O(n/\log n)}$. Next, we show its correctness.

Note that the algorithm B , on input x , will return a integer t that is “good” for x so that Equation (32) holds. For such t , there is some program Π_{y_t} of length at most $s := cn / \log n$ that outputs y_t , which is K^t -witness of x , within $T := 2^{cn^\varepsilon}$ steps. Since we enumerate all programs of size s , we will encounter Π_{y_t} and hence obtain y_t . This ensures that our algorithm can always find a t -time program for x , and the final program output by the algorithm has size at most $|y_t| = \mathsf{K}^t(x)$. \blacktriangleleft

References

- 1 Eric Allender, Joshua A. Grochow, Dieter van Melkebeek, Cristopher Moore, and Andrew Morgan. Minimum circuit size, graph isomorphism, and related problems. *SIAM J. Comput.*, 47(4):1339–1372, 2018. doi:10.1137/17M1157970.
- 2 Luis Filipe Coelho Antunes and Lance Fortnow. Worst-case running times for average-case algorithms. In *Conference on Computational Complexity (CCC)*, pages 298–303, 2009. doi:10.1109/CCC.2009.12.
- 3 Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. On the theory of average case complexity. *J. Comput. Syst. Sci.*, 44(2):193–219, 1992. doi:10.1016/0022-0000(92)90019-F.
- 4 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1), 2006. doi:10.1561/0400000004.
- 5 Halley Goldberg and Valentine Kabanets. A simpler proof of the worst-case to average-case reduction for polynomial hierarchy via symmetry of information. *Electron. Colloquium Comput. Complex.*, TR22-007:1–14, 2022. URL: <https://eccc.weizmann.ac.il/report/2022/007>, arXiv:TR22-007.
- 6 Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. Probabilistic Kolmogorov complexity with applications to average-case complexity. In *Computational Complexity Conference (CCC)*, pages 16:1–16:60, 2022. doi:10.4230/LIPIcs.CCC.2022.16.
- 7 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *Symposium on Foundations of Computer Science (FOCS)*, pages 247–258, 2018. doi:10.1109/FOCS.2018.00032.
- 8 Shuichi Hirahara. Characterizing average-case complexity of PH by worst-case meta-complexity. In *Symposium on Foundations of Computer Science (FOCS)*, pages 50–60, 2020. doi:10.1109/FOCS46700.2020.00014.
- 9 Shuichi Hirahara. Non-disjoint promise problems from meta-computational view of pseudorandom generator constructions. In *Conference on Computational Complexity (CCC)*, pages 20:1–20:47, 2020. doi:10.4230/LIPIcs.CCC.2020.20.
- 10 Shuichi Hirahara. Unexpected hardness results for Kolmogorov complexity under uniform reductions. In *Symposium on Theory of Computing (STOC)*, pages 1038–1051, 2020. doi:10.1145/3357713.3384251.
- 11 Shuichi Hirahara. Average-case hardness of NP from exponential worst-case hardness assumptions. In *Symposium on Theory of Computing (STOC)*, pages 292–302, 2021. doi:10.1145/3406325.3451065.
- 12 Shuichi Hirahara. Meta-computational average-case complexity: A new paradigm toward excluding heuristica. *Bull. EATCS*, 136, 2022. URL: <http://bulletin.eatcs.org/index.php/beatcs/article/view/688>.
- 13 Shuichi Hirahara. Symmetry of information from meta-complexity. In *Computational Complexity Conference (CCC)*, pages 26:1–26:41, 2022. doi:10.4230/LIPIcs.CCC.2022.26.
- 14 Shuichi Hirahara, Rahul Ilango, Zhenjian Lu, Mikito Nanashima, and Igor C. Oliveira. A duality between one-way functions and average-case symmetry of information. In *Symposium on Theory of Computing (STOC)*, pages 1039–1050, 2023. doi:10.1145/3564246.3585138.
- 15 Shuichi Hirahara, Rahul Ilango, and Ryan Williams. Beating brute force for compression problems. *Electron. Colloquium Comput. Complex.*, 171:1–30, 2023. URL: <https://eccc.weizmann.ac.il/report/2023/171/>, arXiv:TR23-171.
- 16 Rahul Ilango. Connecting Perebor conjectures: Towards a search to decision reduction for minimizing formulas. In *Computational Complexity Conference (CCC)*, pages 31:1–31:35, 2020. doi:10.4230/LIPIcs.CCC.2020.31.
- 17 Rahul Ilango, Hanlin Ren, and Rahul Santhanam. Hardness on any samplable distribution suffices: New characterizations of one-way functions by meta-complexity. *Electron. Colloquium Comput. Complex.*, page 82, 2021. URL: <https://eccc.weizmann.ac.il/report/2021/082>, arXiv:TR21-082.

- 18 Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference*, pages 134–147, 1995. doi:10.1109/SCT.1995.514853.
- 19 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Symposium on Theory of Computing (STOC)*, pages 812–821, 1990. doi:10.1109/FSCS.1990.89604.
- 20 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on Theory of Computing (STOC)*, pages 220–229. ACM, 1997. doi:10.1145/258533.258590.
- 21 Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. doi:10.1007/978-3-030-11298-1.
- 22 Yanyi Liu and Rafael Pass. On one-way functions and Kolmogorov complexity. In *Symposium on Foundations of Computer Science (FOCS)*, pages 1243–1254, 2020. doi:10.1109/FOCS46700.2020.00118.
- 23 Yanyi Liu and Rafael Pass. One-way functions and the hardness of (probabilistic) time-bounded Kolmogorov complexity w.r.t. samplable distributions. In *Annual Cryptology Conference (CRYPTO)*, pages 645–673, 2023. doi:10.1007/978-3-031-38545-2_21.
- 24 Luc Longpré and Osamu Watanabe. On symmetry of information and polynomial time invertibility. *Inf. Comput.*, 121(1):14–22, 1995. doi:10.1006/inco.1995.1120.
- 25 Zhenjian Lu and Igor C. Oliveira. An efficient coding theorem via probabilistic representations and its applications. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 94:1–94:20, 2021. doi:10.4230/LIPIcs.ICALP.2021.94.
- 26 Zhenjian Lu and Igor C. Oliveira. Theory and applications of probabilistic Kolmogorov complexity. *Bull. EATCS*, 137, 2022. URL: <http://bulletin.eatcs.org/index.php/beatcs/article/view/700>.
- 27 Zhenjian Lu, Igor C. Oliveira, and Marius Zimand. Optimal coding theorems in time-bounded Kolmogorov complexity. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 92:1–92:14, 2022. doi:10.4230/LIPIcs.ICALP.2022.92.
- 28 Noam Mazor and Rafael Pass. A note on the universality of black-box MK^tP solvers. *Electron. Colloquium Comput. Complex.*, 192:1–11, 2023. URL: <https://eccc.weizmann.ac.il/report/2023/192/>, arXiv:TR23-192.
- 29 Noam Mazor and Rafael Pass. The non-uniform peregbor conjecture for time-bounded Kolmogorov complexity is false. In *Innovations in Theoretical Computer Science (ITCS)*, pages 80:1–80:20, 2024. doi:10.4230/LIPIcs.ITCS.2024.80.
- 30 Noam Mazor and Rafael Pass. Search-to-decision reductions for Kolmogorov complexity. *Electron. Colloquium Comput. Complex.*, 3:TR24-003, 2024. URL: <https://eccc.weizmann.ac.il/report/2024/003>.
- 31 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless condensers, unbalanced expanders, and extractors. *Combinatorica*, 27(2):213–240, 2007. doi:10.1007/s00493-007-0053-2.
- 32 Boris A. Trakhtenbrot. A survey of Russian approaches to peregbor (brute-force searches) algorithms. *IEEE Ann. Hist. Comput.*, 6(4):384–400, 1984. doi:10.1109/MAHC.1984.10036.

A Symmetry of Information for pK^t

► **Lemma 36** (Symmetry of Information for pK^t). *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist polynomials p_{sol} and p_0 such that for all sufficiently large $x, y \in \{0, 1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$pK^{p_{\text{sol}}(t)}(y | x) \leq pK^t(x, y) - pK^{p_{\text{sol}}(t)}(x) + \log p_{\text{sol}}(|x| + |y|) + \log p_{\text{sol}}(\log t).$$

► **Definition 37** (Direct Product Generator [11, Definiton 3.10]). For $k, n \in \mathbb{N}$, we define the k -wise direct product generator to be the function

$$\text{DP}_k: \{0, 1\}^n \times \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk+k}$$

such that

$$\text{DP}_k(x; z^1, \dots, z^k) := (z^1, \dots, z^k, x \cdot z^1, \dots, x \cdot z^k).$$

► **Lemma 38** (pK^t Reconstruction Lemma [6, Lemma 22]). For $\varepsilon > 0$, $x \in \{0, 1\}^n$, $s \in \mathbb{N}$, and $k \in \mathbb{N}$ satisfying $k \leq 2n$, let D be a randomized algorithm that takes an advice string β and runs in time t_D such that D ε -distinguishes $\text{DP}_k(x; \mathcal{U}_{nk})$ from \mathcal{U}_{nk+k} . Then there is a polynomial p_{DP} such that

$$\text{pK}^{\tilde{O}(t_D) \cdot p_{\text{DP}}(n/\varepsilon)}(x \mid \beta) \leq k + \log p_{\text{DP}}(n/\varepsilon) + \log p_{\text{DP}}(\log t_D).$$

► **Remark 39.** One difference between Lemma 38 and [6, Lemma 22] is that the pK^t bound in Lemma 38 has an additive term $O(\log \log t_D)$ instead of $O(\log t_D)$ in [6, Lemma 22], where t_D is the running time of the distinguisher. The reason why we have an additive $O(\log t_D)$ term in [6, Lemma 22] is because in the reconstruction procedure we need to encode the number t_D , which takes $O(\log t_D)$ bits. However, we can assume without loss of generality that t_D is a power of two. Hence we can encode it using only $O(\log \log t_D)$ bits.

Proof of Lemma 36. Let τ be the smallest power of two that is at least t . Note that τ can be encoded using $O(\log \log \tau)$ bits.

Let $x \in \{0, 1\}^n$, $y \in \{0, 1\}^\ell$, and $k, k' \in \mathbb{N}$ to be defined later. Let $\text{DP}_{(-)}$ be the generator from Definition 37. Also, let $c \geq 1$ be a sufficiently large constant specified later.

To begin, observe that there exist a polynomial p_0 and a constant $d \geq 1$ such that for any $t \geq p_0(n, \ell)$, any choice of $z \in \{0, 1\}^{nk}$ and $z' \in \{0, 1\}^{\ell k'}$,

$$\text{pK}^{2\tau}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')) \leq \text{pK}^t(x, y) + |z| + |z'| + d \log(n\ell). \quad (34)$$

In particular, $p_0(n, \ell)$ reflects the time required to deterministically compute $\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')$ given xy, z, z' , and $d \log(n\ell)$ bits of information to delineate x from y . In what follows, we will give a lower bound on $\text{pK}^{2\tau}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z'))$ and thereby a lower bound on $\text{pK}^t(x, y)$.

Since we assume that $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, there exist a constant $c' > 0$, a polynomial ρ and a probabilistic polynomial-time algorithm B such that the following hold for all sufficiently large n' , all $t' \geq \rho(n')$, and $s' \leq n' - c' \cdot \log \log t'$, and .

1. If $r \in \{0, 1\}^{n'}$ and $\text{K}^{t'}(r) \leq s'$, then $\Pr_B[B(r, 1^{s'}, 1^{t'}) = 1] \geq 1 - \frac{1}{10n'}$.
2. With probability at least $1/n'$ over $r \sim \{0, 1\}^{n'}$, $\Pr_B[B(r, 1^{s'}, 1^{t'}) = 0] \geq 1 - \frac{1}{10n'}$.

Let $c > 0$ be a sufficiently large constant to be specified later, and consider the following parameters.

- $k := \text{pK}^{q(\tau)}(x) - \log q(\log \tau) - \log p_G(n\ell) - 1$ and $k' := \text{pK}^{q(\tau)}(y \mid x) - \log q(\log \tau) - \log q(n\ell) - 1$, where q is a sufficiently large polynomial specified later.
- $n' := nk + k + \ell k' + k' + \tau^c$.
- $s' := nk + k + \ell k' + k' + \tau^c - c \cdot \log \log \tau - c \cdot \log(n\ell)$.
- $t' := \tau^{2c}$.

We show the following which will imply a lower bound on $\text{rK}^{2t}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z'))$.

▷ Claim 40. There exist $z \in \{0, 1\}^{nk}$ and $z' \in \{0, 1\}^{\ell k'}$ such that

$$\Pr_{w \sim \{0, 1\}^{\tau^c}} \left[K^{t'}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z') \circ w) \leq s' \right] < 1 - \frac{1}{10n'}.$$

Proof of Claim 40. We first claim the following:

$$\Pr_{z, v, w, B} \left[B(\text{DP}_k(x; z) \circ vw, 1^{s'}, 1^{t'}) = 1 \right] \leq 1 - \frac{4}{10n'}. \quad (35)$$

Toward a contradiction, suppose

$$\Pr_{z, v, w, B} \left[B(\text{DP}_{k'}(x; z) \circ vw, 1^{s'}, 1^{t'}) = 1 \right] > 1 - \frac{4}{10n'}. \quad (36)$$

By the property of the algorithm B (Item 2), we have

$$\Pr_{u, v, w, B} \left[B(uvw, 1^{s'}, 1^{t'}) = 0 \right] \geq \frac{1}{2n'}.$$

In this case, comparing with Equation (36), we get a randomized distinguisher for $\text{DP}_k(x; \mathcal{U}_{nk})$ with advantage $1/10n'$, defined by sampling $v \sim \mathcal{U}_{\ell k' + k'}$, $w \sim \mathcal{U}_{\tau^c}$, and outputting $B(- \circ vw, 1^{s'}, 1^{t'})$. By Lemma 38, there exists some polynomial q such that

$$\mathbf{pK}^{q(\tau)}(x) \leq k + \log q(\log \tau) + \log p_G(n\ell). \quad (37)$$

Recall that $k = \mathbf{pK}^{q(t)}(x) - \log q(\log \tau) - \log q(n\ell) - 1$, so Equation (37) gives a contradiction. This shows Equation (35).

Now, toward a contradiction, suppose that for *all* z, z' ,

$$\Pr_w \left[K^{t'}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z') \circ w) \leq s' \right] \geq 1 - \frac{1}{10n'}.$$

By the property of B (Item 1), this implies that

$$\Pr_{z, z', w, B} \left[B(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z') \circ w, 1^{s'}, 1^{t'}) = 1 \right] \geq 1 - \frac{2}{10n'}.$$

In this case, comparing with Equation (35), we get a randomized distinguisher for $\text{DP}_{k'}(y; \mathcal{U}_{\ell k'})$ with advantage $(2/10n')$, defined by sampling $z \sim \mathcal{U}_{k'}$, $w \sim \mathcal{U}_{\tau^c}$, and outputting $B(\text{DP}_k(x; z) \circ - \circ w, 1^{s'}, 1^{t'})$. Again, by Lemma 38, we have

$$\mathbf{pK}^{q(\tau)}(y | x) \leq k' + \log q(\log \tau) + \log q(n\ell). \quad (38)$$

Recall that $k' = \mathbf{pK}^{q(\tau)}(y | x) - \log q(\tau) - \log q(n\ell) - 1$, so that Equation (38) gives a contradiction. This completes the proof of Claim 40. ◁

Next, we show that Claim 40 implies there exist z, z' such that

$$\mathbf{pK}_{1-1/10n'}^{\tau^c}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')) > s =: |z| + k + |z'| + k' - 2c \cdot \log \log \tau.$$

Suppose this is not the case. Then there exists a *deterministic* program M of length at most s such that $U(M, w)$ outputs $\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')$ within τ^c steps for at least $1 - 1/(10n')$ of the $w \in \{0, 1\}^{\tau^c}$, which implies

$$\Pr_{w \sim \{0, 1\}^{\tau^c}} \left[K^{\tau^{2c}}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z') \circ w) \leq s + \tau^c + O(\log(c \cdot \log \tau)) \right] \geq 1 - \frac{1}{10n'}. \quad (39)$$

On the other hand, we have

$$\begin{aligned} s + \tau^c + O(\log(c \log \tau)) &= (nk + k + \ell k' + k' - 2c \cdot \log \log \tau) + \tau^c + O(\log(c \cdot \log \tau)) \\ &\leq s', \end{aligned} \quad (40)$$

where the last inequality holds if we choose c to be a sufficiently large constant. Equation (39) and Equation (40) together imply that

$$\Pr_{w \sim \{0,1\}^{\tau^c}} \left[\mathbf{K}^{t'}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z') \circ w) \leq s' \right] \geq 1 - \frac{1}{10n'},$$

which contradicts Claim 40.

Therefore, there exist z, z' such that

$$\mathbf{pK}_{1-1/10n'}^{2c}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')) > |z| + k + |z'| + k' - 2c \cdot \log \log \tau.$$

By amplification techniques (Lemma 10), the above implies

$$\mathbf{pK}^{2\tau}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')) > |z| + k + |z'| + k' - 2c \cdot \log \log \tau - O(\log \log n'). \quad (41)$$

Finally, we get

$$\begin{aligned} \mathbf{pK}^t(x, y) &\geq \mathbf{pK}^{2\tau}(\text{DP}_k(x; z) \circ \text{DP}_{k'}(y; z')) - |z| - |z'| - d \log(n\ell) && \text{(by Equation (34))} \\ &> k + k' - 2c \cdot \log \log \tau - O(\log \log n') - d \log(n\ell) && \text{(by Equation (41))} \\ &= \left(\mathbf{pK}^{q(\tau)}(x) - \log q(\log \tau) - \log p_G(n\ell) - 1 \right) \\ &\quad + \left(\mathbf{pK}^{q(\tau)}(y | x) - \log q(\log \tau) - \log q(n\ell) - 1 \right) \\ &\quad - 2c \cdot \log \log \tau - O(\log \log n') - d \log(n\ell) \\ &\geq \mathbf{pK}^{p_{\text{sol}}(t)}(x) + \mathbf{pK}^{p_{\text{sol}}(t)}(y | x) - \log p_{\text{sol}}(|x| + |y|) - \log p_{\text{sol}}(\log t), \end{aligned}$$

where the last inequality holds by letting p_{sol} be a large enough polynomial. \blacktriangleleft

B Quasi-Polynomial-Time Average-Case Search-to-Decision Reduction for \mathbf{rK}^t

We introduce the following statement.

“**MINrKT** \in **AvgBPTIME** $[2^{O(\log^3 n)}]$ ”: For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a probabilistic algorithm A such that the following hold for all $\lambda \in (0, 1)$, all $n, s, \ell, k \in \mathbb{N}$, and all $t \geq \rho(n) \cdot \log(1/(1 - \lambda))$.

1. For all $x \in \{0, 1\}^n$, $A(x, \lambda, 1^t, 1^\ell, 1^k)$ runs in time $2^{O(\log^3 n)} \cdot \text{poly}(|\lambda|, t, \ell, k)$.
2. For all $x \in \{0, 1\}^n$,

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) \text{ outputs either an } (1/\ell)\text{-rK}_\lambda^t\text{-witness of } x \text{ or } \perp] \geq 1 - \frac{1}{2^k}.$$

3. With probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) \text{ outputs an } (1/\ell)\text{-rK}_\lambda^t\text{-witness of } x] \geq 1 - \frac{1}{2^k}.$$

In this section we prove the following quasipolynomial-time version of Theorem 3.

► **Theorem 41.** *We have*

$$\text{“MINrKT} \in \text{AvgBPP} \text{”} \implies \text{“MINrKT} \in \text{AvgBPTIME}[2^{O(\log^3 n)}] \text{”}.$$

B.1 Technical Tools

We begin with some technical tools.

B.1.1 A Generator with rK^t Reconstruction

We will use the following pseudorandom generator construction.

► **Lemma 42** (see e.g., [13] and [14, Lemma 26]). *There exists a polynomial p such that, for all sufficiently large $n, m, t \in \mathbb{N}$ such that $m \leq 2n$ and $t \geq n$, there exists a “pseudorandom generator construction”*

$$G_m: \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^m$$

such that for every $x \in \{0, 1\}^n$ and any function $D: \{0, 1\}^m \times \{0, 1\}^t \rightarrow \{0, 1\}$, if

$$\left| \Pr_{\substack{z \sim \{0, 1\}^d \\ w' \sim \{0, 1\}^t}} [D(G_m(x; z); w') = 1] - \Pr_{\substack{w \sim \{0, 1\}^m \\ w' \sim \{0, 1\}^t}} [D(w; w') = 1] \right| \geq \frac{1}{m},$$

then

$$\text{rK}^{p(t), D}(x) \leq m + O(\log^3 n).$$

Here, $d = O(\log^3 n)$ and G_m can be computed in time $\text{poly}(n)$.

B.1.2 Symmetry of Information for rK^t

► **Lemma 43** (Symmetry of Information for rK^t). *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist polynomials p_{sol} and p_0 such that for all sufficiently large $x, y \in \{0, 1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$\text{rK}^t(x, y) > \text{rK}^{p_{\text{sol}}(t)}(x) + \text{rK}^{p_{\text{sol}}(t)}(y | x) - \log p_{\text{sol}}(t) - \log^3 p_{\text{sol}}(|x| \cdot |y|).$$

Proof. The proof follows closely to that of Lemma 36.

Let $x \in \{0, 1\}^n$, $y \in \{0, 1\}^\ell$, and $m, m' \in \mathbb{N}$ to be defined later. Let $G_{(-)}$ be the generator from Lemma 42. Also, let $c \geq 1$ be a sufficiently large constant specified later.

To begin, observe that there exist a polynomial p_0 and a constant $d \geq 1$ such that for any $t \geq p_0(n, \ell)$, any choice of $z \in \{0, 1\}^{O(\log^3 n)}$ and $z' \in \{0, 1\}^{O(\log^3 \ell)}$,

$$\text{rK}^{2t}(G_m(x; z) \circ G_{m'}(y; z')) \leq \text{rK}^t(x, y) + |z| + |z'| + d \log t. \quad (42)$$

In particular, $p_0(n, \ell)$ reflects the time required to deterministically compute $G_m(x; z) \circ G_{m'}(y; z')$ given xy, z, z' , and $d \log t$ bits of information to delineate x from y . In what follows, we will give a lower bound on $\text{rK}^{2t}(G_m(x; z) \circ G_{m'}(y; z'))$ and thereby a lower bound on $\text{rK}^t(x, y)$.

Since we assume that $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, there exist a constant $c' > 0$, a polynomial ρ and a probabilistic polynomial-time algorithm B such that the following hold for all sufficiently large n' , all $t' \geq \rho(n')$, and all $s' \leq n' - c' \cdot \log \log t'$.

1. If $r \in \{0, 1\}^{n'}$ and $\text{K}^{t'}(r) \leq s'$, then $\Pr_B[B(r, 1^{s'}, 1^{t'}) = 1] \geq 1 - \frac{1}{10n'}$.
2. With probability at least $1/n'$ over $r \sim \{0, 1\}^{n'}$, $\Pr_B[B(r, 1^{s'}, 1^{t'}) = 0] \geq 1 - \frac{1}{10n'}$.

Let $c > 0$ be a sufficiently large constant to be specified later, and consider the following parameters.

- $m := \text{rk}^{p_G(t)}(x) - \log p_G(t) - \log^3 p_G(n\ell) - 1$ and $m' := \text{rk}^{p_G(t)}(y \mid x) - \log p_G(t) - \log^3 p_G(n\ell) - 1$, where p_G is a sufficiently large polynomial specified later.
- $n' := m + m' + t^c$.
- $s' := m + m' + t^c - c^2 \cdot \log(t) - c \cdot \log^3(n\ell)$.
- $t' := t^{2c}$.

We show the following which will imply a lower bound on $\text{rk}^{2t}(G_m(x; z) \circ G_{m'}(y; z'))$.

▷ **Claim 44.** There exist $z \in O(\log^3 n)$ and $z' \in O(\log^3 \ell)$ such that

$$\Pr_{w \sim \{0,1\}^{t^c}} \left[\mathbf{K}^{t'}(G_m(x; z) \circ G_{m'}(y; z') \circ w) \leq s' \right] < 1 - \frac{1}{10n'}.$$

Proof of Claim 44. We first claim the following:

$$\Pr_{z, v, w, B} \left[B(G_m(x; z) \circ vw, 1^{s'}, 1^{t'}) = 1 \right] \leq 1 - \frac{4}{10n'}. \quad (43)$$

Toward a contradiction, suppose

$$\Pr_{z, v, w, B} \left[B(G_m(x; z) \circ vw, 1^{s'}, 1^{t'}) = 1 \right] > 1 - \frac{4}{10n'}. \quad (44)$$

By the property of the algorithm B (Item 2), we have

$$\Pr_{u, v, w, B} \left[B(uvw, 1^{s'}, 1^{t'}) = 0 \right] \geq \frac{1}{2n'}.$$

In this case, comparing with Equation (44), we get a randomized distinguisher for $G_m(x; \mathcal{U}_{O(\log^3 n)})$ with advantage $1/10n'$, defined by sampling $v \sim \mathcal{U}_{m'}$, $w \sim \mathcal{U}_{t^c}$, and outputting $B(- \circ vw, 1^{s'}, 1^{t'})$. By Lemma 42, there exists some polynomial p_G such that

$$\text{rk}^{p_G(t)}(x) \leq m + \log p_G(t) + \log^3 p_G(n\ell). \quad (45)$$

Recall that $m = \text{rk}^{p_G(t)}(x) - \log p_G(t) - \log^3 p_G(n\ell) - 1$, so Equation (45) gives a contradiction. This shows Equation (43).

Now, toward a contradiction, suppose that for *all* z, z' ,

$$\Pr_w \left[\mathbf{K}^{t'}(G_m(x; z) \circ G_{m'}(y; z') \circ w) \leq s' \right] \geq 1 - \frac{1}{10n'}.$$

By the property of B (Item 1), this implies that

$$\Pr_{z, z', w, B} \left[B(G_m(x; z) \circ G_{m'}(y; z')) \circ w, 1^{s'}, 1^{t'} = 1 \right] \geq 1 - \frac{2}{10n'}.$$

In this case, comparing with Equation (43), we get a randomized distinguisher for $G_{m'}(y; \mathcal{U}_{O(\log^3 \ell)})$ with advantage $(2/10n')$, defined by sampling $z \sim \mathcal{U}_{O(\log^3 \ell)}$, $w \sim \mathcal{U}_{t^c}$, and outputting $B(G_m(x; z) \circ - \circ w, 1^{s'}, 1^{t'})$. Again, by Lemma 42, we have

$$\text{rk}^{p_G(t)}(y \mid x) \leq m' + \log p_G(t) + \log^3 p_G(n\ell). \quad (46)$$

Recall that $m' = \text{rk}^{p_G(t)}(y \mid x) - \log p_G(t) - \log^3 p_G(n\ell) - 1$, so that Equation (46) gives a contradiction. This completes the proof of Claim 44. \triangleleft

Next, we show that Claim 44 implies there exist z, z' such that

$$\text{rK}_{1-1/10n'}^{t^c}(G_m(x; z) \circ G_{m'}(y; z')) > s =: |z| + m + |z'| + m' - c^3 \log(t).$$

Suppose this is not the case. Then there exists a *deterministic* program M of length at most s such that $U(M, w)$ outputs $G_m(x; z) \circ G_{m'}(y; z')$ within t^c steps for at least $1 - 1/(10n')$ of the $w \in \{0, 1\}^{t^c}$, which implies

$$\Pr_{w \sim \{0, 1\}^{t^c}} \left[\text{K}^{t^c}(G_m(x; z) \circ G_{m'}(y; z') \circ w) \leq s + t^c + O(c \log t) \right] \geq 1 - \frac{1}{10n'}. \quad (47)$$

On the other hand, we have

$$\begin{aligned} s + t^c + O(c \log t) &= (m + m' + O(\log^3 n) + O(\log^3 \ell) - c^3 \log(t)) + t^c + O(c \log t) \\ &\leq s', \end{aligned} \quad (48)$$

where the last inequality holds if we choose c to be a sufficiently large constant. Equation (47) and Equation (48) together imply that

$$\Pr_{w \sim \{0, 1\}^{t^c}} \left[\text{K}^{t'}(G_m(x; z) \circ G_{m'}(y; z') \circ w) \leq s' \right] \geq 1 - \frac{1}{10n'},$$

which contradicts Claim 44.

Therefore, there exist z, z' such that

$$\text{rK}_{1-1/10n'}^{t^c}(G_m(x; z) \circ G_{m'}(y; z')) > |z| + m + |z'| + m' - c^2 \log(t).$$

By amplification techniques (Lemma 10), the above implies

$$\text{rK}^{2t}(G_m(x; z) \circ G_{m'}(y; z')) > |z| + m + |z'| + m' - c^2 \log(t) - O(\log \log n'). \quad (49)$$

Finally, we get

$$\begin{aligned} \text{rK}^t(x, y) &\geq \text{rK}^{2t}(G_m(x; z) \circ G_{m'}(y; z')) - |z| - |z'| - d \log t && \text{(by Equation (42))} \\ &> m + m' - c \log^3(n\ell) - c^3 \log(t) - O(\log \log n') - d \log t && \text{(by Equation (49))} \\ &= (\text{rK}^{p_G(t)}(x) - \log p_G(t) - \log^3 p_G(n\ell) - 1) + (\text{rK}^{p_G(t)}(y | x) - \log p_G(t) - \log^3 p_G(n\ell) - 1) \\ &\quad - c \log^3(n\ell) - c^3 \log(t) - O(\log \log n') - d \log t \\ &\geq \text{rK}^{p_{\text{Sol}}(t)}(x) + \text{rK}^{p_{\text{Sol}}(t)}(y | x) - \log p_{\text{Sol}}(t) - \log^3 p_{\text{Sol}}(n\ell), \end{aligned}$$

where the last inequality holds by letting p_{Sol} be a large enough polynomial. \blacktriangleleft

B.1.3 Coding Theorem for rK^t

► **Lemma 45** (An Efficient Coding Theorem for rK^t). *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_n$, there exists a polynomial p_{code} such that for every $n \in \mathbb{N}$ and $x \in \text{Support}(\mathcal{D}_n)$,*

$$\text{rK}^{p_{\text{code}}(n)}(x) \leq \log \frac{1}{\mathcal{D}_n(x)} + \log^3 p_{\text{code}}(n).$$

We need the following technical lemma.

► **Lemma 46** ([2, 1]; See also [11, Lemma 9.7]). *Let $\{\mathcal{D}_n\}_n$ be any polynomial-time samplable family of distributions. Then, there exist polynomials p and q such that, for every $n \in \mathbb{N}$ and every $x \in \text{Support}(\mathcal{D}_n)$,*

$$\Pr_{r \sim \{0,1\}^{q(n)}} \left[\mathsf{K}^{p(n)}(x, r) \leq \frac{1}{\mathcal{D}_n(x)} + |r| + \log p(n) \right] \geq \frac{1}{4}.$$

We now show Lemma 45.

Proof of Lemma 45. The proof is essentially the same as that of [11, Corollary 9.8].

Note that for any $x \in \{0,1\}^*$ and $t \in \mathbb{N}$, we have $\mathsf{rK}^t(x) \leq \mathsf{K}^t(x)$. On the one hand, by Lemma 46, we have for some polynomials p and q and for every $x \in \text{Support}(\mathcal{D}_n)$,

$$\Pr_{r \sim \{0,1\}^{q(n)}} \left[\mathsf{rK}^{p(n)}(x, r) \leq \frac{1}{\mathcal{D}_n(x)} + |r| + \log p(n) \right] \geq \frac{1}{4}. \quad (50)$$

On the other hand, by symmetry of information (Lemma 43), for every $x \in \{0,1\}^n$ and $r \in \{0,1\}^{q(n)}$, we have

$$\mathsf{rK}^{p(n)}(x, r) \geq \mathsf{rK}^{p_{\text{Sol}}(p(n))}(x) + \mathsf{rK}^{p_{\text{Sol}}(p(n))}(r | x) - \log p_{\text{Sol}}(p(n)) - \log^3 p_{\text{Sol}}(n \cdot q(n)). \quad (51)$$

Also, by a simple counting argument, we have for any fixed $x \in \{0,1\}^*$,

$$\Pr_{r \sim \{0,1\}^{q(n)}} \left[\mathsf{rK}^{p_{\text{Sol}}(p(n))}(r | x) \geq |r| - \log n \right] > \frac{3}{4}. \quad (52)$$

Combining Equations (51) and (52), we get that with probability greater than $3/4$,

$$\mathsf{rK}^{p(n)}(x, r) \geq \mathsf{rK}^{p_{\text{Sol}}(p(n))}(x) + |r| - \log p_{\text{Sol}}(p(n)) - \log^3 p_{\text{Sol}}(n \cdot q(n)) - \log n,$$

which, together with Equation (50), implies that there exists some r such that

$$\mathsf{rK}^{p_{\text{Sol}}(p(n))}(x) \leq \log \frac{1}{\mathcal{D}_n(x)} + \log p(n) + \log p_{\text{Sol}}(p(n)) + \log^3 p_{\text{Sol}}(n \cdot q(n)) + \log n.$$

The desired conclusion follows by choosing p_{code} to be a sufficiently large polynomial. ◀

B.1.4 Approximate Computational Depth for rK^t

In this subsection, we show an algorithm that can approximate the (randomized) computational depth of a given string.

► **Lemma 47.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist a constant $c > 0$, a polynomial τ and an algorithm `Approx-depth` that, on input $(x, 1^{t_1}, 1^{t_2}, 1^k)$, where $x \in \{0,1\}^n$, $t_1, t_2, k \in \mathbb{N}$ with $t_1, t_2 \geq cn$, runs in time $\text{poly}(n, t_1, t_2, k)$ and with probability $1 - 2^{-k}$ outputs an integer s such that*

$$\mathsf{rK}^{\tau(t_1)}(x) - \mathsf{rK}^{t_2}(x) \leq s \leq \mathsf{rK}^{t_1}(x) - \mathsf{rK}^{\tau(t_2)}(x) + \log \tau(t_1) + \log \tau(t_2) + \log^3 \tau(n).$$

To show Lemma 47, we need the following “worst-case-to-average-case reduction” result.

► **Lemma 48.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exists a polynomial τ such that the following promise problem is in prBPP :*

$$\begin{aligned} \text{YES} &:= \{(x, 1^s, 1^t) \mid \mathsf{rK}^t(x) \leq s\}, \\ \text{NO} &:= \{(x, 1^s, 1^t) \mid \mathsf{rK}^{\tau(t)}(x) > s + \log \tau(t) + \log^3 \tau(n)\}. \end{aligned}$$

Proof. Fix an instance $(x, 1^s, 1^t)$, where $x \in \{0, 1\}^n$. Without loss generality, we assume that $t \geq |x|$. Let $G_{(-)}$ be the generator from Lemma 42.

Since we assume that $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, There exist a constant $c' > 0$, a polynomial ρ and a probabilistic polynomial-time algorithm B such that the following hold for all sufficiently large n' , all $t' \geq \rho(n')$ and all $s' \leq n' - c' \cdot \log \log t'$, and .

1. If $y \in \{0, 1\}^{n'}$ and $\mathbf{K}^{t'}(y) \leq s'$, then $\Pr_B[B(y, 1^{s'}, 1^{t'}) = 1] \geq 1 - \frac{1}{10n'}$.
2. With probability at least $1/n'$ over $y \in \{0, 1\}^{n'}$, $\Pr_B[B(y, 1^{s'}, 1^{t'}) = 0] \geq 1 - \frac{1}{10n'}$.

Let $c > 0$ be a sufficiently large constant and consider the following parameters.

- $m := s + c^3 \cdot \log t + c \log^3 n$.
- $n' := m + t^c$.
- $s' := s + t^c + c^2 \cdot \log t + c \log^3 n$.
- $t' := t^{2c}$.

Now, define an algorithm B' as follows:

On input $(x, 1^s, 1^t)$ with $x \in \{0, 1\}^n$, sample $z \sim \{0, 1\}^{O(\log^3 n)}$ and $w \sim \{0, 1\}^{t^c}$, and then output $B(G_m(x; z) \circ w, 1^{s'}, 1^{t'})$.

Below, we show that B' solves (YES, NO) correctly with high probability in the worst case.

First, consider the case that $(x, 1^s, 1^t) \in \text{YES}$, i.e., $\mathbf{rK}^t(x) \leq s$. By amplification techniques (Lemma 10), we get that

$$\mathbf{rK}_{1-1/10n'}^{t^c}(x) \leq s + O(\log \log n').$$

In other words, there exists a *deterministic* program M of length at most $s + O(\log \log n')$ such that $U(M, w)$ outputs x within t^c steps for at least $1 - 1/10n'$ of the $w \in \{0, 1\}^{t^c}$. This implies that for any choice of $z \in \{0, 1\}^{O(\log^3 n)}$,

$$\Pr_{w \sim \{0, 1\}^{t^c}} \left[\mathbf{K}^{t^{2c}}(G_m(x; z) \circ w) \leq s + O(\log \log n') + t^c + O(\log^3 n) + O(c \log t) \right] \geq 1 - \frac{1}{10n'}. \quad (53)$$

Also, note that by letting c be a sufficiently large constant, we have

$$s + O(\log \log n') + t^c + O(\log^3 n) + O(c \log t) \leq s' \quad (54)$$

Equation (53) and Equation (54) together imply that

$$\Pr_{w \sim \{0, 1\}^{t^c}} \left[\mathbf{K}^{t'}(G_m(x; z) \circ w) \leq s' \right] \geq 1 - \frac{1}{10n'}.$$

Then by the property of B (Item 1) and a union bound, we have

$$\Pr_{w, z, B} \left[B(G_m(x; z) \circ w, 1^{s'}, 1^{t'}) = 1 \right] \geq 1 - \frac{1}{5n'},$$

and so

$$\Pr_{B'} \left[B'(x, 1^s, 1^t) = 1 \right] \geq 1 - \frac{1}{5n'}. \quad (55)$$

Now Let τ be a sufficiently large polynomial specified later, and consider any $(x, 1^s, 1^t) \in \text{NO}$, i.e., $\mathbf{rK}^{\tau(t)}(x) > s + \log \tau(t)$. We will show that

$$\Pr_{B'} \left[B'(x, 1^s, 1^t) = 1 \right] \leq 1 - \frac{2}{5n'}. \quad (56)$$

Note that by combining Equation (55) and Equation (56), B' yields an polynomial-time algorithm for (YES, NO) via standard success amplification techniques.

Suppose, for the sake of contradiction, Equation (56) is not true. Then by the definition of B' , we have

$$\Pr_{w,z,B} \left[B(G_m(x; z) \circ w, 1^{s'}, 1^{t'}) = 1 \right] > 1 - \frac{2}{5n'}. \quad (57)$$

On the other hand, by the property of B (Item 2), we get

$$\Pr_{u,w,B} \left[B(u \circ w, 1^{s'}, 1^{t'}) = 1 \right] < 1 - \frac{1}{2n'}. \quad (58)$$

Comparing Equation (57) and Equation (58), it is clear that $B(- \circ \mathcal{U}_{t^c}, 1^{s'}, 1^{t'})$ distinguishes $G_m(x; \mathcal{U}_{O(\log^3 n)})$ from \mathcal{U}_m with advantage $1/10n'$. By Lemma 42 and by letting τ be a sufficiently large polynomial, we get

$$\begin{aligned} \mathbf{rK}^{\tau(t)}(x) &\leq m + O(\log^3 n) + O(\log t') \\ &\leq s + c^3 \cdot \log t + c \log^3 n + O(\log^3 n) + O(c \log t) \\ &\leq s + \log \tau(t) + \log^3 \tau(n). \end{aligned}$$

This means $(x, 1^s, 1^t)$ is *not* in NO, which gives the desired contradiction. \blacktriangleleft

► **Corollary 49.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist a constant $c > 0$, a polynomial τ , and a probabilistic polynomial-time algorithm **Approx-rK** that, on input $(x, 1^t, 1^k)$ where $x \in \{0, 1\}^n$, $t \geq cn$ and $k \in \mathbb{N}$, with probability at least $1 - 2^{-k}$ outputs an integer s such that*

$$\mathbf{rK}^{\tau(t)}(x) - \log \tau(t) - \log^3 \tau(n) \leq s \leq \mathbf{rK}^t(x).$$

Proof. Consider a randomized polynomial-time algorithm A that solves the promise problem from Lemma 48. By standard error reduction techniques, assume without loss of generality that on the inputs satisfying the promise its error is at most $2^{-k}/n^2$, where $n = |x|$. Note that the running time of A becomes $\text{poly}(n, k)$. Our algorithm **Approx-rK** runs A on $(x, 1^s, 1^t)$ for $s = 1, 2, \dots, n + \log n$, and outputs the first s such that $A(x, 1^s, 1^t) = 1$.

The correctness of **Approx-rK** follows by a union bound. Indeed, if $s < \mathbf{rK}^{\tau(t)}(x) - \log \tau(t) - \log^3 \tau(n)$, i.e., $\mathbf{rK}^{\tau(t)}(x) > s + \log \tau(t) + \log^3 \tau(n)$, using the promise we get that $\Pr_A[A(x, 1^s, 1^t) = 1] \leq 2^{-k}/n^2$. On the other hand, if $s = \mathbf{rK}^t(x)$, which implies that $\mathbf{rK}^t(x) \leq s$ and the promise is satisfied, we have $\Pr_A[A(x, 1^s, 1^t) = 1] \geq 1 - 2^{-k}/n^2$. Since $\mathbf{rK}^t(x) \leq n + \log n$, if $t \geq cn$, where $c \geq 1$ is a sufficiently large constant, then with high probability over the internal randomness of **Approx-rK**, it outputs a value s such that $\mathbf{rK}^{\tau(t)}(x) - \log \tau(t) - \log^3 \tau(n) \leq s \leq \mathbf{rK}^t(x)$. \blacktriangleleft

We are now ready to prove Lemma 47.

Proof of Lemma 47. Let τ' be the polynomial from Corollary 49, and let **Approx-rK** be the algorithm from Corollary 49. By running **Approx-rK** $(x, 1^{t_1}, 1^{k+1})$, with probability at least $1 - 2^{-k}/2$, we get some integer s_0 such that

$$\mathbf{rK}^{\tau'(t_1)}(x) - \log \tau'(t_1) - \log^3 \tau'(n) \leq s_1 \leq \mathbf{rK}^{t_1}(x).$$

Similarly, by running **Approx-rK** $(x, 1^{t_2}, 1^{k+1})$, with probability at least $1 - 2^{-k}/2$, we get some integer s_2 such that

$$\mathbf{rK}^{\tau'(t_2)}(x) - \log \tau'(t_2) - \log^3 \tau'(n) \leq s_2 \leq \mathbf{rK}^{t_2}(x).$$

Then with probability at least $1 - 2^{-k}$, we have

$$s_1 - s_2 \leq \mathbf{rK}^{t_1}(x) - \left(\mathbf{rK}^{\tau'(t_2)}(x) - \log \tau'(t_2) - \log^3 \tau'(n) \right) \quad (59)$$

and

$$s_1 - s_2 \geq \left(\mathbf{rK}^{\tau'(t_1)}(x) - \log \tau'(t_1) - \log^3 \tau'(n) \right) - \mathbf{rK}^{t_2}(x). \quad (60)$$

We can then output

$$s := s_1 - s_2 + \log \tau'(t_1) + \log^3 \tau'(n).$$

Note that using Equations (59) and (60), we have

$$\begin{aligned} s &\leq \mathbf{rK}^{t_1}(x) - \mathbf{rK}^{\tau'(t_2)}(x) + \log \tau'(t_1) + \log \tau'(t_2) + 2 \cdot \log^3 \tau'(n) \\ &\leq \mathbf{rK}^{t_1}(x) - \mathbf{rK}^{\tau(t_2)}(x) + \log \tau(t_1) + \log \tau(t_2) + \log^3 \tau(n), \end{aligned}$$

and $s \geq \mathbf{rK}^{\tau'(t_1)}(x) - \mathbf{rK}^{t_2}(x) \geq \mathbf{rK}^{\tau(t_1)}(x) - \mathbf{rK}^{t_2}(x)$, where in the above we let $\tau > \tau'$ be a large polynomial. \blacktriangleleft

B.2 Proof of Theorem 41

In this subsection, we prove the following, which implies Theorem 41 via Proposition 11.

► **Theorem 50.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each \mathcal{D}_n is over $\{0, 1\}^n$, there exist a polynomial ρ and a probabilistic algorithm A such that the following hold for all $\lambda \in (0, 1)$, all $n, s, \ell, k \in \mathbb{N}$, and all $t \geq \rho(n) \cdot \log(1/(1-\lambda))$.*

1. *For all $x \in \{0, 1\}^n$, $A(x, \lambda, 1^t, 1^\ell, 1^k)$ runs in time $2^{O(\log^3 n)} \cdot \text{poly}(|\lambda|, t, \ell, k)$ and outputs either a program or \perp .*
2. *For all $x \in \{0, 1\}^n$,*

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) \text{ outputs neither an } (1/\ell)\text{-}\mathbf{rK}_\lambda^t\text{-witness of } x \text{ nor } \perp] \leq 2^{-k}.$$

3. *With probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,*

$$\Pr_A[A(x, \lambda, 1^t, 1^\ell, 1^k) = \perp] \leq 2^{-k}.$$

Proof. Throughout the proof, we will assume that $t \geq \rho(n) \cdot \log(1/(1-\lambda))$ for some polynomial ρ , which will be specified later. Here we assume without loss of generality that $\lambda \geq 2/3$. The proof can be easily adapted to the case where $\lambda \leq 2/3$.

Let $t \in \mathbb{N}$ be such that $t \geq p_0(3n)$, where p_0 is the polynomial from Lemma 43. Consider any $x \in \{0, 1\}^n$ and let y_t be a \mathbf{rK}_λ^t -witness of x . That is, y_t is a program such that $U(y_t, r)$ outputs x within t steps with probability at least λ over $r \sim \{0, 1\}^t$ and $|y_t| = \mathbf{rK}_\lambda^t(x)$. Also, let $q := \lceil 1/(1-\lambda) \rceil$. Note that $\log(q) \leq O(|\lambda|)$.

By symmetry of information (Lemma 43), we have, for some polynomial p_{Sol} ,

$$\begin{aligned} &\mathbf{rK}^{p_{\text{Sol}}(2t)}(y_t \mid x) \\ &\leq \mathbf{rK}^{2t}(x, y_t) - \mathbf{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(3n) \\ &\leq |y_t| - \mathbf{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(3n) + O(1) \\ &= \mathbf{rK}_\lambda^t(x) - \mathbf{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(3n) + O(1) \quad (\text{by the definition of } y_t) \\ &\leq \mathbf{rK}^{t/O(\log q)}(x) - \mathbf{rK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + \log^3 p_{\text{Sol}}(3n) + O(\log \log q), \\ &\hspace{15em} (\text{by Lemma 10}) \end{aligned}$$

where the second inequality follows from the fact that given y_t , one can also output x within t steps with probability at least $2/3$.

Let $t' := t/O(\log(1/(1-\lambda)))$. Then we have

$$\mathbf{rK}^{p_{\text{sol}}(2t)}(y_t | x) \leq \mathbf{rK}^{t'}(x) - \mathbf{rK}^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + \log^3 p_{\text{sol}}(3n) + O(\log |\lambda|). \quad (61)$$

Let $d > 0$ be some constant specified later, we say that $x \in \{0, 1\}^n$ is (t, k) -good if

$$\mathbf{rK}^{t'}(x) - \mathbf{rK}^{p_{\text{sol}}(2t)}(x) \leq d \cdot (\log t + \log^3 n) + \log k. \quad (62)$$

Consider any x, t, k such that x is (t, k) -good. Equation (61) implies that

$$\begin{aligned} \mathbf{rK}^{t^d}(y_t | x) &\leq \mathbf{rK}^{p_{\text{sol}}(2t)}(y_t | x) \\ &\leq \mathbf{rK}^{t'}(x) - \mathbf{rK}^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + \log^3 p_{\text{sol}}(3n) + O(\log \log q) \\ &\leq 2d \cdot (\log t + \log^3 n) + \log k + d \cdot \log |\lambda|, \end{aligned} \quad (63)$$

provided that d is a sufficiently large constant (which depends on p_{sol}).

Given Equation (63) and using standard success amplification techniques (Lemma 10), we get that for some sufficiently large constant $c > d$, there is a randomized program Π_{y_t} of length at most

$$s := c \cdot (\log^3 n + \log t + \log k + \log |\lambda|) \quad (64)$$

that, given x , outputs y_t within $T := t^c \cdot k^c$ steps with probability at least $1 - 2^{-k}/2$. We aim to find such a y_t .

Let Valid be the algorithm from Claim 21, and let A' be the following algorithm that, given $(x, \lambda, 1^t, 1^\ell, 1^k)$ such that x is (t, k) -good, aims to output an $(1/\ell)$ - \mathbf{rK}_λ^t -witness of x .

■ **Algorithm 4** Search for \mathbf{rK}^t -Witnesses for Good x 's.

```

1: procedure  $A'(x, \lambda, 1^t, 1^\ell, 1^k)$ 
2:    $n := |x|$ 
3:    $M := 0^{2n}$ 
4:    $s := c \cdot (\log^3 n + \log t + \log k + \log |\lambda|)$ , where  $c$  is the constant from Equation (64).
5:    $T := t^c \cdot k^c$ 
6:
7:   for  $\Pi \in \{0, 1\}^{\leq s}$  do
8:      $r :=$  a uniformly random string in  $\{0, 1\}^T$ .
9:      $y :=$  the output of  $U(\Pi, x, r)$  after running  $T$  steps.
10:    if  $|y| < |M|$  and  $\text{Valid}(x, y, \lambda, 1^t, 1^\ell, 1^{k+s+2})$  then
11:       $M := y$ 
12:  Output  $M$ 

```

It is easy to verify that $A'(x, \lambda, 1^t, 1^k, 1^\ell)$ runs in time $2^{O(\log^3 n)} \cdot \text{poly}(|\lambda|, t, k, \ell)$. Next, we argue that if x is (t, k) -good, then the above algorithm outputs an $(1/\ell)$ - \mathbf{rK}_λ^t -witness of x with probability $1 - 2^{-k}$.

Note that if x is (t, k) -good, then as described in previous paragraphs there is a randomized program Π_{y_t} of length at most $s := c \cdot (\log^3 n + \log t + \log k + \log |\lambda|)$ such that $U(\Pi_{y_t}, x, r)$ outputs y_t within $T := t^c \cdot k^c$ steps with probability at least $1 - 2^{-k}/2$ over $r \sim \{0, 1\}^T$. For such an x , our algorithm A' will successfully output an $(1/\ell)$ - \mathbf{rK}_λ^t -witness of x if both of the following are true.

1. The algorithm `Valid` succeeds (meaning that the condition stated in Claim 21 is satisfied) in all of the $m := \sum_{i=1}^s 2^i \leq 2^{s+1}$ executions, which happens with probability at least $1 - 2^m \cdot 2^{-k-s-2} \geq 1 - 2^{-k}/2$.
2. For $\Pi = \Pi_{y_t}$, $U(\Pi, x, r)$ outputs y_t within T steps, which happens with probability at least $1 - 2^{-k}/2$ over $r \sim \{0, 1\}^T$.

To see this, if the first item is true, then the randomized program M output by the algorithm is always a “valid” one that outputs x within t steps with probability at least $\lambda - 1/\ell$. If the second item is true, we are guaranteed that that $|M| \leq |y_t| = \text{rK}_\lambda^t(x)$, since $\text{Valid}(x, y_t, \lambda, 1^t, 1^\ell, 1^{k+s+1}) = 1$ (for a successful execution of `Valid`). The correctness of the algorithm then follows by a union bound.

We now describe our final algorithm A in the theorem. Let τ be the quasi-polynomial in Lemma 47, and let `Approx-depth` be the algorithm from Lemma 47. Our final algorithm A works as follows.

On input $(x, \lambda, 1^t, 1^\ell, 1^k)$, we first check if

$$\text{Approx-depth}\left(x, 1^{\lfloor \tau^{-1}(t') \rfloor}, 1^{p_{\text{sol}}(2t)}, 1^k\right) \leq d \cdot (\log t + \log^3 n) + \log k,$$

where d is the constant in Equation (62). If yes, we output $A'(x, \lambda, 1^t, 1^\ell, 1^k)$. Otherwise, we output \perp .

We argue that the algorithm A above satisfies the three conditions stated in the theorem. The first condition is easy to verify.

For the second condition, we consider two cases. Suppose x is not (t, k) -good, meaning that

$$\text{rK}^{t'}(x) - \text{rK}^{p_{\text{sol}}(2t)}(x) > d \cdot (\log t + \log^3 n) + \log k.$$

Note that by Lemma 47, in this case `Approx-depth` $\left(x, 1^{\lfloor \tau^{-1}(t') \rfloor}, 1^{p_{\text{sol}}(2t)}, 1^k\right)$ outputs, with probability at least $1 - 2^{-k}$, some s that satisfies

$$\begin{aligned} s &\geq \text{rK}^{\tau(\lfloor \tau^{-1}(t') \rfloor)}(x) - \text{rK}^{p_{\text{sol}}(2t)}(x) \\ &\geq \text{rK}^{t'}(x) - \text{rK}^{p_{\text{sol}}(2t)}(x) \\ &> d \cdot (\log t + \log^3 n) + \log k. \end{aligned}$$

Therefore, our algorithm will output \perp with probability at least $1 - 2^{-k}$.

Now suppose x is (t, k) -good. Then $A'(x, \lambda, 1^t, 1^\ell, 1^k)$ will output an $(1/\ell)$ - rK^t -witness of x with probability at least $1 - 2^{-\ell}$, which suffices to imply that the probability that our algorithm outputs neither an $(1/\ell)$ -optimal rK^t -witness of x nor \perp is at most 2^{-k} in this case, which also yields the second condition in this case.

Finally, for the third condition, we will show that in the above algorithm the criteria using `Approx-depth` will fail (hence output \perp) with probability at most $1/k$ over $x \sim \mathcal{D}_n$. To show this, we claim the following.

▷ **Claim 51.** For every $t, k \in \mathbb{N}$ such that $t \geq \rho(n)$, with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, we have

$$\begin{aligned} \zeta &:= \text{rK}^{\lfloor \tau^{-1}(t') \rfloor}(x) - \text{rK}^{\tau(p_{\text{sol}}(2t))}(x) + \log \tau(\lfloor \tau^{-1}(t') \rfloor) + \log \tau(p_{\text{sol}}(2t)) + \log^3 \tau(n) \\ &\leq d \cdot (\log t + \log^3 n) + \log k. \end{aligned}$$

Proof of Claim 51. Recall the coding theorem for rK (Lemma 45). By letting ρ be a sufficiently large polynomial so that for all $t \geq \rho(n) \cdot \log(1/(1-\lambda))$, it is satisfied that $\lfloor \tau^{-1}(t') \rfloor \geq p_{\text{code}}(n)$, where p_{code} is the polynomial from Lemma 45, we get that for every $x \in \text{Support}(\mathcal{D}_n)$,

$$\text{rK}^{\lfloor \tau^{-1}(t') \rfloor}(x) \leq \text{rK}^{p_{\text{code}}(n)}(x) \leq \log \frac{1}{\mathcal{D}_n(x)} + \log p_{\text{code}}(n). \quad (65)$$

On the other hand, by Lemma 9, with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$, we have

$$\text{K}(x) \geq \log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k,$$

where $b > 0$ is a constant. In particular, by Fact 6, this implies

$$\text{rK}^{\tau(p_{\text{sol}}(2t))}(x) \geq \log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k. \quad (66)$$

Combining Equations (65) and (66), we get that with probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\begin{aligned} \zeta &:= \text{rK}^{\lfloor \tau^{-1}(t') \rfloor}(x) - \text{rK}^{\tau(p_{\text{sol}}(2t))}(x) + \log \tau(\lfloor \tau^{-1}(t') \rfloor) + \log \tau(p_{\text{sol}}(2t)) + \log^3 \tau(n) \\ &\leq \left(\log \frac{1}{\mathcal{D}_n(x)} + \log p_{\text{code}}(n) \right) - \left(\log \frac{1}{\mathcal{D}_n(x)} - b \log n - \log k \right) \\ &\quad + \log t' + \log \tau(p_{\text{sol}}(2t)) + \log^3 \tau(n) \\ &= \log p_{\text{code}}(n) + b \log n + \log k + \log t' + \log \tau(p_{\text{sol}}(2t)) + \log^3 \tau(n) \\ &\leq d \cdot (\log t + \log^3 n) + \log k, \end{aligned}$$

where the last inequality holds by letting d be a sufficiently large constant. \triangleleft

To see that the third condition follows from Claim 51, note that by Lemma 47, we have $\text{Approx-depth}(x, 1^{\lfloor \tau^{-1}(t') \rfloor}, 1^{p_{\text{sol}}(2t)}, 1^k)$ outputs, with probability at least $1 - 2^{-k}$, some s such that $s \leq \zeta$. Then by Claim 51, we obtain that for at least $1 - 1/k$ fraction of the x sampled from \mathcal{D}_n , our algorithm will output something other than \perp with probability at least $1 - 2^{-k}$, as desired. \blacktriangleleft

C Search-to-Decision Reductions for the GapMINKT Problem

Mazor and Pass [30, Theorem 1.1] have recently described a sub-exponential time search-to-decision reduction for a gap version of time-bounded Kolmogorov complexity. In this section, we describe some related results obtained via techniques from meta-complexity.

Let MINKT denote the set of strings $(x, 1^s, 1^t)$ such that $\text{K}^t(x) \leq s$. We consider a gap version of the corresponding computational problem, defined as follows. For a polynomial p , we let $\text{Gap}_p \text{MINKT}$ denote the following promise problem:

- YES instances consist of strings $(x, 1^s, 1^t)$ such that $\text{K}^t(x) \leq s$;
- NO instances consist of strings $(x, 1^s, 1^t)$ such that $\text{K}^{p(t, |x|)}(x) > s + \log p(t, |x|)$.

We say that an algorithm A solves $\text{Search-Gap}_p \text{MINKT}$ if given any $\text{Gap}_p \text{MINKT}$ YES instance $(x, 1^s, 1^t)$, A outputs a program Π of length at most $s + \log p(t, |x|)$ such that $U^{p(t, |x|)}(\Pi) = x$. In other words, the algorithm certifies that $(x, 1^s, 1^t)$ is not a NO instance of $\text{Gap}_p \text{MINKT}$. We can think of A as providing an approximate or near-optimal solution to the search problem for K^t , since there is a bounded overhead in the running time and in the description length of the provided solution.

► **Theorem 52.** *Assume that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. If there is a polynomial p such that Gap_pMINKT admits a polynomial-time algorithm, then there is a polynomial q and a polynomial-time algorithm that solves $\text{Search-Gap}_q\text{MINKT}$.*

Proof. We rely on the *efficiency* and *bounded advice complexity* (under $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$) of the reconstruction procedure of the k -wise direct product generator $\text{DP}_k: \{0, 1\}^n \times \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk+k}$, defined as follows:

$$\text{DP}_k(x; z) := (z^1, \dots, z^k, \langle z^1, x \rangle, \dots, \langle z^k, x \rangle),$$

where $\langle z, x \rangle$ denotes the inner product of $z \in \{0, 1\}^n$ and $x \in \{0, 1\}^n$ over $\text{GF}(2)$. We will need the following result.

► **Lemma 53** (Reconstruction Lemma for K^t [11]). *Assume that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. There is a polynomial q_1 such that, for every $n \in \mathbb{N}$, $x \in \{0, 1\}^n$, parameter $k \in \mathbb{N}$, and for every deterministic circuit C of size ℓ such that*

$$\left| \Pr_z[C(\text{DP}_k(x; z)) = 1] - \Pr_w[C(w) = 1] \right| \geq 1/n,$$

where $z \sim \{0, 1\}^{nk}$ and $w \in \{0, 1\}^{nk+k}$, it holds that

$$\text{K}^{q_1(n \cdot \ell)}(x | C) \leq k + \log q_1(n \cdot \ell).$$

Moreover, there is a deterministic algorithm B that, given $x \in \{0, 1\}^n$, $k \in \mathbb{N}$, and C , runs in polynomial time and outputs a string y of length at most $k + \log q_1(n, \ell)$ such that $U^{q_1(n, \ell)}(y, C) = x$.

Sketch of the Proof of Lemma 53. The assumption that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ yields a pseudorandom generator G of seed length $O(\log m)$ that allows us to derandomize non-uniform algorithms of complexity at most m [20]. In the construction described below, we can use G to derandomize any internal procedure of the program that outputs x given C . We note that by fixing a good seed of G in such a derandomization, we will incur an overhead in the description length of the program for x of at most $\log \text{poly}(n, \ell)$ bits, while the overhead in the running time of the program is $\text{poly}(n, \ell)$. These overheads do not create an issue because we can take the polynomial q_1 to be of large enough degree. (Moreover, it would be enough to design a randomized algorithm B , since this algorithm can be derandomized in a standard way by trying all seeds of the generator and outputting the first valid program with the desired parameters.)

Using the circuit C as a distinguisher and Yao's equivalence between breaking a candidate generator and next-bit (un)predictability, it follows that there is an index $i \in [k]$ such that $\langle z^i, x \rangle$ can be predicted with probability at least $1/2 + \Omega(1/(n \cdot k))$, given $z^1, \dots, z^k, \langle z^1, x \rangle, \dots, \langle z^{i-1}, x \rangle$ as input. Since $\langle z^i, x \rangle$ is the z^i -th bit of the Hadamard code of x , we can use the next-bit predictor and the list-decoding algorithm of the Hadamard code to recover with noticeable probability a list of strings of polynomial size that contains x . More precisely, this yields a randomized algorithm M (with access to C) that runs in time polynomial in n and ℓ such that, given a random choice of z^1, \dots, z^k and the corresponding bits $\langle z^1, x \rangle, \dots, \langle z^{i-1}, x \rangle$, outputs with probability at least $1/\text{poly}(n, \ell)$ a list S of size $\text{poly}(n, \ell)$ that contains x .

As explained above, using the generator G , a good choice of the random strings z^1, \dots, z^k as well as of the internal randomness of M can be obtained from some seed $\sigma \in \{0, 1\}^s$, where $s = O(\log \text{poly}(n, \ell))$. Additionally, the $i - 1 \leq k$ "advice bits" $\langle z^1, x \rangle, \dots, \langle z^{i-1}, x \rangle$

can be efficiently computed from x and z^1, \dots, z^{i-1} . These advice bits are stored in the corresponding program y witnessing that $K^{q_1(n, \ell)}(x \mid C) \leq k + \log q_1(n \cdot \ell)$. Finally, the position of x in the list S can be encoded with $O(\log \text{poly}(n, \ell))$ bits of advice and is also stored in y .

The search algorithm B from the “moreover” part of the result tries all seeds σ of the generator until a good seed is found, a condition that can be tested by running the corresponding program y_σ . The overall running time of B is $\text{poly}(n, \ell)$, as desired. ◀

We now describe the search-to-decision reduction. Assume that $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$. Let F be a polynomial-time algorithm that decides Gap_pMINKT . Note that we can assume without loss of generality that F is deterministic, since $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ yields strong pseudorandom generators [20]. For a large enough polynomial q specified later in the proof, we describe a deterministic polynomial-time algorithm A that solves $\text{Search-Gap}_q\text{MINKT}$, i.e., given any $\text{Gap}_q\text{MINKT YES}$ instance $(x, 1^s, 1^t)$, A outputs a $q(t, |x|)$ -time-bounded description of x of length at most $s + \log q(t, |x|)$.

Let $\alpha \geq 1$ be a large enough constant. We assume that $s \leq n - 10\alpha \cdot \log n$, since otherwise a trivial description of x is a correct output for Gap_qMINKT (taking q to be of large enough degree). We can also assume that $t \geq n$, as the polynomial q can depend on $n = |x|$. The search algorithm A sets $k = s + \alpha \cdot \log n$ in the execution of the algorithm B from Lemma 53, and let $C(v)$ be the (deterministic) circuit of size $\ell = \text{poly}(t)$ obtained from F on inputs of the form $(v, 1^{s'}, 1^{t'})$, where $|v| = nk + k$, $s' = nk + k - (\alpha/4) \cdot \log n$, and $t' = q_2(t)$, for a polynomial q_2 of large enough degree.

For a given $\text{Gap}_q\text{MINKT YES}$ instance $(x, 1^s, 1^t)$, it is not hard to see that for every string $z \in \{0, 1\}^{nk}$,

$$K^{q_2(n)}(\text{DP}_k(x; z)) \leq |z| + K^t(x) + O(\log n) \leq nk + s + O(\log n) \leq nk + k - (\alpha/2) \cdot \log n,$$

where we used that α is large enough. On the other hand, for a random string $w \sim \{0, 1\}^{nk+k}$, a simple counting argument gives that

$$K(w) \geq nk + k - \log n$$

with probability at least $1/n$. Recall that $C(v)$ accepts every instance v such that $K^{t'}(v) \leq s'$ and rejects every instance v such that $K^{p(t', |v|)}(v) > s' + \log p(t', |v|)$. Consequently, due to our choices of s' and t' and using a large enough α , it is not hard to see that $C(v)$ satisfies the condition in the statement of Lemma 53.

Therefore, the algorithm B on inputs x , $k = s + \alpha \cdot \log n$, and C (as defined above), runs in time $\text{poly}(x, k, |C|) = \text{poly}(n, s, t)$ and outputs a string y of length at most

$$k + \log q_1(n, \ell) = s + \alpha \cdot \log n + \log q_1(n, \text{poly}(t))$$

such that $U^{q_1(n, \ell)}(y, C) = x$. Since C can be efficiently computed from the code of F (which has description length $O(1)$) and parameters s and t (which can be described with $\log n + \log t$ bits), if we take q to be a large enough polynomial, A can produce from y a string Π of length at most $s + \log q(t, |x|)$ such that $U^{q(t, |x|)}(\Pi) = x$. This completes the proof of Theorem 52. ◀

► **Remark 54 (Comparison with [30, Theorem 1.1]).** On the one hand, the (black-box) search-to-decision reduction in [30, Theorem 1.1] is unconditional, while Theorem 52 relies on a standard derandomization assumption and is non-black-box. On the other hand, Theorem 52 provides a *polynomial*-time search to decision reduction for GapMINKT , as opposed to the *sub-exponential* running time of [30, Theorem 1.1].

► **Remark 55** (Exact versus Gap Search-to-Decision Reductions for K^t). We note that the assumption in Theorem 52 on the existence of a polynomial p such that Gap_pMINKT admits a polynomial-time algorithm is implied by “ $\text{MINKT} \in \text{AvgBPP}$ ” and $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$ (see [10, 9] or [12]). In other words, the assumption on the easiness of Gap_pMINKT can be replaced by “ $\text{MINKT} \in \text{AvgBPP}$ ” in Theorem 52. In particular, under the assumptions of Theorem 1 we can efficiently solve the exact search problem for K^t on average and the gap search problem for K^t in the worst case.

We observe that it is possible to derive a weaker unconditional consequence from Theorem 52 via a win-win argument. We will need the following simple result.

► **Proposition 56.** *Suppose that $E \subseteq \text{i.o.SIZE}[2^{o(n)}]$. Then, for every $\varepsilon > 0$, there exists infinitely many values of n and a circuit C_n of size at most $2^{\varepsilon \cdot n}$ such that, given a string $x \in \{0, 1\}^n$ and $1 \leq t \leq 2^n$ (represented as an n -bit string), $C_n(x, t)$ outputs a minimum length program Π such that $U^t(\Pi) = x$.*

Proof. Under the assumption, for every $L \in E$ and for every $\varepsilon > 0$, there is an infinite set $S \subseteq \mathbb{N}$ such that, for every $n \in S$, there is a circuit C_n of size at most $2^{\varepsilon \cdot n}$ that computes L_n , i.e., L restricted to inputs of length exactly n .

We consider a paddable language L (with a padding input parameter k) that contains all tuples $\langle x, w, i, s, t, 1^k \rangle$ such that:

- (i) $|x| = n$ for some n , $|w| = n$, $|i| = \log n$, $|s| = \log n$, $|t| = n$, and k is arbitrary. We view i as an integer such that $1 \leq i \leq n$, and t as an integer such that $1 \leq t \leq 2^n$.
- (ii) Let $w_{\leq i}$ be the i -th bit prefix w . There is a program Π that extends w_i and is of length at most s such that $U^t(\Pi) = x$.

We assume that the tuples in L employ an encoding such that the bit-length of $\langle x, w, i, s, t, 1^k \rangle$ as a string is precisely $4n + k$, whenever n is sufficiently large. This is easy to get, since $|x| + |w| + |i| + |s| + |t| = 3n + 2 \log n$ for positive instances. The particular choice of encoding is not important as long as the tuple can be efficiently encoded and decoded.

Observe that $L \in E$. Under the assumption of Proposition 56, for every $\delta > 0$ there are infinitely many input lengths m such that L on input length m admits a circuit D_m of size at most $2^{\delta \cdot m}$. Using the padding parameter k , it is not hard to see that we can use D_m to decide tuples $\langle x, w, i, s, t, 1^k \rangle$ with $|x| = m/5$. Finally, let $n = |x|$. Given D_m , by a standard binary search over prefixes of w , we can optimally solve the search problem for K^t on x in size $2^{\delta \cdot m} \cdot \text{poly}(m) \leq 2^{6 \cdot \delta \cdot n}$. Since $\delta > 0$ can be taken arbitrarily small, the result follows. ◀

By combining Theorem 52 and Proposition 56, we get the following unconditional search-to-decision reduction. (Since we consider Boolean circuits in the next statement, which are devices that operate over fixed input lengths, we assume an upper bound on the input parameters as a function of n .)

► **Theorem 57.** *If there is a polynomial p such that Gap_pMINKT admits a polynomial-time algorithm, then there is a polynomial q such that, for every $\varepsilon > 0$, there are infinitely many input lengths n such that $\text{Search-Gap}_q\text{MINKT}$ can be solved by a circuit of size at most $2^{\varepsilon \cdot n}$ on inputs $(x, 1^s, 1^t)$, where we assume that $x \in \{0, 1\}^n$, $1 \leq s \leq n + \log n$, and $1 \leq t \leq 2^{o(n)}$.*

Proof. If $E \not\subseteq \text{i.o.SIZE}[2^{o(n)}]$, the result immediately follows from Theorem 52. Otherwise, we get that $E \subseteq \text{i.o.SIZE}[2^{o(n)}]$. Let C_n be one of the circuits from Proposition 56. Then we can use C_n to solve the search problem of any Gap_qMINKT YES instance $(x, 1^s, 1^t)$. This completes the proof. ◀

Note that, in contrast to the search-to-decision reduction from [30, Theorem 1.1], which provides a *uniform* algorithm for $\text{Search-Gap}_q\text{MINKT}$ with the sub-exponential-time $2^{\varepsilon \cdot s} \cdot \text{poly}(n, t, s)$ (for every $\varepsilon > 0$), Theorem 57 only provides a non-uniform infinitely often sub-exponential-time $2^{\varepsilon \cdot n}$ algorithm (for every $\varepsilon > 0$), and so has similar sub-exponential in s efficiency only for $s \in \Omega(n)$.⁹

D Errorless Average-Case Search-to-Decision Reduction for K^t over the Uniform Distribution

In this section, we describe polynomial-time errorless average-case search-to-decision reduction over the uniform distribution for K^t . We get the following polynomial-time average-case search-to-decision reduction for K^t in the errorless setting over the uniform distribution. This complements a similar result in [22], which holds in the error-prone setting.

► **Theorem 58.** *If $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, then there exist a polynomial ρ and a probabilistic polynomial-time algorithm A such that the following holds for all $n, s, k \in \mathbb{N}$, and all $t \geq \rho(n)$.*

1. For all $x \in \{0, 1\}^n$,

$$\Pr_A[A(x, 1^t, 1^k) \text{ outputs either an } \mathsf{K}^t\text{-witness of } x \text{ or } \perp] \geq 1 - \frac{1}{2^k}.$$

2. With probability at least $1 - 1/k$ over $x \sim \mathcal{D}_n$,

$$\Pr_A[A(x, 1^t, 1^k) \text{ outputs } \perp] \leq \frac{1}{2^k}.$$

Proof. The proof follows a similar approach to that of Theorem 50.

Let $n \in \mathbb{N}$ and let $t \geq \rho(n)$ for some polynomial ρ specified later.

Consider any $x \in \{0, 1\}^n$ and let y_t be a K^t -witness of x .

By the assumption that $(\text{coMINKT}, \mathcal{U}) \in \text{Avg}^1\text{BPP}$ holds, it follows from Lemma 17 that there exist polynomials p_{Sol} such that

$$\begin{aligned} \mathsf{pK}^{p_{\text{Sol}}(2t)}(y_t \mid x) &\leq \mathsf{pK}^{2t}(x, y_t) - \mathsf{pK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) \\ &\leq \mathsf{K}^{2t}(x, y_t) - \mathsf{pK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) \\ &\leq |y_t| - \mathsf{pK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) \\ &= \mathsf{K}^t(x) - \mathsf{pK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t). \end{aligned}$$

Using standard amplification techniques for probabilistic time-bounded Kolmogorov complexity (see, e.g., [6, Lemma 21]), we get

$$\mathsf{pK}_{1-2^{-k}}^{p_{\text{Sol}}(2t) \cdot \text{poly}(k)}(y_t \mid x) \leq \mathsf{K}^t(x) - \mathsf{pK}^{p_{\text{Sol}}(2t)}(x) + \log p_{\text{Sol}}(2t) + O(\log k). \quad (67)$$

Let $d > 0$ be a constant determined later. We say that x is (t, k) -good if

$$\mathsf{pK}^{p_{\text{Sol}}(2t)}(x) > n - d \cdot \log t - \log k.$$

⁹ In Theorem 57 there is a dependence on n in the exponent of the circuit size, as opposed to a dependence on s in the running time as in [30, Theorem 1.1]. This is inherent in the non-uniform model when the parameter s is part of the input, since the circuit is fixed and must work for all values of s including $s = \Theta(n)$. In other words, in a uniform algorithm the running time can depend on a given input instance, but in a circuit its size is fixed for all inputs of a given input length.

Note that if x is (t, k) -good, then the quantity in Equation (67) becomes

$$\begin{aligned} \mathsf{pK}_{1-2^{-k}}^{(t,k)^d}(y_t | x) &\leq \mathsf{K}^t(x) - \mathsf{pK}^{p_{\text{sol}}(2t)}(x) + \log p_{\text{sol}}(2t) + O(\log k) \\ &< (n + O(1)) - (n - d \cdot \log t - \log k) + \log p_{\text{sol}}(2t) + O(\log k) \\ &\leq 2d \cdot (\log t + \log k), \end{aligned}$$

if we let d be a sufficiently large constant. This implies that for at least $1 - 2^{-k}$ fraction of the $w \in \{0, 1\}^T$, where $T := (tk)^d$, there is a program Π_{y_t} of size at most $s := 2d \cdot (\log t + \log k)$ such that $U(\Pi, w)$ outputs y_t within T steps. Therefore, the following procedure A' will be able to find a K^t -witness of x with probability at least $1 - 2^{-k}$.

On input $(x, 1^t, 1^k)$, we pick $w \sim \{0, 1\}^T$, enumerate all $\Pi \in \{0, 1\}^{\leq s}$, run $U(\Pi, w)$ for T steps and obtain a list of candidate programs $\{y\}$ (which is guaranteed to contain y_t). We then return the shortest y that outputs x within t steps.

Let M be the randomized algorithm that approximates pK^t as in Lemma 16. By standard amplification techniques, we can amplify its success probability to be $1 - 2^{-k}$, by blowing up the running time by at most $\text{poly}(k)$. Consider the following algorithm **Certify**:

On input $(x, 1^t, 1^k)$, let $t' := p_{\text{sol}}(2t)$ and let $\theta := n - d \cdot \log t - \log k$. We accept if and only if $M(x, 1^{t'}) \geq \theta$.

Our final algorithm A works as follows:

On input $(x, 1^t, 1^k)$, we runs **Certify** $(x, 1^t, 1^k)$, if it rejects, we output \perp ; otherwise, we run $A'(x, 1^t, 1^k)$ and output whatever it outputs.

We show the first condition of Theorem 58. Note that on the one hand, if x is not (t, k) -good, then by the correctness of M , **Certify** $(x, 1^t, 1^k)$ will reject with probability at least $1 - 2^{-k}$ over its internal randomness.

On the other hand, if x is indeed (t, k) -good, then our algorithm A' will return a K^t -witness of x with probability with probability at least $1 - 2^{-k}$ over its internal randomness.

To see the second condition, note that by a simple counting argument, with probability at least $1 - 1/k$ over $x \sim \{0, 1\}^n$, it holds that

$$\begin{aligned} \mathsf{pK}^{\tau(p_{\text{sol}}(2t))}(x) &\geq \mathsf{K}(x) - O(\log \tau(p_{\text{sol}}(2t))) \\ &\geq n - O(\log \tau(p_{\text{sol}}(2t))) - \log k \\ &> n - d \cdot \log t - \log k, \end{aligned}$$

where the last inequality holds if we choose d to be a sufficiently large constant. Again, by the correctness of M , this implies that **Certify** $(x, 1^t, 1^k)$ will accept at least $(1 - 1/k)$ -fraction of $x \in \{0, 1\}^n$ (with probability at least $1 - 2^{-k}$ over its internal randomness). ◀

The Computational Advantage of MIP^* Vanishes in the Presence of Noise

Yangjing Dong  



State Key Laboratory for Novel Software Technology, Nanjing University, China

Honghao Fu  

Massachusetts Institute of Technology, Cambridge, MA, USA

Anand Natarajan  


Massachusetts Institute of Technology, Cambridge, MA, USA

Minglong Qin  

State Key Laboratory for Novel Software Technology, Nanjing University, China

Haochen Xu  

State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

Penghui Yao  

State Key Laboratory for Novel Software Technology, Nanjing University, China
Hefei National Laboratory, China

Abstract

The class MIP^* of quantum multiprover interactive proof systems with entanglement is much more powerful than its classical counterpart MIP [7, 29, 28]: while $MIP = NEXP$, the quantum class MIP^* is equal to RE , a class including the halting problem. This is because the provers in MIP^* can share unbounded quantum entanglement. However, recent works [46, 47] have shown that this advantage is significantly reduced if the provers' shared state contains noise. This paper attempts to exactly characterize the effect of noise on the computational power of quantum multiprover interactive proof systems. We investigate the quantum two-prover one-round interactive system MIP^* [poly, $O(1)$], where the verifier sends polynomially many bits to the provers and the provers send back constantly many bits. We show noise completely destroys the computational advantage given by shared entanglement in this model. Specifically, we show that if the provers are allowed to share arbitrarily many EPR states, where each EPR state is affected by an arbitrarily small constant amount of noise, the resulting complexity class is equivalent to $NEXP = MIP$. This improves significantly on the previous best-known bound of $NEEEXP$ (nondeterministic triply exponential time) [46]. We also show that this collapse in power is due to the noise, rather than the $O(1)$ answer size, by showing that allowing for noiseless EPR states gives the class the full power of $RE = MIP^*$ [poly, poly]. Along the way, we develop two technical tools of independent interest. First, we give a new, deterministic tester for the positivity of an exponentially large matrix, provided it has a low-degree Fourier decomposition in terms of Pauli matrices. Secondly, we develop a new invariance principle for smooth matrix functions having bounded third-order Fréchet derivatives or which are Lipschitz continuous.

2012 ACM Subject Classification Theory of computation \rightarrow Quantum complexity theory

Keywords and phrases Interactive proofs, Quantum complexity theory, Quantum entanglement, Fourier analysis, Matrix analysis, Invariance principle, Derandomization, PCP, Locally testable code, Positivity testing

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.30

Related Version *Full Version*: <https://arxiv.org/abs/2312.04360> [17]

Funding *Yangjing Dong*: supported by National Natural Science Foundation of China (Grant No. 62332009, 12347104, 61972191) and Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302901).



© Yangjing Dong, Honghao Fu, Anand Natarajan, Minglong Qin, Haochen Xu, and Penghui Yao;
licensed under Creative Commons License CC-BY 4.0

39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 30; pp. 30:1–30:71



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Honghao Fu: supported by the US National Science Foundation QLCI program (grant OMA-2016245).

Minglong Qin: supported by National Natural Science Foundation of China (Grant No. 62332009, 12347104, 61972191) and Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302901).

Haochen Xu: supported by the Key Research Program of the Chinese Academy of Sciences, Grant NO. ZDRW-XX-2022-1

Penghui Yao: supported by National Natural Science Foundation of China (Grant No. 62332009, 12347104, 61972191) and Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302901).

Acknowledgements P.Y. would like to thank the discussion with Zhengfeng Ji. Part of the work was done when H.F. and H.X. visited Nanjing University.

1 Introduction

The power of entanglement in computation has been a central topic in the theory of quantum computing. In particular, the effect of entanglement in multiprover interactive proof systems has been studied for decades [33, 32, 25, 27, 54, 53] leading to the seminal result $\text{MIP}^* = \text{RE}$ [29, 28] due to Ji, Natarajan, Vidick, Wright, and Yuen, which states that all recursively enumerable languages can be decided by multiprover interactive proof systems empowered by quantum entanglement. More precisely, the system only has two provers, one round of interaction between the provers and the verifier, and the provers share arbitrarily many copies of the EPR state.

Given the appearance of intractable complexity classes like RE in the previous result, a natural question is to what extent the body of results on MIP^* are relevant to the physical world. Of course, in reality, devices do not have access to unbounded numbers of perfect EPR pairs; in a sense, what $\text{MIP}^* = \text{RE}$ means is that the power of two entangled grows unboundedly as the number of shared EPR pairs increases, even when the message size is constrained to be polynomial. In fact, using a finite number of iterations of the “compression” procedure from $\text{MIP}^* = \text{RE}$, one can show that the class $\text{NTIME}[T(n)]$ for $T(n)$ any finite tower of exponentials has an MIP^* protocol, where the provers need only share a finite number of perfect EPR pairs scaling roughly with $\log T(n)$. However, the requirement that the EPR pairs be perfect seems essential to these protocols. The question naturally arises whether similar complexity results can be obtained even when the provers have access to *imperfect* entanglement only.

To isolate the role played by noise, in this work we ask the following question: what is the power of MIP^* when the provers are given access to an *unbounded* number of *imperfect* EPR pairs, where each EPR pair is independently perturbed by a constant amount of depolarizing noise? (We choose this noise model because it is mathematically elegant and also physically relevant, as recent experiments suggest that the dominating noise is the localized depolarizing noise in the neutral atom platform [11].) On the one hand, known MIP^* protocols all break down with states of this form. On the other hand, according to standard measures of entanglement such as distillable entanglement and entanglement of formation, such states have entanglement that grows unboundedly as the number of copies goes to infinity. Thus, it seems *a priori* reasonable that the corresponding MIP^* class may also have unbounded power.

It is worth noting that this question is orthogonal to fault tolerance in quantum devices. As usual in MIP^* , we assume that the provers are computationally unbounded, and may perform any quantum operation of their choice with no error. Nevertheless, this does not

mean they can use techniques from fault tolerance to simulate provers with noiseless entangled states. This is because the provers cannot jointly correct their shared entangled state, since they are not allowed to communicate in this model.

This question is closely related to the quantum information primitive of self-testing. Self-tests are essentially MIP^* protocols that certify physical properties of quantum states, rather than computational statements. The protocols in $\text{MIP}^* = \text{RE}$ all rely on highly efficient self-tests for EPR pairs, but these tests are not at all tolerant of noise. Designing self-tests that *are* tolerant to noise, and certify some useful measure of entanglement, is a current research question [5, 3], and studying the power of MIP^* in the presence of noise gives us insight on this question from a different angle. In particular, for an entangled state ρ , one can think of the power of the complexity class $\text{MIP}^*[\rho]$ where the provers are restricted to sharing copies of ρ , as a particular operational measure of the amount of useful entanglement in ρ . In passing, we remark that recent work of Vidick, Arnon-Friedman and Brakerski has studied “computationally efficient” measures of entanglement from somewhat different perspective [4].

The first partial answer to this question was given by Qin and Yao [46]. They investigated two-player nonlocal games¹ when the states shared between the players are arbitrarily many copies of a maximally entangled state (MES) with an arbitrarily small but *constant* amount of noise on each copy, which is termed as *noisy MES* in their paper. They showed that the supremum winning probability over all strategies using these states can be computably approximated to any finite precision. In fact, they showed that for any ε , there is a number of copies of the noisy MES, which is a computable function of only ε and the size of the nonlocal game, that is sufficient to achieve winning probability within ε of this supremum. This implies that any language in MIP^* restricted to such states is decidable, meaning that this class is strictly smaller than RE.

This result was later generalized to nonlocal games that allow quantum questions and quantum answers [47]. To put these results in the language of complexity classes, let $\text{MIP}^*[q, a, \psi]$ be the set of languages that are decidable in the model of two-prover, one-round quantum multiprover interactive proof systems, where the provers share arbitrarily many copies of ψ , the messages from the verifier are classical and q -bits long, and the messages from the provers are also classical and a -bits long. If the messages are quantum, the complexity class is denoted by $\text{QMIP}[q, a, \psi]$. Thus, the prior work implies that both the class $\text{QMIP}[\text{poly}, \text{poly}, |\text{EPR}\rangle]$ and the class $\text{MIP}^*[\text{poly}, \text{poly}, |\text{EPR}\rangle]$ are equal to RE [49, 29, 28], while both the complexity classes $\text{MIP}^*[\text{poly}, \text{poly}, \psi]$ and $\text{QMIP}[\text{poly}, \text{poly}, \psi]$ are computable if ψ is a noisy MES state [46, 47]. Moreover, [46, 47] showed explicit, though very large, time bounds for computing approximations to the game value for noisy states.

Although these results show that the full power of MIP^* is not robust against noise in the shared entanglement, it is still possible that multiprover interactive proof systems gain a finite but very large computational advantage by sharing noisy maximally entangled states, since the time bounds from the previous work are much larger than for the classes with no entanglement. Thus, it was consistent with prior work that $\text{MIP}^*[\text{poly}, \text{poly}, \psi]$ is contained in nondeterministic quadruply exponential time complexity class for noisy ψ [46], which is much more powerful than $\text{MIP}[\text{poly}, \text{poly}] = \text{NEXP}$. This paper attempts to answer this question by investigating the complexity classes $\text{MIP}^*[\text{poly}, O(1), \psi]$ (i.e. protocols with constant-size answers) when ψ is a noisy MES, whose local dimension is a constant. Classically, it is known that $\text{MIP}[\text{poly}, \text{poly}] = \text{MIP}[\text{poly}, O(1)] = \text{NEXP}$ [7]. Our main result, stated in the language of nonlocal games, is the following.

¹ An MIP^* protocol is essentially a uniform family of two-player nonlocal games, with efficient algorithms for sampling pairs of questions and for evaluating the game decision predicate.

► **Theorem 1 (Informal).** *Given a nonlocal game in which the players share arbitrarily many copies of a noisy MES ψ , and the size of the answer sets is constant, then approximating the value of the game up to any sufficiently small constant precision is NP-complete.*

The runtime in Theorem 1 is measured in terms of the size of a description of the nonlocal game as a table containing the distribution over question pairs and the verifier’s predicate for every tuple of questions and answers. Translating this result to the MIP* world requires parametrizing the runtime in terms of the *number of bits* in the questions and answers. Thus, Theorem 1 shows that MIP* with $O(\log(n))$ -bit questions and $O(1)$ -bit answers is NP-complete. Scaling our result up to MIP* protocols with $O(\text{poly}(n))$ -bit questions and $O(1)$ -bit answers, we get the following.

► **Corollary 2.** $\text{MIP}^*[\text{poly}, O(1), \psi] = \text{NEXP}$, where ψ is a noisy MES.

Intuitively, Theorem 1 says that for any nonlocal game, if the shared MES has constant noise, the players’ optimal strategy has a concise classical description which is also easy to verify. It is interesting to compare such nonlocal games with their classical counterparts. Håstad in his seminal work [23] proved that it is NP-hard to approximate the value of a classical nonlocal game to a constant precision even if the size of the answer set is a constant. It is also worth noting that sharing entanglement does not always strengthen the hardness of nonlocal games. It may weaken the hardness of certain games as well. For example, the quantum XOR games and quantum unique games are easy [15, 33], while the classical XOR games are NP-hard, and the classical unique games are conjectured to be NP-hard as well [34]. Thus introducing noisy quantum states doesn’t introduce any quantum effect to the hardness at all.

One may wonder whether this surprising collapse in complexity is caused by the restriction to noisy states or the restriction to $O(1)$ -size answers. We give strong evidence that it is the former, by showing that MIP* with *noiseless* states and $O(1)$ -sized answers is still equal to RE.

► **Theorem 3 (Theorem 36).** *RE is equal to $\text{MIP}^*[\text{poly}, O(1), |\text{EPR}\rangle]$ with completeness 1 and constant soundness.*

To put this in context, the original work [29, 28] proves that nonlocal games with noiseless EPR states are RE-complete to approximate if both the question set and answer set are of *polynomial* size. Very recently, Natarajan and Zhang [40] proved, by repeatedly applying the “question reduction” technique from [28], that it is still RE-complete if the question length is $O(1)$ and the answer length is $\text{polylog}(n)$, respectively. Here, we achieve constant *answer* length by one application of an “answer reduction” transformation: the error-correcting code-based scheme of [39], instantiated with the Hadamard code.

Theorems 1 and 3 give us strong evidence that the computational power of MIP* will vanish in the presence of noise. So for any complexity class slightly larger than NEXP, we cannot hope for an MIP* protocol robust against noise. They also suggest that the key resource behind the computational power of MIP* is specifically copies of the MES state, not just entanglement. This is because as we remarked above, as n tends to infinity, n copies of a noisy MES contain an amount of entanglement going to infinity under standard entanglement measures. Alternately, using the power of $\text{MIP}^*[\psi]$ as a measure of entanglement for ψ , we show that an MES and an ε -noisy MES are sharply separated by this measure for any constant ε .

Since efficient self-tests for large entangled states are the key technique behind the proof of $\text{MIP}^* = \text{RE}$, our result puts some limitations on the design of self-tests robust against noise. More specifically, our results suggests that to noise-robustly self-test larger entangled states, the numbers of questions and answers must grow with the dimension of the tested state. For comparison, if we don’t need a self-test to be noise-robust, this is not necessary [19].

1.1 Proof Overview

1.1.1 Approximating the Values of Noisy Games is NP-Complete

The harder part is to show that there is an NP-algorithm for this problem, so we give an overview of this algorithm first.

Given a nonlocal game sharing arbitrary copies of a noisy MES ψ , Qin and Yao [46] showed that it suffices for the players to share D copies of ψ to achieve the value of the game to an arbitrarily small precision, where D only depends on the size of the game and the precision.

We first improve the upper bound D to make it only depend exponentially on the length of the questions instead of doubly exponentially as in [46]. To prove this upper bound, we use ideas from Fourier analysis. For illustration, let's assume $\psi = \rho |EPR\rangle\langle EPR| + (1 - \rho)\mathbb{1}_2/2 \otimes \mathbb{1}_2/2$ is a depolarized noisy EPR state for simplicity. Given a strategy S , let P be a POVM element from the strategy, which acts on n qubits. We are going to show the upper bound is independent of n , so in the rest of the section by "constant" we mean independent of n . Let the Pauli expansion of P be

$$P = \sum_{\sigma \in \{0,1,2,3\}^n} \widehat{P}(\sigma) \mathcal{P}_\sigma,$$

where $\mathcal{P}_\sigma = \otimes_{i=1}^n \mathcal{P}_{\sigma_i}$ and $\mathcal{P}_0 = I, \mathcal{P}_1 = X, \mathcal{P}_2 = Y, \mathcal{P}_3 = Z$ are the single-qubit Pauli operators. The degree of a term $\widehat{P}(\sigma) \mathcal{P}_\sigma$ is the number of nontrivial Pauli's in it, denoted by $|\sigma|$. First, we adapt the smoothing technique in [46], which applies a depolarizing channel with small noise to P and removes the high-degree part of P , i.e. terms with $|\sigma| > d$ where d is a constant. After smoothing, S only contains degree- d operators

$$P^{(\text{Smooth})} = \sum_{\sigma: |\sigma| \leq d} \widehat{P^{(\text{Smooth})}}(\sigma) \mathcal{P}_\sigma,$$

so we denote the new strategy by $S^{(\text{Smooth})}$. Using the argument in [46], the probability of winning the game with this new strategy changes at most slightly, i.e.

$$\text{val}^*(G, S^{(\text{Smooth})}) \approx \text{val}^*(G, S).$$

Let τ be a small constant independent of n . Since the degree of $P^{(\text{Smooth})}$ is d , using a standard argument in the analysis of Boolean functions, the number of registers having influence that exceeds a given small τ is at most d/τ . Notice that d is independent of n , so is d/τ . Assume without loss of generality that $H = \{1, \dots, |H|\}$ is the set of all registers whose influence exceeds τ . We apply the invariance principle from [46] to replace all the non-identity Pauli bases in the registers with low influence by Gaussian variables while maintaining the strategy value. Let

$$\mathbf{P}^{(\text{Apprx})} = \sum_{\sigma: |\sigma| \leq d} \widehat{P^{(\text{Smooth})}}(\sigma) \mathcal{P}_{\sigma_1} \otimes \mathcal{P}_{\sigma_2} \otimes \dots \otimes \mathcal{P}_{\sigma_{|H|}} \otimes \mathbf{z}_{\sigma_{|H|+1}}^{(|H|+1)} \mathbb{1}_2 \otimes \mathbf{z}_{\sigma_{|H|+2}}^{(|H|+2)} \mathbb{1}_2 \otimes \dots \otimes \mathbf{z}_{\sigma_n}^{(n)} \mathbb{1}_2,$$

where $\mathbb{1}_2$ is a 2×2 identity matrix; $\{\mathbf{z}_j^{(i)}\}_{|H|+1 \leq i \leq n, 1 \leq j \leq 3}$ are independent Gaussian variables and $\mathbf{z}_0^{(|H|+1)} = \dots = \mathbf{z}_0^{(n)} = 1$. Denote the new strategy by $S^{(\text{Apprx})}$, then

$$\text{val}^*(G, S^{(\text{Apprx})}) \approx \text{val}^*(G, S^{(\text{Smooth})}).$$

Notice that this process significantly reduces the dimension of $\mathbf{P}^{(\text{Approx})}$ to a constant. To round such a randomized strategy back to a valid POVM strategy, we first need to reduce the number of Gaussian variables from $O(n)$ to a constant, which is the most difficult step. In this paper, we avoid the use of a crude union bound as in [46], by taking the distribution of the questions into account. Furthermore, we manage to ensure that the expectation of the distance from a random operator in the intermediate step to positive matrices after the dimension reduction step is independent of the question size. Then the inverse of the invariance principle allows us to round the randomized strategy back to a valid POVM strategy only acting on constantly many qubits. The improvements in the Gaussian dimension reduction step give us the improved bound.

This upper bound has already yielded an NEXP algorithm, where the certificate is an exponential-sized description of the strategy. To design a more efficient nondeterministic algorithm, we need to further compress the certificate to polynomial length. To compress the certificate, we first smoothen again the strategy by introducing additional noise as in the proof of the upper bound of D to remove all the high-degree terms. Such a transformation exponentially reduces the length of the certificate. The smoothed strategy only contains a polynomial number of coefficients since the maximal degree is a constant. Nonetheless, the smoothed strategy is only a *pseudo-strategy*, probably not a valid strategy because these smoothed operators may not form valid POVMs. The prover sends the description of a pseudo-strategy to the verifier, which is of polynomial length. The verifier performs a test on the given certificate to see if it is close to a valid strategy that gives a high winning probability with the following steps:

1. Check that the pseudo-POVM elements contained in the pseudo-strategy still sum up to the identity.
 2. Compute and check the winning probability of the pseudo-strategy.
 3. Check that all the operators in the pseudo-strategy are close to being positive semidefinite.
- Item 1 is straightforward. For item 2, notice that $\text{Tr}(\mathcal{P}_i \otimes \mathcal{P}_j) \psi = \delta_{i,j} c_i$, where $c_0 = 1$ and $c_1 = c_2 = c_3 = \rho$. Thus for any degree- d operators A, B , we have

$$\text{Tr}(A \otimes B) \psi^{\otimes D} = \sum_{\sigma: |\sigma| \leq d} \widehat{A}(\sigma) \widehat{B}(\sigma) c_\sigma, \quad (1)$$

where $c_\sigma = c_{\sigma_1} \cdots c_{\sigma_n}$. This computation can be done in polynomial time. The winning probability is simply a linear combination of a polynomial number of the terms in the form of Eq.(1), which, therefore, can also be computed in polynomial time. Item 3 is the most challenging. Notice that the dimension of each operator in the pseudo-strategy is still exponential. Thus, the verifier cannot directly compute its eigenvalues and check its positivity. Instead, we need an efficient positivity tester for large matrices.

The key component of our efficient positivity tester is a *derandomized invariance principle*, which enables us to further reduce the dimension of the operators to a constant and maintain the distance between the operator and the set of positive operators. To be more specific, let us define the real function ζ to be

$$\zeta(x) = \begin{cases} x^2 & \text{if } x \leq 0 \\ 0 & \text{otherwise} \end{cases}. \quad (2)$$

Then $\text{Tr} \zeta(P)$ is the distance from P to its positive part. As before, when the degree of an operator is bounded by a constant d , the number of quantum registers having influence that exceeds a given small constant τ is at most d/τ , which is also a constant. To further reduce the dimension of the operators, we prove a more general invariance principle for all

smooth functions compared with the one in [46]. It states that if all non-identity Pauli bases in the registers with low influence are substituted by Rademacher variables or Gaussian variables, the expectation of the distance to the set of positive semidefinite matrices is almost unchanged. We replace all such registers with Rademacher variables, which significantly reduces the dimension of a constant-degree operator to a constant, making it possible to compute its expected ζ function value efficiently. However, the invariance principle introduces $\text{poly}(s)$ -many random variables, where s is the size of the question sets. This only leads to a randomized positivity tester. To reduce the randomness, we further apply the well-known Meka-Zuckerman pseudorandom generator [36] to obtain a derandomized invariance principle, which only uses a logarithmic number of independent bits to simulate these variables². This gives a deterministic algorithm to approximately compute the expected ζ function values of all the measurement operators.

To prove the approximation problem is NP-hard, we can compile any $\text{MIP}[\log, O(1)]$ protocol for 3-SAT into a family of noisy nonlocal games one for each 3-SAT instance such that if a 3-SAT instance is satisfiable, the corresponding game has value 1 and if not, the value of the corresponding game is below some constant. In the compiled nonlocal game, the verifier checks with equal probability, if the provers can give consistent answers for the same question or if the provers can give valid answers for queries of their assignment of the instance. Using Fourier analysis, we show that when the provers share noisy MESs, winning the consistency checks with high probability implies that their strategy is essentially deterministic. Then we can relate the classical completeness and soundness of the MIP protocol to the values of the noisy nonlocal games.

1.1.2 Hardness of Noiseless $\text{MIP}^*[\text{poly}, O(1)]$

To show hardness of $\text{MIP}^*[\text{poly}, O(1)]$, we start from the known result $\text{MIP}^*[\text{poly}, \text{poly}] = \text{RE}$ [28], and apply an *answer reduction* transformation to the protocol to get answer length $O(1)$. Answer reduction is essentially PCP composition adapted to the MIP^* setting, and was already an essential component in [39] and [28]. Intuitively, the idea of answer reduction is to ask the two provers in an MIP^* protocol to compute a PCP proof that their answers satisfy the verifier’s predicate. The verifier will check this proof rather than checking the answers directly. In order to instantiate this, one requires a PCP of proximity that remains sound when implemented as a two-player quantum game. Showing this soundness condition is technically challenging and usually involves showing that the local tester for a locally testable code, when converted to a two-prover game, is sound against entangled provers. In [28], the code that was used was the Reed-Muller code, which has superconstant alphabet size, ultimately yielding poly-sized answers.

In order to obtain $O(1)$ -sized answers, we use the Hadamard code, which is a locally testable code over the binary alphabet. Fortunately for us, it is known that the local tester for this code is “quantum sound” [26, 38]. Moreover, the answer-reduction protocol in [39] is *modular*: it was shown in that work that *any* code with sufficiently good parameters and a quantum-sound tester can be combined with an off-the-shelf PCP of proximity to achieve answer reduction. Our main challenge is to show that the Hadamard code (or a slight variant of it) has a tester meeting the conditions of this theorem. Our new tester for the Hadamard code allows us to reduce the answer length from poly to $O(1)$ directly.

² An alternate approach is using Gaussian variables and derandomizing the Gaussian variables as in [30], which discretizes the Gaussian variables via the Box-Muller transformation and further derandomizes the discrete random variables.

1.2 Technical Contributions

1.2.1 Invariance Principle and Derandomized Invariance Principle for Matrix Functions

The invariance principle [37] is a generalization of the Berry-Esseen Theorem, which is a quantitative version of the Central Limit Theorem, to multilinear low-degree polynomials. Before illustrating the invariance principle, we need to introduce the notion of *influence*, a fundamental notion in the analysis of Boolean functions. Given a real function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and i.i.d. random variables $\mathbf{x}_1, \dots, \mathbf{x}_n$, the influence of i -th coordinate is

$$\text{Inf}_i(f) = \mathbb{E} \left[\left| f(\mathbf{x}) - f(\mathbf{x}^{(i)}) \right|^2 \right],$$

where $\mathbf{x}^{(i)}$ is obtained from \mathbf{x} by resampling the i -th variable. Hence, it captures the effect of the i -th variable on the function in average. Given a multilinear low-degree polynomial f in which all variables have low influence, the invariance principle states that the distributions of $f(X_1, \dots, X_n)$ and $f(Y_1, \dots, Y_n)$ are similar as long as the first and second moments of the random vectors (X_1, \dots, X_n) and (Y_1, \dots, Y_n) match, and the variables X_i, Y_i behave nicely³. The invariance principle is a versatile tool that allows us to connect the distribution of a function on complicated random variables to the distribution obtained by replacing these random variables with simpler ones, such as Gaussian variables or Rademacher random variables. The proof of the classical invariance principle in [37] is via Lindeberg's hybrid argument, which is also a classic method to prove the Central Limit Theorem.

In [46], Qin and Yao started investigating the invariance principle on matrix spaces. Suppose that P is a $m^n \times m^n$ matrix, viewed as an operator acting on n registers, each of dimension m . Let $\xi : \mathbb{R} \rightarrow \mathbb{R}$ be a smooth real function. Suppose all registers have low influence in P , where the influence is a generalization of the influence for functions. When substituting all registers with independent standard Gaussians or Rademacher variables multiplied by an identity matrix, we expect that the change of $\text{Tr} \xi(P)$ is small in expectation. The most challenging part of extending Lindeberg's argument to matrix functions is computing the high-order Fréchet derivatives, which are complicated and difficult to analyze in general [50]. Qin and Yao [46] established an invariance principle for a specific spectral function by directly computing the Fréchet derivatives and applying many complicated matrix-analytic techniques. Hence, the first obstacle we face is to prove an invariance principle for more general functions.

To overcome it, we adapt the theory of multilinear operator integrals [52], which provides a unified way to compute and bound the Fréchet derivatives. With such a tool, we establish an invariance principle applicable to a broader class of functions, including those that are smooth with a bounded third derivative and those that are Lipschitz continuous.

The invariance principle reduces the dimension from poly to constant but introduces a poly number of independent random variables. Thus, the second obstacle is that the size of the overall probability space is exponential. To improve the computational efficiency of our invariance principle, we use the ideas of [36, 22, 44] to use a Pseudorandom generator (PRG) to reduce the number of independent random variables. We apply this derandomized invariance principle to our positivity tester introduced below. Derandomized invariance principles build

³ To be more specific, $\mathbf{x}_i, \mathbf{y}_i$ need to be hypercontractive. Informally speaking, the p -norms $\|\mathbf{x}_i\|_p = \mathbb{E} [|\mathbf{x}_i|^p]^{1/p}$, $\|\mathbf{y}_i\|_p = \mathbb{E} [|\mathbf{y}_i|^p]^{1/p}$ do not increase drastically with respect to p . Many basic random variables, such as uniformly random variables, and Gaussian variables, are hypercontractive.

upon the crucial observation that the highest moment of variables involved in the proof is at most $2d$, where d is the degree of the operator, which is a constant. Thus, it suffices to use $4d$ -wise uniform random variables instead of polynomially many independent random variables when we replace the Pauli basis elements in the low-influence registers, which saves the randomness exponentially. To this end, we employ the well-known Meka-Zuckerman pseudorandom generator [36] to construct $4d$ -wise uniform random variables.

As the invariance principle has found numerous applications, we anticipate that the invariance principle for spectral functions is interesting in its own right. The positivity testing for low-degree matrices introduced below is an example of its applications.

1.2.2 Positivity Tester for Low-degree Matrices

A Hermitian matrix A is said to be positive semidefinite (PSD) if all the eigenvalues of A are non-negative. This testing problem has received increasing attention in the past couple of years [35, 21, 8, 41]. In this work, we present an efficient PSD tester for low-degree matrices, where the input matrix is given in terms of its Fourier coefficients. Given an $m^n \times m^n$ matrix, viewed as an operator acting on n -qudits, each of which has dimension m , if the degree of the operator is d , then the number of Fourier coefficients is bounded by $\sum_{i \leq d} \binom{n}{i} (m^2 - 1)^i = O(dn^d m^{2d})$. Hence, this allows for a compact description of a low-degree, exponential-dimension operator. If m, d are constants, the input is of size $\text{poly}(n)$, and we work in this setting when we explain how the tester works below.

Given the Fourier coefficients of a matrix P , our tester estimates the distance between P and the set of positive semidefinite matrices measured by $\text{Tr}\zeta(P)$, where $\zeta(\cdot)$ is defined in Eq. (2). Estimating $\text{Tr}\zeta(P)$ involves applying the derandomized invariance principle introduced above. More specifically, our tester enumerates all the possible seeds of the Meka-Zuckerman PRG to estimate this distance. For each seed, the computation time is $O(1)$ because the derandomized invariance principle has effectively reduced the dimension of P to a constant. Hence, our tester runs in time $\text{poly}(n)$, because there are only $\text{poly}(n)$ seeds. Its guarantees are summarized below.

► **Theorem (informal).** *Given as input the Fourier coefficients of a degree- d operator P acting on n qudits, each of dimension m , and error parameters $\beta \geq \delta \geq 0$, there exists an algorithm that runs in time $\exp(m^d/\delta) \cdot \text{poly}(n)$ such that*

- *the algorithm accepts if there exists a PSD operator Q such that $\|P - Q\|_F^2 < (\beta - \delta) m^n$;*
- *the algorithm rejects if $\|P - Q\|_F^2 > (\beta + \delta) m^n$ for any PSD operator Q .*

This approach completely differs from all previous works on positivity testing [41, 21, 8], where they only consider polynomial-sized matrices and the testers are randomized. In contrast, our tester is deterministic, and the dimension of the testing matrix can be exponential in input size if the degree is constant.

1.2.3 Answer Reduction with the Hadamard Code

As mentioned above, we obtain $O(1)$ -sized answers in the noiseless setting by applying the code-based answer reduction of [39], with the code chosen to be the Hadamard code. To implement this required two new technical components. First, we showed a *quantum-sound subset tester* for the Hadamard code: essentially, an interactive protocol that forces the provers to respond with the values of a subset F of the coordinates of a Hadamard codeword, where F is sampled from some (not necessarily uniform) distribution. Our proof of this result is essentially a generalization of the Fourier-analytic proof of the quantum soundness

of the BLR test [10, 38]. Secondly, the answer reduction procedure in [39] only works if the code has a relative distance close to 1 (i.e., distinct codewords differ on almost all locations), whereas the Hadamard code has a distance $1/2$. To overcome this, we slightly modified the answer-reduced verifier’s protocol of [39] by querying a large constant number of “dummy coordinates” from the provers. It is worth mentioning that the answer reduction procedure from [39] is different from the procedure used in [28]; the former works for any error-correcting code satisfying certain properties but does not yield protocols that can be recursively compressed, whereas the latter is specialized to the low-degree code but is compatible with recursive compression. In the end, we are in effect using both versions of answer reduction: the [28] version inside the recursive compression to obtain a protocol for RE, and then one layer of the [39] version to bring the answer size successively down from polynomial to constant, using the Hadamard code.

We remark that it might be possible to achieve constant answer size by repeatedly applying the answer reduction technique of [28], but we decide to proceed with the current approach for a one-shot solution, which is easier to analyze and gives better soundness.

1.3 Discussions and Open Problems

Our result characterizes the effect of noise on the computational complexity class MIP^* . To our knowledge, this is the first example of a quantum computational complexity class whose quantum advantage over its classical counterpart completely vanishes in the presence of noise. For comparison, noise causes *no* collapse in the BQP model, or in general, for BQTIME because the algorithms in these classes can be implemented fault-tolerantly. Even for algorithms with bounded space, it seems that the same reasoning still applies because all the intermediate measurements to achieve fault tolerance can be eliminated without a large space overhead [18]. Hence, our work raises the natural question of which quantum complexity classes are truly fault tolerant. In contrast, for complexity classes like MIP^* , the fault-tolerance theorem [1] cannot be applied as the model of computation disallows the operations needed to perform error correction. For the specific case of MIP^* , our result further shows that no form of fault tolerance is possible.

More broadly, we know other examples where constant noise destroys the quantum advantage. Random circuit sampling has been proposed to demonstrate the quantum advantage offered by near-term quantum devices [12]. However, when the random circuits are subject to constant noise, this sampling task becomes classically easy [2]. We have more of such examples in quantum query algorithms. For example, if the oracle is noisy or faulty, no quantum algorithm can achieve any speed-up in the unstructured search problem [48]. In a setting closer to the near-term devices, where each gate in the circuit is subject to independent noise but the oracle is perfect, the authors of [14] showed that no quantum algorithm could achieve any speed-up in the unstructured search problem either. For a more detailed survey about the effect of noise on quantum query algorithms, we refer to [14, Section 3].

Our result also raises some natural but intriguing questions. We list some of them below.

1. For MIP^* protocols with more rounds of interactions and larger answer sets, it is unclear how big the effect of noise is. Hence, we ask: Does the vanishing phenomenon for computational advantages occur for general MIP^* protocols?
2. What is the computational power of MIP^* with unbounded copies of a pure (noiseless) non-EPR state? Will $\text{MIP}^* = \text{RE}$ still hold for any noiseless non-EPR state? The MIP^* protocol for RE of [28] requires EPR states for the provers to succeed, and in general, it is known that any protocol which is symmetric and synchronous requires the provers

to use an MES [56, 45]. Moreover, the question reduction technique of [39, 28] first certifies that the provers share many copies of the EPR state, on which they sample their own questions. Hence, to accommodate any non-EPR state, we need to redesign MIP* protocols.

3. What non-computational capabilities of the MIP* model remain in the noisy setting? Specifically, it is known that nonlocal games and correlations can be used to self-test entangled states. In the noisy setting, can we certify any properties of the provers' shared entanglement? Previous work on this question has studied entanglement of formation [5] and one-shot distillable entanglement [3], but the general picture remains unclear.
4. Invariance principle has found applications in designing various pseudorandom generators and counting algorithms [22, 44, 43, 6, 31]. Will our invariance principle lead to new pseudorandom generators?
5. Testing whether a matrix is positive has played an important role in the study of algorithm designs for linear algebra problems, community structure detection, differential equations, etc (see [8] and references therein). Multiple studies have been devoted to designing efficient algorithms for positivity testing [41, 21, 8]. Will our algorithm of positivity testing find new applications?

2 Nonlocal Games and MIP* Protocols

In this paper we use the standard notations for matrix spaces, random variables etc. For a detailed description see Appendix A. Two-player one-round MIP* protocols are also nonlocal games. We follow the notations of [28] for nonlocal games.

► **Definition 4** (Two-player one-round games). *A two-player one-round game G is specified by a tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, \mu, V)$ where*

- \mathcal{X} and \mathcal{Y} are finite sets, called the question sets,
- \mathcal{A} and \mathcal{B} are finite sets, called the answer sets,
- μ is a probability distribution over $\mathcal{X} \times \mathcal{Y}$, called the question distribution, and
- $V : \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B} \rightarrow \{0, 1\}$ is a function, called the decision predicate.

► **Definition 5** (Tensor-product strategies). *A tensor-product strategy S of a nonlocal game $G = (\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, \mu, V)$ is a tuple (ψ, A, B) where*

- a bipartite quantum state $\psi \in \mathcal{H}_A \otimes \mathcal{H}_B$ for finite dimensional complex Hilbert spaces \mathcal{H}_A and \mathcal{H}_B ,
- A is a set $\{A^x\}$ such that for every $x \in \mathcal{X}$, $A^x = \{A_a^x \mid a \in \mathcal{A}\}$ is a POVM over \mathcal{H}_A , and
- B is a set $\{B^y\}$ such that for every $y \in \mathcal{Y}$, $B^y = \{B_b^y \mid b \in \mathcal{B}\}$ is a POVM over \mathcal{H}_B .

► **Definition 6** (Tensor product value). *The tensor product value of a tensor product strategy $S = (\psi, A, B)$ for a nonlocal game $G = (\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, \mu, V)$ is defined as*

$$\text{val}^*(G, S) = \sum_{x, y, a, b} \mu(x, y) V(x, y, a, b) \text{Tr}(A_a^x \otimes B_b^y) \psi.$$

For $v \in [0, 1]$ we say that the strategy passes or wins G with probability v if $\text{val}^*(G, S) \geq v$. The quantum value or tensor product value of G is defined as

$$\text{val}^*(G) = \sup_S \text{val}^*(G, S)$$

where the supremum is taken over all tensor product strategies S for G .

30:12 The Computational Advantage of MIP* Vanishes in the Presence of Noise

When we prove the quantum soundness of an MIP* protocol, we focus on projective strategies, where the measurements A^x and B^y are all projective, following Naimark's Dilation theorem [29, Theorem 5.1].

► **Definition 7.** A game $G = (\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, \mu, V)$ is symmetric if $\mathcal{X} = \mathcal{Y}$ and $\mathcal{A} = \mathcal{B}$, the distribution μ is symmetric (i.e. $\mu(x, y) = \mu(y, x)$ for all x and y), and the predicate V treats both players symmetrically (i.e. $V(x, y, a, b) = V(y, x, b, a)$ for all x, y, a, b).

We call a strategy $S = (|\psi\rangle, A, B)$ symmetric if $|\psi\rangle$ is a pure state in $\mathcal{H} \otimes \mathcal{H}$, for some Hilbert space \mathcal{H} , that is invariant under permutation of the two factors, and the measurement operators of both players are identical.

A symmetric game is denoted by $(\mathcal{X}, \mathcal{A}, \mu, V)$, and a symmetric strategy is denoted by $(|\psi\rangle, M)$ where M denotes the set of measurement operators for both players.

► **Lemma 8** (Lemma 5.7 in [28]). Let $G = (\mathcal{X}, \mathcal{A}, \mu, V)$ be a symmetric game with value $1 - \varepsilon$ for some $\varepsilon \geq 0$. Then there exists a symmetric and projective strategy $S = (|\psi\rangle, M)$ such that the $\text{val}^*(G, S) \geq 1 - \varepsilon$.

Hence, for symmetric nonlocal games, it suffices to only consider symmetric strategies.

3 Invariance Principle for Matrix Spaces

This section will present an invariance principle for general functions on matrix spaces. Hypercontractivity is crucial in the proofs of all previous invariance principles [37]. We also need to establish a new hypercontractive inequality before proving the invariance principle. The proofs of the results in this section can be found in Appendix B.1.

3.1 Hypercontractivity

In this subsection, we adopt the concept of orthonormal ensembles as introduced in [37].

► **Definition 9.** Given $m, n \in \mathbb{Z}_{>0}$, a collection of n real random variables $\{\mathbf{z}_1, \dots, \mathbf{z}_n\}$ are orthonormal if $\mathbb{E}[\mathbf{z}_i \mathbf{z}_j] = \delta_{i,j}$. We call a collection of m orthonormal real random variables, the first of which is constant 1, an m -orthonormal ensemble. We call \mathbf{x} an (m, n) ensemble if $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$, where for all $i \in [n]$, $\mathbf{x}_i = \{\mathbf{x}_{i,0} = 1, \mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,m-1}\}$ is an m -orthonormal ensemble.

► **Definition 10.** Given $m, n \in \mathbb{Z}_{>0}$, $\tau \in [m]_{\geq 0}^n$ and an (m, n) ensemble \mathbf{x} , denote $\mathbf{x}_\tau = \prod_{i=1}^n \mathbf{x}_{i,\tau_i}$. Define a multilinear polynomial over \mathbf{x} to be $Q(\mathbf{x}) = \sum_{\tau \in [m]_{\geq 0}^n} \widehat{Q}(\tau) \mathbf{x}_\tau$, where the $\widehat{Q}(\tau)$'s are real constants.

For $\gamma \in [0, 1]$, we define the operator T_γ acting on multilinear polynomial $Q(\mathbf{x})$ by

$$T_\gamma Q(\mathbf{x}) = \sum_{\tau \in [m]_{\geq 0}^n} \gamma^{|\tau|} \widehat{Q}(\tau) \mathbf{x}_\tau.$$

► **Definition 11.** For $1 \leq r < \infty$, let \mathbf{y} be a random variable with $\mathbb{E}[|\mathbf{y}|^r] < \infty$. Define $\|\mathbf{y}\|_r = (\mathbb{E}[|\mathbf{y}|^r])^{1/r}$. Given $1 \leq p \leq q < \infty$, $0 < \eta < 1$, $m, n \in \mathbb{Z}_{>0}$ and an (m, n) ensemble \mathbf{x} , we say that \mathbf{x} is (p, q, η) -hypercontractive if for any multilinear polynomial Q , it holds that $\|(T_\eta Q)(\mathbf{x})\|_q \leq \|Q(\mathbf{x})\|_p$.

Consider an (m, n) ensemble \mathbf{x} . If for all $i \in [n]$, $j \in [m-1]$, $\mathbf{x}_{i,j}$ are either independent standard Gaussians or independent Rademacher variables, then \mathbf{x} is $(2, q, (q-1)^{-1/2})$ -hypercontractive. These two types are represented as significant examples of hypercontractive ensembles. Readers can refer to [37] for an extensive treatment on hypercontractive ensembles.

We then introduce the noise operator Γ_γ for random matrices, which is a hybrid of T_γ in Definition 10 and Δ_γ in Definition 47.

► **Definition 12.** Given $0 \leq \gamma \leq 1$, $h, n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, an (m^2, n) ensemble \mathbf{x} , and a random matrix $P(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{x}) \mathcal{B}_\sigma$, where $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$ is a standard orthonormal basis and p_σ is a real multilinear polynomial for all $\sigma \in [m^2]_{\geq 0}^h$, the noise operator Γ_γ is defined to be

$$\Gamma_\gamma(P(\mathbf{x})) = \sum_{\sigma \in [m^2]_{\geq 0}^h} (T_\gamma p_\sigma)(\mathbf{x}) \Delta_\gamma(\mathcal{B}_\sigma).$$

The main result in this subsection is stated below.

► **Theorem 13** (Hypercontractivity for random matrices). Given $h, n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, $0 < \eta < 1$, $0 \leq \gamma \leq \min\{\eta, (9m)^{-1/4}\}$, a $(2, 4, \eta)$ -hypercontractive (m^2, n) ensemble \mathbf{x} and a random matrix $P(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{x}) \mathcal{B}_\sigma$, where $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$ is a standard orthonormal basis, and p_σ is a real multilinear polynomial for all $\sigma \in [m^2]_{\geq 0}^h$, it holds that

$$\mathbb{E}_{\mathbf{x}} \left[\|\Gamma_\gamma(P(\mathbf{x}))\|_4^4 \right] \leq \left(\mathbb{E}_{\mathbf{x}} \left[\|P(\mathbf{x})\|_2^2 \right] \right)^2,$$

where Γ_γ is defined in Definition 12.

The following is an application of Theorem 13.

► **Theorem 14.** Given $h, n, m, d \in \mathbb{Z}_{>0}$, $m \geq 2$, $0 < \eta < 1$, a $(2, 4, \eta)$ -hypercontractive (m^2, n) ensemble \mathbf{x} , and a random matrix $P(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{x}) \mathcal{B}_\sigma$, where $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$ is a standard orthonormal basis and for all $\sigma \in [m^2]_{\geq 0}^h$ and p_σ is a real multilinear polynomial satisfying $\deg(p_\sigma) + |\sigma| \leq d$, it holds that

$$\mathbb{E} \left[\|P(\mathbf{x})\|_4^4 \right] \leq \max\{9m, 1/\eta^4\}^d \left(\mathbb{E} \left[\|P(\mathbf{x})\|_2^2 \right] \right)^2.$$

3.2 Invariance Principle

We are now prepared to introduce an invariance principle on matrix space applicable to general functions. Initially, we establish the proof for functions in \mathcal{C}^4 .

► **Theorem 15.** Given $0 < \tau, \eta < 1$, $d, h, m, n \in \mathbb{Z}_{>0}$, $H \subseteq [n]$ of size $|H| = h$, $\xi \in \mathcal{C}^3$ satisfying $\|\xi^{(3)}\|_\infty \leq B$ where B is a constant, and a $(2, 4, \eta)$ -hypercontractive (m^2, n) ensemble \mathbf{x} , let $P \in \mathcal{H}_m^{\otimes n}$ be a degree- d operator satisfying $\text{Inf}_i(P) \leq \tau$ for all $i \notin H$. Suppose that P has a Fourier expansion $P = \sum_{\sigma \in [m^2]_{\geq 0}^n} \hat{P}(\sigma) \mathcal{B}_\sigma$. Let $P^H(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^n} \hat{P}(\sigma) \mathbf{x}_{\sigma_H} \mathcal{B}_{\sigma_H}$. If $\sum_{\sigma \neq 0} \hat{P}(\sigma)^2 \leq 1$, we have

$$\left| m^{-n} \text{Tr} \xi(P) - m^{-h} \mathbb{E} \left[\text{Tr} \xi(P^H(\mathbf{x})) \right] \right| \leq CB \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d$$

for some absolute constant C .

30:14 The Computational Advantage of MIP* Vanishes in the Presence of Noise

For those functions that are not sufficiently smooth, if they have a mollifier, which is a smooth approximator with a bounded third derivative, then the invariance principle still holds. The following lemma proves an invariance principle for $\zeta(\cdot)$ defined in Appendix A.2.4, which has a mollifier $\zeta_\lambda(\cdot)$ guaranteed by Fact 57.

► **Lemma 16.** *Given $0 < \tau, \eta < 1$, $d, h, m, n \in \mathbb{Z}_{>0}$, $H \subseteq [n]$ of size $|H| = h$, a $(2, 4, \eta)$ -hypercontractive (m^2, n) ensemble \mathbf{x} and a degree- d $P \in \mathcal{H}_m^{\otimes n}$ satisfying $\text{Inf}_i(P) \leq \tau$ for all $i \notin H$. Suppose that P has a Fourier expansion $P = \sum_{\sigma \in [m^2]_{\geq 0}^n} \hat{P}(\sigma) \mathcal{B}_\sigma$. Let $P^H(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^n} \hat{P}(\sigma) \mathbf{x}_{\sigma_H} \mathcal{B}_{\sigma_H}$. If $\sum_{\sigma \neq 0} \hat{P}(\sigma)^2 \leq 1$, we have*

$$\left| m^{-n} \text{Tr} \zeta(P) - m^{-h} \mathbb{E}[\text{Tr} \zeta(P^H(\mathbf{x}))] \right| \leq 3 \left(C B_3 \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d \right)^{2/3}$$

for some universal constants C and B_3 .

► **Remark 17.** It is possible to prove an invariance principle for a broader class of functions. For example, we can prove it for Lipschitz continuous functions using the argument in [24, Lemma 3.5]. However, it is out of the focus of this paper. We will leave it for further research.

3.3 Derandomized Invariance Principle

From Theorem 15, it is not hard to see that the non-identity basis elements can be substituted by independent Rademacher variables. In this section, we will replace those Rademacher variables with pseudorandom variables to save the randomness. It is worth noting that there is a large body of research on derandomization through invariance principles (readers may refer to [44] and the references therein). We adopt the pseudorandom generator (PRG) introduced in [36]. The PRG is constructed by pairwise uniform hash functions as follows.

For $\mathcal{F} = \{f : [n] \rightarrow [p]\}$, define $G : \mathcal{F} \times (\{-1, 1\}^n)^p \rightarrow \{-1, 1\}^n$ by

$$G(f, z^1, \dots, z^p) = x, \text{ where } x_i = z_i^{f(i)} \text{ for } i \in [n]. \quad (3)$$

We define the influence of a random variable in a random matrix using the notation $\text{VarInf}(\cdot)$ to distinguish from the notation for the influence of a register in Definition 38.

► **Definition 18.** *Given $n, p \in \mathbb{Z}_{>0}$, let $P(\mathbf{b}) = \sum_{S \subseteq [n]} \mathbf{b}_S P_S$ be a random matrix with \mathbf{b} being drawn uniformly from $\{\pm 1\}^n$ and $\mathbf{b}_S = \prod_{i \in S} \mathbf{b}_i$. Then the influence of i 'th coordinate of \mathbf{b} is defined to be*

$$\text{VarInf}_i(P(\mathbf{b})) = \sum_{S \ni i} \|P_S\|_2^2.$$

We also define the influence of a block of coordinates. Let $j \in [p]$ and $f : [n] \rightarrow [p]$ be a function, define the influence on the block $f^{-1}(j) \subseteq [n]$ to be

$$\text{VarInf}_{f,j}(P(\mathbf{b})) = \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \|P_S\|_2^2.$$

The following is the main theorem in this section.

► **Theorem 19** (Derandomized invariance principle for ζ). *Given $d, h, m, n \in \mathbb{Z}_{>0}$, $m > 1$, and a random matrix $P(\mathbf{b}) = \sum_{S \subseteq [n]} \mathbf{b}_S P_S$, where $\mathbf{b} \sim_{\mathbf{u}} \{-1, 1\}^n$, $\mathbb{E}_{\mathbf{b}} \left[\|P(\mathbf{b})\|_2^2 \right] \leq 1$, $\mathbf{b}_S = \prod_{i \in S} \mathbf{b}_i$ and $P_S \in \mathcal{H}_m^{\otimes h}$, they satisfy $|S| + \deg(P_S) \leq d$ and $\text{VarInf}_i(P(\mathbf{b})) \leq \tau$ for all $i \in [n]$.*

Let p be the smallest power of 2 satisfying $p \geq d/\tau$; $\mathcal{F} = \{f : [n] \rightarrow [p]\}$ be a family of pairwise uniform hash functions. For any $i \in [p]$, define \mathbf{z}^i to be a $4d$ -wise uniform random vector drawn from $\{\pm 1\}^n$, and \mathbf{z}^i are independent across $i \in [p]$. Given $f \in \mathcal{F}$, denote $\mathbf{x}_f = G(f, \mathbf{z}^1, \dots, \mathbf{z}^p)$ as in Equation (3). Then it holds that

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}} [\text{Tr } \zeta(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_f} [\text{Tr } \zeta(\mathbf{P}(\mathbf{x}_f))] \right| \leq C \sqrt{(9m)^d d \tau},$$

where \mathbf{f} is drawn uniformly from \mathcal{F} and C is a universal constant.

We first prove a derandomized invariance principle for the functions with bounded fourth derivative.

► **Theorem 20** (Derandomized invariance principle). *Given $d, h, m, n \in \mathbb{Z}_{>0}$, $m > 1$, and a random matrix $P(\mathbf{b}) = \sum_{S \subseteq [n]} \mathbf{b}_S P_S$, where $\mathbf{b} \sim_{\mathbf{u}} \{-1, 1\}^n$, $\mathbb{E}_{\mathbf{b}} [\|P(\mathbf{b})\|_2^2] \leq 1$, $\mathbf{b}_S = \prod_{i \in S} \mathbf{b}_i$ and $P_S \in \mathcal{H}_m^{\otimes h}$, they satisfy that $|S| + \deg(P_S) \leq d$ and $\text{VarInf}_i(P(\mathbf{b})) \leq \tau$ for all $i \in [n]$.*

Let p be the smallest power of 2 satisfying $p \geq d/\tau$; $\mathcal{F} = \{f : [n] \rightarrow [p]\}$ be a family of pairwise uniform hash functions. For any $i \in [p]$, define \mathbf{z}^i to be a $4d$ -wise uniform random vector drawn from $\{\pm 1\}^n$, and \mathbf{z}^i are independent across $i \in [p]$. Given $f \in \mathcal{F}$, denote $\mathbf{x}_f = G(f, \mathbf{z}^1, \dots, \mathbf{z}^p)$ as in Equation (3). Then for any $\xi \in \mathcal{C}^4$ with $\|\xi^{(4)}\|_{\infty} \leq C_0$ where C_0 is a constant, it holds that

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}} [\text{Tr } \xi(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_f} [\text{Tr } \xi(\mathbf{P}(\mathbf{x}_f))] \right| \leq 4C_1 C_0 (9m)^d d \tau,$$

where \mathbf{f} is drawn uniformly from \mathcal{F} and C_1 is a universal constant.

► **Remark 21.** It is also possible to generalize Theorem 19 to Lipschitz continuous functions using the argument in [24, Lemma 3.5].

Assuming Theorem 20, Theorem 19 is straightforward.

4 Positivity Tester for Low Degree Operators

In this section, we will present an algorithm deciding whether a low-degree operator is $(\beta - \delta)$ -close to a positive semidefinite matrix or $(\beta + \delta)$ -far from all positive semidefinite matrices, for error parameters $\beta > \delta > 0$. The input operator is given in the form of a Fourier expansion. The algorithm and the proofs can be found in Appendix B.2.

► **Definition 22** (Positivity testing problem). *Given $d, D, m \in \mathbb{Z}_{>0}$, $m > 1$, and real numbers $\beta > \delta > 0$, the input is a degree- d operator in $\mathcal{H}_m^{\otimes D}$ given in the form of Fourier expansion*

$$P = \sum_{\substack{\sigma \in [m^2]_{\geq 0}^D \\ \sigma: |\sigma| \leq d}} \widehat{P}(\sigma) \mathcal{B}_{\sigma}.$$

Distinguish the following two cases.

- Yes: if $m^{-D} \text{Tr } \zeta(P) < \beta - \delta$.
- No: if $m^{-D} \text{Tr } \zeta(P) > \beta + \delta$.

Notice that the number of Fourier coefficients is $\sum_{i=0}^d \binom{D}{i} (m^2 - 1)^i$. If we are concerned with constant-degree operators, then the dimension of the operator is exponential in the input size.

30:16 The Computational Advantage of MIP* Vanishes in the Presence of Noise

► **Theorem 23.** *Given $d, D, m \in \mathbb{Z}_{>0}$, $m > 1$, and real numbers $\beta > \delta > 0$, there exists a deterministic algorithm for the positivity testing problem that runs in time*

$$\exp(\text{poly}(m^d, 1/\delta)) \cdot D^{O(d)}.$$

In particular, if m, d, δ are constants, then the algorithm runs in time $\text{poly}(D)$.

The algorithm applies the invariance principle Lemma 16 to reduce the dimension of the matrices and then Theorem 19 to derandomize, while the distance to positive operators is approximately preserved.

5 Noisy Nonlocal Games are NP-complete

► **Definition 24** (Noisy Nonlocal Game Value Problem). *The input consists of the description of a nonlocal game, which is a tuple $\mathfrak{G} = (\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, \mu, V)$, and real values ρ, β and ε . \mathcal{X} and \mathcal{Y} are question sets and assume $|\mathcal{X}| = |\mathcal{Y}| = s$. \mathcal{A} and \mathcal{B} are answer sets and assume $|\mathcal{A}| = |\mathcal{B}| = t$. Let μ be a distribution on $\mathcal{X} \times \mathcal{Y}$ and $V : \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B} \rightarrow \{0, 1\}$ be the predicate.*

Let $v = \text{val}^(\mathfrak{G}, \psi_{AB})$ be the value of the nonlocal game, where Alice and Bob share arbitrarily many copies of a noisy MES ψ_{AB} with the maximal correlation ρ . Let $1 > \beta > \varepsilon > 0$. The task is to distinguish the following two cases.*

- *Yes: $v > \beta + \varepsilon$.*
- *No: $v < \beta - \varepsilon$.*

In this section, we show:

► **Theorem 25.** *The noisy nonlocal game value problem is NP-complete.*

It follows from the two propositions below, whose proofs can be found in Appendix B.3.

► **Proposition 26.** *There exists a nondeterministic algorithm that runs in time*

$$\text{poly}\left(s, \text{eexp}\left(t, \log\left(\frac{1}{\rho}\right), \frac{1}{\varepsilon}\right)\right)$$

that solves the noisy nonlocal game value problem. Here $\text{eexp}(\cdot)$ means doubly exponential. In particular, if t, ρ, ε are constants, then the problem is in NP.

► **Proposition 27.** *For each 3-SAT instance ϕ , there is a nonlocal game $G(\phi)$ such that its noisy game value is 1 if ϕ is satisfiable, and below some constant c if ϕ is not satisfiable.*

5.1 The Nondeterministic Algorithm

We first present an upper bound on the number of noisy MES sufficient to approximate the value of a nonlocal game to an arbitrary precision. The upper bound from [46] is $D = \exp(\text{poly}(s), \exp(\text{poly}(t)))$. The follow-up work [47] studied fully quantum games in which both questions and answers are quantum and proved a better upper bound $D = \exp(\text{poly}(s), \text{poly}(t))$ using a refined Gaussian dimension reduction. We observe that this upper bound can be further improved to $D = \text{poly}(s, \exp(\text{poly}(t)))$ for nonlocal games.

► **Theorem 28.** *Given parameters $0 < \epsilon, \rho < 1$, $n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, a noisy MES state ψ_{AB} , i.e., $\psi_A = \psi_B = \frac{1}{m}$ with the maximal correlation $\rho = \rho(\psi_{AB}) < 1$ as defined in Definition 41, let \mathfrak{G} be a nonlocal game with the question sets \mathcal{X}, \mathcal{Y} and the answer sets \mathcal{A}, \mathcal{B} . Suppose the players share arbitrarily many copies of ψ_{AB} . Let $\omega_n(\mathfrak{G}, \psi_{AB})$ be the*

highest winning probability that the players can achieve when sharing n copies of ψ_{AB} . Then there exists an explicitly computable bound $D = D(|\mathcal{X}|, |\mathcal{Y}|, |\mathcal{A}|, |\mathcal{B}|, m, \epsilon, \rho)$, such that for any $n > D$, $\omega_n(\mathfrak{G}, \psi_{AB}) - \omega_D(\mathfrak{G}, \psi_{AB}) \leq \epsilon$. In particular, one may choose

$$D = \text{poly} \left(|\mathcal{X}|, |\mathcal{Y}|, \exp \left(\text{poly} \left(|\mathcal{A}|, |\mathcal{B}|, \frac{1}{\epsilon}, \frac{1}{1-\rho} \right), \log m \right) \right).$$

The proof largely follows the framework in [46] with several refinements.⁴

Next we present the algorithm, which is deterministic provided with a certificate. By Theorem 28 we know that sharing D copies of ψ_{AB} is sufficient to approximate the game value. However, outlining a strategy that shares D copies of ψ_{AB} requires $\exp(D)$ bits, rendering it excessively costly. Despite this, we've devised a more affordable certificate. Interpreted as a degree- d pseudo-strategy, this certificate is presented through its Fourier coefficients. By pseudo-strategy we mean two sets of operators $\{P_a^x\}$ and $\{Q_b^y\}$ that may not be a valid quantum strategy. However, we can still define the winning probability on a pseudo-strategy, mathematically. The algorithm is given in Appendix B.3.

5.2 NP-Hardness

In this subsection, we first show that if $L \in \text{MIP}$ then $L \in \text{noisy MIP}^*$. Then Proposition 27 directly follows from the fact that 3-SAT $\in \text{MIP}[\log, 1]$ [7].

► **Proposition 29.** *Let $V = (\text{Alg}_Q, \text{Alg}_V)$ be an MIP protocol for a language L with perfect completeness. Then there exists a verifier V^* that is a noisy MIP* verifier for L with the following conditions:*

Completeness. *If $\text{input} \in L$, there is a value-1 strategy for V^* .*

Soundness. *Given input , if there is a strategy for V^* with value $1 - \epsilon$, then there is a strategy for V with value $1 - 2\epsilon - \frac{16\epsilon}{1-\rho}$.*

6 MIP* Protocol for RE with $O(1)$ -size Answers

In this section, we prove that there is an MIP* protocol for any language in RE with poly-size questions and constant-size answers. The key step is to develop a new answer reduction technique that can reduce the answer size of an MIP* protocol from $O(\log n)$ to $O(1)$ while maintaining other parameters of the protocol. We achieve it by modifying the answer reduction technique from [39]. Natarajan and Wright's answer reduction follows a modular design with two major components: Probabilistically checkable proofs of proximity (PCPP) and a tester of the low-degree code. Hence, to achieve constant answer size, it suffices to change the code to the Hadamard code, and derive a new tester for the Hadamard code that allows a verifier to test multiple bits of a codeword at the same time. Then in our final construction of the MIP* protocol for RE, we apply our new answer reduction with the Hadamard code to the MIP* protocol for RE from [28]. The proofs of the results of this section can be found in Appendix B.4.

⁴ One may wonder why the upper bound in [47] is still exponential in the size of the question set with the refined Gaussian dimension reduction. This is because of the different treatment of the questions. When the questions are classical, we take into account the distribution of the questions. However, if the questions are quantum as considered in [47], the question registers are expressed as a linear combination of matrix basis elements, where an extra factor on the size of the question sets is introduced.

Note that [28] doesn't use the answer reduction technique of [39]. The authors of [28] use a specific PCPP tailored to the low individual-degree code in their answer reduction technique so that it fits the recursive compression framework. However, the answer reduction technique of [28] is more difficult to modify due to its less modular design.

6.1 Subset Tester for the Hadamard Code

To use the [39] answer reduction procedure with a particular error-correcting code, one must show that this code satisfies certain efficient testability properties. Here we show this for the Hadamard code. Specifically, we show that the Hadamard code has a *subset tester* in the sense of [39, Section 16], which ensures that the provers have a global Hadamard encoding of some bitstring.

First, we recall the definition and key properties of the Hadamard code.

- **Definition 30.** *The Hadamard code encodes $x \in \mathbb{F}_2^k$ as $\text{Enc}_k(x) = (x \cdot y)_{y \in \mathbb{F}_2^k}$. Moreover,*
 - *For $x \neq y \in \mathbb{F}_2^k$, $\text{Enc}_k(x)$ and $\text{Enc}_k(y)$ have normalized Hamming agreement at most $\eta_H = \frac{1}{2}$.*
 - *There exists an embedding $\mu_k : [k] \rightarrow [2^k]$ such that for each $i \in [k]$, $\mu_k(i) = 2^{i-1}$ and $x_i = (\text{Enc}(x))_{\mu_k(i)}$.*
 - *There exists a decoding algorithm Dec_k such that $\text{Dec}_k(\text{Enc}_k(x)) = x$ and, for every w not in the range of Enc_k , $\text{Dec}_k(w) = \perp$.*

The decoding algorithm Dec_k on input w , first computes $x = (w_{\mu_k(k)}, \dots, w_{\mu_k(1)})$ outputs x if $w = \text{Enc}_k(x)$ and \perp otherwise. Note that both Enc_k and Dec_k run in time exponential in k .

- **Proposition 31.** *For the subset $F = \{x_1, \dots, x_k\} \subseteq \mathbb{F}_2^n$ sampled according to a distribution D and a uniformly random $y \in \mathbb{F}_2^n$, if a quantum strategy with $|\psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$ and measurements*

$$\left\{ M_{a,c,a'}^{F,y} \mid a, a' \in \mathbb{F}_2^k, c \in \mathbb{F}_2 \right\}, \left\{ N_b^F \mid b \in \mathbb{F}_2^k \right\}, \left\{ N_d^y \mid d \in \mathbb{F}_2 \right\}$$

can pass the subset tester with probability $1 - \varepsilon$, then there is a Hilbert space $\mathcal{H}'_A \otimes \mathcal{H}'_B$, a state $|aux\rangle = |aux_A\rangle \otimes |aux_B\rangle \in \mathcal{H}'_A \otimes \mathcal{H}'_B$ and a projective measurement $\{\hat{G}_u \mid u \in \mathbb{F}_2^n\}$ on $\mathcal{H}_B \otimes \mathcal{H}'_B$ such that if we write $|\psi'\rangle = |\psi\rangle \otimes |aux\rangle$

$$\mathbb{E}_{F \sim D} \sum_{a \in \mathbb{F}_2^k} \|N_a^F \otimes \mathbb{1}_{\mathcal{H}'} \otimes \mathbb{1}_B |\psi'\rangle - \mathbb{1}_A \otimes \sum_{\substack{u: u_{x_i} = a_i \\ \forall i \in [k]}} \hat{G}_u |\psi'\rangle\|^2 \leq (2k-1)^2 (45 + 12\sqrt{k}) \sqrt{\varepsilon}.$$

6.2 Answer Reduction Protocol

The subset tester of the Hadamard code implies that we can replace the low-degree code of the answer reduction technique in [39, Section 17.4] by the Hadamard code. The other key ingredient of Natarajan and Wright's answer reduction is probabilistically checkable proofs of proximity, so we recall its definition and key properties that we will use later.

- **Definition 32** (Probabilistically checkable proofs of proximity (PCPP)). *For functions $r, q : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$, $t : \mathbb{Z}^+ \times \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$, and constants $s, \gamma \in [0, 1]$, a pair language $L \subseteq \{0, 1\}^* \times \{0, 1\}^*$ is in $\text{PCPP}_{s,\gamma}[r, q, t]$ if there exists an (r, q, t) -restricted PCPP verifier V with the following properties:*

Completeness: If $(x, y) \in L$, there exists a proof π such that $\Pr_R[V^{y,\pi}(x, |y|; R) = 1] = 1$ where $V^{y,\pi}(x, |y|; R)$ denotes the decision V on input $(x, |y|)$, oracle access to (y, π) with $q(|x|)$ queries, and randomness R from $r(|x|)$ coin tosses.

Soundness: Let $L_x = \{y \mid (x, y) \in L\}$. If (x, y) is such that y is γ -far from $L_x \cap \{0, 1\}^{|y|}$, then for every π , $\Pr_R[V^{y,\pi}(x, |y|; R) = 1] \leq s$.

We work with the PCPP such that when L is an $\text{NTIME}(T)$ pair language,

Randomness complexity: $r(m) = \log_2 T(m) + O(\log_2 \log_2 T(m))$,

Query complexity: $q(m) = O(1)$, and

Verification time: $t(m, K) = \text{poly}(m, \log_2 K, \log_2 T(m + K))$.

We are going to apply the PCPP defined above to the following language.

► **Definition 33.** Let $V = (\text{Alg}_Q, \text{Alg}_A)$ be an MIP^* verifier, where Alg_Q is his algorithm to sample the questions and Alg_A is his algorithm to check the answers. Suppose on inputs of length n it has question length $\ell_Q(n)$ and answer length $\ell_A(n)$. We define

$$L_{\text{Enc}} = \{(\text{input}, x_0, x_1, \text{Enc}_{\ell_A(|\text{input}|)}(y_0), \text{Enc}_{\ell_A(|\text{input}|)}(y_1)) \mid \text{Alg}_A(\text{input}, x_0, x_1, y_0, y_1) = 1\},$$

which are all the accepted tuples with the answers encoded by $\text{Enc}_{\ell_A(|\text{input}|)}$.

Note that when $|\text{input}| = n$, the running time of the decider of L_{Enc} is the maximal of the running time of Alg_A and $\text{Dec}_{\ell_A(n)}$ as pointed out in [39, Proposition 17.7]. Suppose $\gamma \leq \eta_H/2 = 1/4$. Then by [39, Proposition 17.8], if $(\text{input}, x_0, x_1, z_0, z_1)$ does not correspond to the encoding of any assignment accepted by Alg_A , for every proof π

$$\Pr_R[V_{\text{PCPP}}^{z_0, z_1, \pi}(\text{input}, x_0, x_1, |z_0| + |z_1|; R) = 1] \leq s$$

where s is the soundness of V_{PCPP} .

► **Definition 34.** We instantiate the answer-reduced MIP^* protocol with the following components and notations.

- 1) Let $V = (\text{Alg}_Q, \text{Alg}_A)$ be an MIP^* verifier for a Language L . Suppose on inputs of size n , the verifier V has question length $\ell_{V,Q}(n)$, answer length $\ell_{V,A}(n)$.
- 2) Let $G_k(\mathbf{T})$ be the subset tester from Section 6.1 for the Hadamard code of \mathbb{F}_2^k with the embedding μ_k , and for the subset \mathbf{T} sampled according to some distribution D .
- 3) Let L_{Enc} be the language defined in Definition 33, and let V_{PCPP} be its PCPP verifier with $\gamma \leq 1/4$ and constant soundness s . Suppose on inputs of size n it has proof length $\ell_\pi(n)$.
- 4) We write $\ell_1 := \ell_{V,A}(n)$ and $\ell_2 := \ell_\pi(n)$.

Next, we give the protocol of the answer reduced verifier V^{AR} , which requires the provers to encode their proof π by the Hadamard code of $\mathbb{F}_2^{\ell_2}$. The protocol is very similar to the protocol presented in [39, Figure 15], and can be found in Appendix B.4.

► **Theorem 35.** Let $V = (\text{Alg}_Q, \text{Alg}_A)$ be an MIP^* protocol for a language L . Suppose the PCPP verifier is chosen so that $\gamma \leq 1/4$. Suppose further that V has the following property: for any input $\in L$, the prover has a real commuting symmetric EPR strategy with a value 1. Then V^{AR} obtained by applying the the answer reduction procedure to V is also an MIP^* verifier for L with the following two conditions:

Completeness. If $\text{input} \in L$, there is a value-1 strategy for V^{AR} .

Soundness. Given input , suppose there is a strategy for V^{AR} with value $1 - \varepsilon$. Then there exists constants K_1 and K_2 such that there is a strategy for V on input with value $1 - K_1 - K_2\varepsilon^{1/96}$.

► **Theorem 36.** RE is contained in $\text{MIP}^*[\text{poly}, O(1)]$ with completeness 1 and a constant soundness.

Alternatively, we can first apply the answer reduction technique from [39] to the oracularized protocol to reduce its answer size to $O(\log(n))$ and then apply our answer reduction to further reduce it to $O(1)$. Compared with the approach above, this approach gives us an MIP^* protocol for RE with shorter questions but worse soundness.

References

- 1 Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM Journal on Computing*, 38(4):1207, 2008.
- 2 Dorit Aharonov, Xun Gao, Zeph Landau, Yunchao Liu, and Umesh Vazirani. A polynomial-time classical algorithm for noisy random circuit sampling. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 945–957, 2023.
- 3 Rotem Arnon-Friedman and Jean-Daniel Bancal. Device-independent certification of one-shot distillable entanglement. *New Journal of Physics*, 21(3):033010, 2019.
- 4 Rotem Arnon-Friedman, Zvika Brakerski, and Thomas Vidick. Computational entanglement theory. *arXiv preprint*, 2023. [arXiv:2310.02783](https://arxiv.org/abs/2310.02783).
- 5 Rotem Arnon-Friedman and Henry Yuen. Noise-Tolerant Testing of High Entanglement of Formation. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:12, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2018.11.
- 6 Srinivasan Arunachalam and Penghui Yao. Positive spectrahedra: invariance principles and pseudorandom generators. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 208–221, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3519965.
- 7 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, March 1991. doi:10.1007/BF01200056.
- 8 Ainesh Bakshi, Nadiia Chepurko, and Rajesh Jayaram. Testing positive semi-definiteness via random submatrices. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*, FOCS 2020, pages 1191–1202. IEEE, 2020.
- 9 Salman Beigi. A new quantum data processing inequality. *Journal of Mathematical Physics*, 54(8):082202, 2013. doi:10.1063/1.4818985.
- 10 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.
- 11 Dolev Bluvstein, Simon J Evered, Alexandra A Geim, Sophie H Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter, et al. Logical quantum processor based on reconfigurable atom arrays. *Nature*, pages 1–3, 2023.
- 12 Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J Bremner, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14(6):595–600, 2018.
- 13 J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, STOC 1977, pages 106–112, New York, NY, USA, 1977. Association for Computing Machinery. doi:10.1145/800105.803400.
- 14 Sitan Chen, Jordan Cotler, Hsin-Yuan Huang, and Jerry Li. The complexity of NISQ. *Nature Communications*, 14(1):6001, September 2023. doi:10.1038/s41467-023-41217-6.

- 15 R. Cleve, P. Hoyer, B. Toner, and J. Watrous. Consequences and limits of nonlocal strategies. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004.*, pages 236–249, 2004. doi:10.1109/CCC.2004.1313847.
- 16 Rodney Coleman. *Calculus on Normed Vector Spaces*. Springer-Verlag, New York, New York, NY, 1997.
- 17 Yangjing Dong, Honghao Fu, Anand Natarajan, Minglong Qin, Haochen Xu, and Penghui Yao. The computational advantage of MIP* vanishes in the presence of noise. *arXiv preprint*, 2023. arXiv:2312.04360.
- 18 Bill Fefferman and Zachary Remscrim. Eliminating intermediate measurements in space-bounded quantum computation. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2021*, pages 1343–1356, 2021.
- 19 Honghao Fu. Constant-sized correlations are sufficient to self-test maximally entangled states with unbounded dimension. *Quantum*, 6:614, January 2022. doi:10.22331/q-2022-01-03-614.
- 20 Badih Ghazi, Prithish Kamath, and Prasad Raghavendra. Dimension reduction for polynomials over gaussian space and applications. In *Proceedings of the 33rd Computational Complexity Conference, CCC '18*, pages 28:1–28:37, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2018.28.
- 21 Insu Han, Dmitry Malioutov, Haim Avron, and Jinwoo Shin. Approximating spectral sums of large-scale matrices using stochastic Chebyshev approximations. *SIAM Journal on Scientific Computing*, 39(4):A1558–A1585, 2017.
- 22 Prahladh Harsha, Adam Klivans, and Raghu Meka. An invariance principle for polytopes. *J. ACM*, 59(6), January 2013. doi:10.1145/2395116.2395118.
- 23 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001. doi:10.1145/502090.502098.
- 24 Marcus Isaksson and Elchanan Mossel. Maximally stable Gaussian partitions with discrete applications. *Israel Journal of Mathematics*, 189(1):347–396, 2012.
- 25 Tsuyoshi Ito, Hirotada Kobayashi, and Keiji Matsumoto. Oracularization and two-prover one-round interactive proofs against nonlocal strategies. In *Proceedings of the 2009 24th Annual IEEE Conference on Computational Complexity, CCC 2009*, pages 217–228, Washington, DC, USA, 2009. IEEE Computer Society. doi:10.1109/CCC.2009.22.
- 26 Tsuyoshi Ito and Thomas Vidick. A multi-prover interactive proof for NEXP sound against entangled provers. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science, FOCS 2012*, pages 243–252. IEEE, 2012.
- 27 Zhengfeng Ji. Compression of quantum multi-prover interactive proofs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 289–302, New York, NY, USA, 2017. ACM. doi:10.1145/3055399.3055441.
- 28 Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. MIP* = RE. *arXiv preprint*, 2020. arXiv:2001.04383.
- 29 Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. Quantum soundness of the classical low individual degree test. *arXiv preprint*, 2020. arXiv:2009.12982.
- 30 Daniel M. Kane. A Polylogarithmic PRG for Degree 2 Threshold Functions in the Gaussian Setting. In David Zuckerman, editor, *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 567–581, Dagstuhl, Germany, 2015. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2015.567.
- 31 Zander Kelley and Raghu Meka. Random restrictions and prgs for ptf in gaussian space. In *Proceedings of the 37th Computational Complexity Conference, CCC '22*, Dagstuhl, DEU, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2022.21.
- 32 Julia Kempe, Hirotada Kobayashi, Keiji Matsumoto, Ben Toner, and Thomas Vidick. Entangled games are hard to approximate. *SIAM Journal on Computing*, 40(3):848–877, 2011. doi:10.1137/090751293.

- 33 Julia Kempe, Oded Regev, and Ben Toner. Unique games with entangled provers are easy. *SIAM Journal on Computing*, 39(7):3207–3229, 2010. doi:10.1137/090772885.
- 34 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 767–775, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.510017.
- 35 Robert Krauthgamer and Ori Sasson. Property testing of data dimensionality. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2003, pages 18–27, USA, 2003. Society for Industrial and Applied Mathematics.
- 36 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC 2010, pages 427–436, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806749.
- 37 Elchanan Mossel, Ryan O’Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. In *46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS 2005, pages 21–30. IEEE, 2005.
- 38 Anand Natarajan and Thomas Vidick. A quantum linearity test for robustly verifying entanglement. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 1003–1015, 2017.
- 39 Anand Natarajan and John Wright. NEEXP is Contained in MIP*. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science*, FOCS 2019, pages 510–518. IEEE, 2019. doi:10.1109/FOCS.2019.00039.
- 40 Anand Natarajan and Tina Zhang. Quantum free games. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 1603–1616, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3564246.3585208.
- 41 Deanna Needell, William Swartworth, and David P. Woodruff. Testing positive semidefiniteness using linear measurements. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science*, FOCS 2022, pages 87–97, 2022. doi:10.1109/FOCS54457.2022.00016.
- 42 Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, Cambridge, UK, 2013.
- 43 Ryan O’Donnell, Rocco A. Servedio, and Li-Yang Tan. Fooling gaussian ptf’s via local hyperconcentration. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, pages 1170–1183, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3357713.3384281.
- 44 Ryan O’Donnell, Rocco A. Servedio, and Li-Yang Tan. Fooling polytopes. *J. ACM*, 69(2), January 2022. doi:10.1145/3460532.
- 45 Connor Paddock. Rounding near-optimal quantum strategies for nonlocal games to strategies using maximally entangled states. *arXiv preprint*, 2022. arXiv:2203.02525.
- 46 Minglong Qin and Penghui Yao. Nonlocal games with noisy maximally entangled states are decidable. *SIAM Journal on Computing*, 50(6):1800–1891, 2021.
- 47 Minglong Qin and Penghui Yao. Decidability of Fully Quantum Nonlocal Games with Noisy Maximally Entangled States. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 97:1–97:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.97.
- 48 Oded Regev and Liron Schiff. Impossibility of a quantum speed-up with a faulty oracle. In *International Colloquium on Automata, Languages, and Programming*, pages 773–781. Springer, 2008.
- 49 Ben W. Reichardt, Falk Unger, and Umesh Vazirani. Classical command of quantum systems. *Nature*, 496(7446):456–460, April 2013. doi:10.1038/nature12035.

- 50 Hristo S. Sendov. The higher-order derivatives of spectral functions. *Linear Algebra and its Applications*, 424(1):240–281, 2007. Special Issue in honor of Roger Horn. doi:10.1016/j.laa.2006.12.013.
- 51 Victor Shoup. New algorithms for finding irreducible polynomials over finite fields. *Mathematics of computation*, 54(189):435–447, 1990.
- 52 Anna Skripka and Anna Tomskova. *Multilinear operator integrals*. Springer, 2019.
- 53 William Slofstra. The set of quantum correlations is not closed. *Forum of Mathematics, Pi*, 7:e1, 2019. doi:10.1017/fmp.2018.3.
- 54 William Slofstra. Tsirelson’s problem and an embedding theorem for groups arising from non-local games. *Journal of the American Mathematical Society*, 33:1–56, 2020. doi:/10.1090/jams/929.
- 55 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/04000000010.
- 56 Thomas Vidick. Almost synchronous quantum correlations. *Journal of mathematical physics*, 63(2), 2022.

A Preliminary

For $n \in \mathbb{Z}_{>0}$, let $[n]$ and $[n]_{\geq 0}$ represent the sets $\{1, \dots, n\}$ and $\{0, \dots, n-1\}$, respectively. Given a finite set \mathcal{X} and a natural number k , let \mathcal{X}^k be the set $\mathcal{X} \times \dots \times \mathcal{X}$, the Cartesian product of \mathcal{X} , k times. For any $\sigma \in \mathbb{Z}_{\geq 0}^k$, we define $|\sigma| = |\{i : \sigma_i \neq 0\}|$.

In this paper, the lowercase letters in bold $\mathbf{x}, \mathbf{y}, \dots$ are reserved for random variables. The capital letters in bold, $\mathbf{A}, \mathbf{B}, \dots$ are reserved for random operators.

A.1 Quantum Mechanics

A quantum system is associated with a complex finite-dimensional Hilbert space, denoted by A . A quantum state in A can be completely described by a density operator, a positive semidefinite operator with trace one. If the dimension of A is m , we denote the set of Hermitian matrices in A by \mathcal{H}_m . The identity matrix is denoted by $\mathbb{1}_m$ or $\mathbb{1}_A$. The state of a composite quantum system is the Kronecker product of the state spaces of the component systems. An important operation on a composite system $A \otimes B$ is the *partial trace* $\text{Tr}_B(\cdot)$ which effectively derives the marginal state of the subsystem A (denoted by ψ_A) from the quantum state ψ_{AB} . The partial trace is given by

$$\psi_A = \text{Tr}_B \psi_{AB} = \sum_i (\mathbb{1}_A \otimes \langle i |) \psi_{AB} (\mathbb{1}_A \otimes |i\rangle),$$

where $\{|i\rangle\}$ is an orthonormal basis in B . A linear map from a system A to a system B is *unital* if it maps $\mathbb{1}_A$ to $\mathbb{1}_B$. A *quantum measurement* is represented by a *positive operator-valued measure* (POVM), which is a set of positive semidefinite operators $\{M_1, \dots, M_n\}$ satisfying $\sum_{i=1}^n M_i = \mathbb{1}$, where n is the number of possible measurement outcomes. Suppose that the state of the quantum system is ψ , then the probability that it produces i is $\text{Tr } M_i \psi$. We use $\vec{M} = (M_1, \dots, M_n)$ to represent an ordered set of operators.

A.2 Matrix Analysis

A.2.1 Matrix Spaces

Given $m \in \mathbb{Z}_{>0}$ and $M \in \mathcal{H}_m$, we use $M_{i,j}$ to represent the (i,j) -th entry of M . For $1 \leq p \leq \infty$, the p -norm of M is defined to be

$$\|M\|_p = \left(\sum_{i=1}^m s_i(M)^p \right)^{1/p},$$

where $(s_1(M), s_2(M), \dots, s_m(M))$ are the singular values of M sorted in nonincreasing order. $\|M\| = \|M\|_\infty = s_1(M)$. The *normalized p -norm* of M is defined as

$$\| \|M\| \|_p = \left(\frac{1}{m} \sum_{i=1}^m s_i(M)^p \right)^{1/p} \quad (4)$$

and $\| \|M\| \| = \| \|M\| \|_\infty = s_1(M)$.

Given $P, Q \in \mathcal{M}_m$, we define

$$\langle P, Q \rangle = \frac{1}{m} \text{Tr } P^\dagger Q. \quad (5)$$

It is easy to verify that $\langle \cdot, \cdot \rangle$ is an inner product. $(\langle \cdot, \cdot \rangle, \mathcal{H}_m)$ forms a Hilbert space. For any $M \in \mathcal{H}_m$, $\| \|M\| \|_2^2 = \langle M, M \rangle$.

We say that $\{\mathcal{B}_0, \dots, \mathcal{B}_{m^2-1}\}$ is a *standard orthonormal basis* in \mathcal{M}_m if it is an orthonormal basis with all elements being Hermitian and $\mathcal{B}_0 = \mathbb{1}_m$.

► **Fact 37** ([46, Lemma 2.10]). *For any integer $m \geq 2$, a standard orthonormal basis exists in \mathcal{M}_m .*

Given a standard orthonormal basis $\mathcal{B} = \{\mathcal{B}_i\}_{i=0}^{m^2-1}$ in \mathcal{H}_m , every matrix $M \in \mathcal{H}_m^{\otimes n}$ has a *Fourier expansion* with respect to the basis \mathcal{B} given by

$$M = \sum_{\sigma \in [m^2]_{\geq 0}^n} \widehat{M}(\sigma) \mathcal{B}_\sigma,$$

where $\mathcal{B}_\sigma = \bigotimes_{i=1}^n \mathcal{B}_{\sigma_i}$.

► **Definition 38.** *Let $\mathcal{B} = \{\mathcal{B}_i\}_{i=0}^{m^2-1}$ be a standard orthonormal basis in \mathcal{H}_m , $P \in \mathcal{H}_m^{\otimes n}$.*

1. *The degree of P is defined to be*

$$\text{deg } P = \max \left\{ |\sigma| : \widehat{P}(\sigma) \neq 0 \right\}.$$

Recall that $|\sigma|$ represents the number of nonzero entries of σ .

2. *For any $i \in [n]$, the influence of i -th coordinate is defined to be:*

$$\text{Inf}_i(P) = \| \|P - \mathbb{1}_m \otimes \text{Tr}_i P\| \|_2^2,$$

where $\mathbb{1}_m$ is in the i 'th quantum system, and the partial trace Tr_i derives the marginal state of the remaining $n - 1$ quantum systems except for the i 'th one.

3. *The total influence is defined by*

$$\text{Inf}(P) = \sum_i \text{Inf}_i(P).$$

► **Fact 39** ([46, Lemma 2.16]). Given $P \in \mathcal{H}_m^{\otimes n}$, a standard orthonormal basis $\mathcal{B} = \{\mathcal{B}_i\}_{i=0}^{m^2-1}$ in \mathcal{H}_m and a subset $S \subseteq [n]$, it holds that

1. $\text{Inf}_i(P) = \sum_{\sigma: \sigma_i \neq 0} |\widehat{P}(\sigma)|^2$;
2. $\text{Inf}(P) = \sum_{\sigma} |\sigma| |\widehat{P}(\sigma)|^2 \leq \text{deg } P \cdot \|P\|_2^2$.

The inequality in item 2 follows from Parseval's identity, which is immediate by the Fourier expansion of P (Fact 37).

► **Fact 40** (Parseval's identity). For any $P \in \mathcal{H}_m^{\otimes n}$,

$$\|P\|_2^2 = \sum_{\sigma} |\widehat{P}(\sigma)|^2.$$

Quantum maximal correlations introduced by Beigi [9] are crucial to our analysis.

► **Definition 41** (Maximal correlation [9]). Given quantum systems A, B of dimension m and a bipartite state ψ_{AB} with $\psi_A = \psi_B = \frac{\mathbb{1}_m}{m}$, the maximal correlation of ψ_{AB} is defined to be

$$\rho(\psi_{AB}) = \sup \left\{ |\text{Tr}((P^\dagger \otimes Q)\psi_{AB})| : \begin{array}{l} P, Q \in \mathbb{C}^{m \times m}, \\ \text{Tr } P = \text{Tr } Q = 0, \|P\|_2 = \|Q\|_2 = 1. \end{array} \right\}$$

► **Fact 42** ([9]). Given quantum systems A, B and a bipartite quantum state ψ_{AB} with $\psi_A = \mathbb{1}_{m_A}/m_A$ and $\psi_B = \mathbb{1}_{m_B}/m_B$, it holds that $\rho(\psi_{AB}) \leq 1$.

► **Definition 43**. Given quantum systems A and B with $\dim(A) = \dim(B) = m$, a bipartite state $\psi_{AB} \in \mathcal{D}(A \otimes B)$ is an m -dimensional noisy maximally entangled state (MES) if $\psi_A = \psi_B = \mathbb{1}_m/m$ and its maximal correlation $\rho = \rho(\psi_{AB}) < 1$.

An interesting class of noisy MESs is the isotropic states, which are the states obtained by depolarizing MESs with arbitrarily small noise.

► **Fact 44** ([46, Lemma 3.9]). For any $0 \leq \epsilon < 1$ integer $m > 1$, it holds that

$$\rho \left((1 - \epsilon) |\Psi\rangle\langle\Psi| + \epsilon \frac{\mathbb{1}_m}{m} \otimes \frac{\mathbb{1}_m}{m} \right) = 1 - \epsilon,$$

where $|\Psi\rangle = \frac{1}{\sqrt{m}} \sum_{i=0}^{m-1} |m, m\rangle$ is an m -dimensional MES.

► **Remark 45**. Fact 44 indicates the maximal correlation of an isotropic state is strictly less than 1. The class of noisy MES also contains other states. It is not hard to prove that any mixture of at least three out of the four orthogonal EPR states is a 2-dimensional noisy MES.

► **Fact 46** ([46, Lemma 7.4]). Given $m \in \mathbb{Z}_{>0}$, $m \geq 2$, and a noisy m -dimensional MES ψ_{AB} . Then there exist standard orthonormal bases $\mathcal{A} = \{\mathcal{A}_i\}_{i=0}^{m^2-1}$ and $\mathcal{B} = \{\mathcal{B}_i\}_{i=0}^{m^2-1}$ in \mathcal{H}_m such that

$$\text{Tr}((\mathcal{A}_i \otimes \mathcal{B}_j)\psi_{AB}) = \begin{cases} c_i & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \quad (6)$$

where $c_0 = 1 \geq c_1 = \rho(\psi_{AB}) \geq c_2 \geq \dots \geq c_{m^2-1} \geq 0$ and $\rho(\psi_{AB})$ is defined in Definition 41.

► **Definition 47**. Given $m \in \mathbb{Z}_{>0}$, $\rho \in [0, 1]$, a noise operator $\Delta_\rho: \mathcal{H}_m \rightarrow \mathcal{H}_m$ is defined as follows. For any $P \in \mathcal{H}_m$,

$$\Delta_\rho(P) = \rho P + \frac{1 - \rho}{m} (\text{Tr } P) \cdot \mathbb{1}_m.$$

With a slight abuse of notations, the noise operator $\Delta_\rho^{\otimes n}$ on the space $\mathcal{H}_m^{\otimes n}$ is also denoted by Δ_ρ .

► **Fact 48** ([46, Lemma 3.5]). *Given integers $d, n, m > 0$, $\rho \in [0, 1]$, a standard orthonormal basis of \mathcal{H}_m : $\mathcal{B} = \{\mathcal{B}_i\}_{i=0}^{m^2-1}$, then for any $P \in \mathcal{H}_m^{\otimes n}$ with a Fourier expansion $P = \sum_{\sigma \in [m^2]_{\geq 0}^n} \widehat{P}(\sigma) \mathcal{B}_\sigma$, it holds that*

$$\Delta_\rho(P) = \sum_{\sigma \in [m^2]_{\geq 0}^n} \rho^{|\sigma|} \widehat{P}(\sigma) \mathcal{B}_\sigma.$$

A.2.2 Random Matrices

For integer $n \geq 1$, γ_n represents the distribution of an n -dimensional standard normal distribution. For any $0 \leq \rho \leq 1$, \mathcal{G}_ρ represents a ρ -correlated Gaussian distribution, which is a 2-dimensional Gaussian distribution

$$(X, Y) \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right).$$

Namely, the marginal distributions X and Y are distributed according to γ_1 and $\mathbb{E}[XY] = \rho$.

► **Definition 49.** *Given $h, n, m \in \mathbb{Z}_{>0}$, we say $P(\mathbf{g})$ is a random matrix if it can be expressed as*

$$P(\mathbf{g}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{g}) \mathcal{B}_\sigma, \tag{7}$$

where $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$ is a standard orthonormal basis in \mathcal{H}_m , $p_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}$ for all $\sigma \in [m^2]_{\geq 0}^h$ and $\mathbf{g} \sim \gamma_n$. Moreover, we say $P(\mathbf{g}) \in L^2(\mathcal{H}_m^{\otimes h}, \gamma_n)$ if $\int_{\mathbb{R}^n} p_\sigma^2(x) \gamma_n(dx) < \infty$ for all $\sigma \in [m^2]_{\geq 0}^h$.

We define the degree of random operators:

► **Definition 50.** *Given integers $n, h > 0, m > 1$ and random operator $\mathbf{P} \in L^p(\mathcal{H}_m^{\otimes h}, \gamma_n)$, the degree of \mathbf{P} , denoted by $\deg(\mathbf{P})$, is*

$$\max_{\sigma \in [m^2]_{\geq 0}^h} \deg(p_\sigma).$$

We say \mathbf{P} is multilinear if $p_\sigma(\cdot)$ is multilinear for all $\sigma \in [m^2]_{\geq 0}^h$.

A.2.3 Fréchet Derivatives and Spectral Functions

The Fréchet derivatives are derivatives on Banach spaces. In this paper, we only concern ourselves with Fréchet derivatives on matrix spaces. Readers may refer to [16] for a detailed treatment.

► **Definition 51.** *Given a map $f : \mathcal{H}_m \rightarrow \mathcal{H}_m$ and $P, Q \in \mathcal{H}_m$, the Fréchet derivative of f at P with direction Q is defined to be*

$$Df(P)[Q] = \left. \frac{d}{dt} f(P + tQ) \right|_{t=0}.$$

The k -th order Fréchet derivative of f at P with direction (Q_1, \dots, Q_k) is defined to be

$$D^k f(P)[Q_1, \dots, Q_k] = \left. \frac{d}{dt} (D^{k-1} f(P + tQ_k)[Q_1, \dots, Q_{k-1}]) \right|_{t=0}.$$

To keep notations short, we use $D^k f(P)[Q]$ to represent $D^k f(P)[Q, \dots, Q]$.

In this paper, we are concerned with *spectral functions*, a special class of matrix functions. We say that the function $F : \mathcal{H}_m \rightarrow \mathcal{H}_m$ is a spectral function if there exists a function $f : \mathbb{R} \rightarrow \mathbb{R}$ such that $f(P) = \sum_i f(\lambda_i) |v_i\rangle\langle v_i|$, where $P = \sum_i \lambda_i |v_i\rangle\langle v_i|$ is a spectral decomposition of P . With slight abuse of notation, we use the same notation f to represent the function on \mathbb{R} and the corresponding spectral function, whenever it is clear from the context.

Given $n \in \mathbb{Z}_{>0}$, we denote \mathcal{C}^n to be the space of functions continuously differentiable n times.

► **Definition 52.** Let $\lambda_0, \dots, \lambda_n \in \mathbb{R}$ and let $f \in \mathcal{C}^n$. The divided difference $f^{[n]}$ is defined recursively by

$$f^{[n]}(\lambda_0, \lambda_1, \tilde{\lambda}) = \begin{cases} \frac{f^{[n-1]}(\lambda_0, \tilde{\lambda}) - f^{[n-1]}(\lambda_1, \tilde{\lambda})}{\lambda_0 - \lambda_1} & \text{if } \lambda_0 \neq \lambda_1, \\ \frac{d}{d\lambda_0} f^{[n-1]}(\lambda_0, \tilde{\lambda}) & \text{if } \lambda_0 = \lambda_1, \end{cases}$$

where $\tilde{\lambda} = (\lambda_2, \dots, \lambda_n)$.

It is well known that $f^{[n]}$ is a symmetric function.

► **Fact 53** ([52, Theorem 5.3.2], [50, Theorem 6.1]). Given $m, n \in \mathbb{Z}_{>0}$, $P, Q \in \mathcal{H}_m$. Suppose that P has a spectral decomposition

$$P = \sum_{i=1}^m \lambda_i \Pi_i, \tag{8}$$

where $\lambda_1 \geq \dots \geq \lambda_m$, $\{\Pi_i\}_{i \in [m]}$ are rank-one projectors satisfying that $\sum_{i=1}^m \Pi_i = \mathbb{1}$ and $\Pi_i \Pi_j = 0$ for all $i \neq j$. Let $f \in \mathcal{C}^n$. Then

$$D^n f(P)[Q] = \sum_{i_0, \dots, i_n \in [m]} f^{[n]}(\lambda_{i_0}, \dots, \lambda_{i_n}) \Pi_{i_0} Q \Pi_{i_1} Q \dots Q \Pi_{i_n}.$$

► **Fact 54** ([52, Theorem 5.3.12]). Given $m, n \in \mathbb{Z}_{>0}$, $P, Q \in \mathcal{H}_m$. Let $f \in \mathcal{C}^n$. Denote

$$\Delta_{n,f}(P, Q) = f(P + Q) - \sum_{k=0}^{n-1} \frac{1}{k!} D^k f(P)[Q],$$

then there exists a constant c_n depending only on n such that

$$|\text{Tr}[\Delta_{n,f}(P, Q)]| \leq c_n \|f^{(n)}\|_\infty \|Q\|_n^n,$$

where $\|f^{(n)}\|_\infty$ denotes the supremum of $f^{(n)}$.

A.2.4 The Distance from PSD Matrices

Define the function $\zeta : \mathbb{R} \rightarrow \mathbb{R}$ as follows.

$$\zeta(x) = \begin{cases} x^2 & \text{if } x \leq 0 \\ 0 & \text{otherwise} \end{cases}. \tag{9}$$

The function ζ measures the distance between a given matrix and its closest positive semi-definite matrix:

► **Fact 55** ([46, Lemma 9.1]). Given an integer $m > 0$, $M \in \mathcal{H}_m$, $\Delta = \{X \in \mathcal{H}_m : X \geq 0\}$, let

$$\mathcal{R}(M) = \arg \min \{\|M - X\|_2 : X \in \Delta\}$$

be a rounding map of Δ with respect to the distance $\|\cdot\|_2$. It holds that

$$\mathrm{Tr} \zeta(M) = \|M - \mathcal{R}(M)\|_2^2.$$

► **Fact 56** ([46, Lemma 10.4]). For any Hermitian matrices P and Q , it holds that

$$|\mathrm{Tr}(\zeta(P + Q) - \zeta(P))| \leq 2(\|P\|_2\|Q\|_2 + \|Q\|_2^2).$$

We will need to let ζ to be mollified⁵ to get a smooth function:

► **Fact 57** ([37, Lemma 3.21]). Given $\lambda > 0$, there exists a C^∞ function ζ_λ satisfying

1. $\|\zeta_\lambda - \zeta\|_\infty \leq 2\lambda^2$,
2. For any integer $n \geq 2$, there exists a constant B_n independent of λ such that

$$\|(\zeta_\lambda)^{(n)}\|_\infty \leq B_n \lambda^{2-n}.$$

A.3 k -wise Uniform Hash Functions and Random Variables

► **Definition 58.** A family $\mathcal{F} = \{f : [n] \rightarrow [p]\}$ of hash functions is k -wise uniform if for any $y_1, \dots, y_k \in [p]$ and distinct $x_1, \dots, x_k \in [n]$:

$$\Pr_{f \in \mathcal{F}} [f(x_i) = y_i \wedge \dots \wedge f(x_k) = y_k] = \frac{1}{p^k}.$$

► **Definition 59.** A random vector $\mathbf{z} \in [p]^n$ is k -wise uniform if for any $y_1, \dots, y_k \in [p]$ and distinct $x_1, \dots, x_k \in [n]$:

$$\Pr_{\mathbf{z}} [\mathbf{z}_{x_i} = y_i \wedge \dots \wedge \mathbf{z}_{x_k} = y_k] = \frac{1}{p^k}.$$

► **Lemma 60.** Let p be a power of 2. There exists an efficient construction of k -wise uniform hash functions $\mathcal{F} = \{f : [n] \rightarrow [p]\}$ of size $|\mathcal{F}| = O(\max(n, p)^k)$.

Proof. For $k = 2$, efficient constructions of size $|\mathcal{F}| = O(np)$ are well known (see, e.g., [13]). For general k , let t be the minimal integer satisfying $2^t > \max(n, p)$ and consider the finite field \mathbb{F}_{2^t} . We can construct an irreducible polynomial in \mathbb{F}_2 of degree t in polynomial time, using, for example, the algorithms of Shoup [51]. Thus, the basic operations in \mathbb{F}_{2^t} can be carried out efficiently. Then the k -wise uniform hash functions $\tilde{\mathcal{F}} : \{\tilde{f} : \mathbb{F}_{2^t} \rightarrow \mathbb{F}_{2^t}\}$ can be efficiently constructed, for example, using the construction in Section 3.5.5 in [55], which has size $|\mathbb{F}_{2^t}|^k = O(\max(n, p)^k)$. Then k -wise uniform hash functions from $[n]$ to \mathbb{F}_{2^t} can be constructed by restricting the input domain to $[n]$. k -wise uniform hash functions from $[n]$ to $[p]$ can be further constructed by cutting the output to $\log p$ bits. ◀

► **Corollary 61.** There exists an efficient construction of k -wise uniform random variables $\mathbf{z} \sim \{-1, 1\}^n$, which can be enumerated in $O(n^k)$ time.

Proof. Construct k -wise uniform hash functions $\mathcal{F} = \{f : [n] \rightarrow \{-1, 1\}\}$, and then define $\mathbf{z} = (f(1), \dots, f(n))$. By the definition of k -wise uniform hash functions, \mathbf{z} is k -wise uniform random variables. Moreover, the construction of \mathcal{F} is efficient. Finally, the enumeration of \mathbf{z} takes time $O(n^k)$ since we only need to enumerate the set \mathcal{F} . ◀

⁵ A mollified function ζ_λ is a smooth function that is close to the original function ζ .

A.4 Lemmas for Noisy MIP*

Smoothing

The following lemma reduces the degrees of the POVMs of an MIP* strategy.

► **Lemma 62.** [46, Lemma 6.1]⁶ Given parameters $0 \leq \rho < 1$, $0 < \delta < 1$, $n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, and an m -dimensional noisy MES ψ_{AB} with the maximal correlation $\rho = \rho(\psi_{AB})$, there exists $d = d(\rho, \delta)$ and a map $f : \mathcal{H}_m^{\otimes n} \rightarrow \mathcal{H}_m^{\otimes n}$, such that for any positive semi-definite matrices $P, Q \in \mathcal{H}_m^{\otimes n}$ satisfying $\|P\|_2 \leq 1$ and $\|Q\|_2 \leq 1$. The matrices $P^{(1)} = f(P)$ and $Q^{(1)} = f(Q)$ satisfy that

1. $P^{(1)}$ and $Q^{(1)}$ are of degree at most d .
2. $\|P^{(1)}\|_2 \leq 1$ and $\|Q^{(1)}\|_2 \leq 1$.
3. $|\text{Tr}((P^{(1)} \otimes Q^{(1)}) \psi_{AB}^{\otimes n}) - \text{Tr}((P \otimes Q) \psi_{AB}^{\otimes n})| \leq \delta$.
4. $\frac{1}{m^n} \text{Tr} \zeta(P^{(1)}) \leq \delta$ and $\frac{1}{m^n} \text{Tr} \zeta(Q^{(1)}) \leq \delta$.
5. the map f is linear and unital.

In particular, we can take $d = \frac{C \log^2 \frac{1}{\delta}}{\delta(1-\rho)}$ for some absolute constant C .

► **Remark 63.** It is easily verified that for the above lemma, for each $\sigma \in [m^2]_{\geq 0}^n$, we have

$$|\widehat{P}^{(1)}(\sigma)| \leq |\widehat{P}(\sigma)| \text{ and } |\widehat{Q}^{(1)}(\sigma)| \leq |\widehat{Q}(\sigma)|.$$

This is because in fact f applies depolarizing noise on P and then eliminates the high degree parts. So the Fourier coefficients are non-increasing in absolute value.

Regularization

The following lemma allows us to identify high-influence registers, and the number of such registers can be upper-bounded.

► **Lemma 64.** [46, Lemma 7.4] Given $0 < \tau < 1$, $d, n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, and a degree- d matrix $P \in \mathcal{H}_m^{\otimes n}$ satisfying $\|P\|_2 \leq 1$, there exists a subset $H \subseteq [n]$ of size $h = |H| \leq \frac{d}{\tau}$ such that for any $i \notin H$,

$$\text{Inf}_i(P^{\leq d}) \leq \tau.$$

Rounding

The following lemma shows that we can round a given set of matrices that sum up to $\mathbb{1}$ to a close-by POVM.

► **Lemma 65.** Given $\vec{X} \in (\mathcal{H}_m^{\otimes n})^t$ satisfying that $\sum_{i=1}^t X_i = \mathbb{1}$, define

$$\mathcal{R}(\vec{X}) = \arg \min \left\{ \left\| \vec{X} - \vec{P} \right\|_2^2 : \vec{P} \text{ is a POVM} \right\}$$

It holds that

$$\left\| \mathcal{R}(\vec{X}) - \vec{X} \right\|_2^2 \leq \frac{3(t+1)}{m^n} \sum_{i=1}^t \text{Tr} \zeta(X_i) + 6 \left(\frac{t}{m^n} \sum_{i=1}^t \text{Tr} \zeta(X_i) \right)^{1/2}.$$

⁶ The statement is slightly different from that in [46, Lemma 6.1]. The difference arises due to our relocation of the truncating step, which was in [46, Lemma 10.5].

Miscellaneous Lemmas

The following lemmas are used throughout Appendix B.3.

► **Fact 66** ([46, Fact 2.1]). *Given registers A, B , operators $P \in \mathcal{H}(A), Q \in \mathcal{H}(B)$ and a bipartite state ψ_{AB} , it holds that*

$$|\mathrm{Tr}((P \otimes Q)\psi_{AB})| \leq (\mathrm{Tr} P^2 \psi_A)^{1/2} \cdot (\mathrm{Tr} Q^2 \psi_B)^{1/2}.$$

► **Lemma 67.** *Let $\{P_a^x\}_{a \in \mathcal{A}}, \{Q_b^y\}_{b \in \mathcal{B}}, \{\tilde{P}_a^x\}_{a \in \mathcal{A}}, \{\tilde{Q}_b^y\}_{b \in \mathcal{B}} \subseteq \mathcal{H}_m^{\otimes n}$ be four sets of matrices. If for all $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$,*

$$|\mathrm{Tr}((P_a^x \otimes Q_b^y)\psi_{AB}^{\otimes n}) - \mathrm{Tr}((\tilde{P}_a^x \otimes \tilde{Q}_b^y)\psi_{AB}^{\otimes n})| \leq \delta \|P_a^x\|_2 \|Q_b^y\|_2$$

for some $\delta > 0$. Then

$$|\mathrm{val}_n(\{P_a^x\}, \{Q_b^y\}) - \mathrm{val}_n(\{\tilde{P}_a^x\}, \{\tilde{Q}_b^y\})| \leq \delta t \left(\sum_{x,a} \mu_A(x) \|P_a^x\|_2^2 \right)^{1/2} \left(\sum_{y,b} \mu_B(y) \|Q_b^y\|_2^2 \right)^{1/2}.$$

Proof.

$$\begin{aligned} & |\mathrm{val}_n(\{P_a^x\}, \{Q_b^y\}) - \mathrm{val}_n(\{\tilde{P}_a^x\}, \{\tilde{Q}_b^y\})| \\ & \leq \sum_{x,y,a,b} \mu(x,y) |\mathrm{Tr}((P_a^x \otimes Q_b^y)\psi_{AB}^{\otimes n}) - \mathrm{Tr}((\tilde{P}_a^x \otimes \tilde{Q}_b^y)\psi_{AB}^{\otimes n})| \\ & \leq \delta \sum_{x,y,a,b} \mu(x,y) \|P_a^x\|_2 \|Q_b^y\|_2 \\ & \leq \delta \left(\sum_{x,y,a,b} \mu(x,y) \|P_a^x\|_2^2 \right)^{1/2} \left(\sum_{x,y,a,b} \mu(x,y) \|Q_b^y\|_2^2 \right)^{1/2} \quad (\text{Cauchy Schwarz}) \\ & = \delta t \left(\sum_{x,a} \mu_A(x) \|P_a^x\|_2^2 \right)^{1/2} \left(\sum_{y,b} \mu_B(y) \|Q_b^y\|_2^2 \right)^{1/2}. \quad \blacktriangleleft \end{aligned}$$

► **Lemma 68** (Truncation). *Let $\{P_a^x\}, \{Q_b^y\}$ be two sets of operators satisfying*

1. For all $x, y, \sum_a P_a^x = \sum_b Q_b^y = \mathbb{1}$.
2. For all $x, a, y, b, \sigma, |\hat{P}_a^x(\sigma)| \leq 1$ and $|\hat{Q}_b^y(\sigma)| \leq 1$.

Let $s_w = D \log m + \log(\frac{2}{\delta})$. Then there exist operators $\{P_a^{x,(2)}\}, \{Q_b^{y,(2)}\}$ satisfying

1. For each x, y, a, b, σ , the Fourier coefficients of $P_a^{x,(2)}$ and $Q_b^{y,(2)}$ consists of at most s_w bits.
2. For all $x, y, \sum_a P_a^{x,(2)} = \sum_b Q_b^{y,(2)} = \mathbb{1}$.
3. For all $x, y, a, b, \left\| P_a^{x,(2)} \right\|_2 \leq 1$ and $\left\| Q_b^{y,(2)} \right\|_2 \leq 1$.
4. For all $x, y, a, b, \left| \mathrm{Tr} \left((P_a^{x,(2)} \otimes Q_b^{y,(2)}) \psi_{AB}^{\otimes n} \right) - \mathrm{Tr} \left((P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n} \right) \right| \leq \delta$.
5. For all $x, y, a, b,$

$$\left| \frac{1}{m^D} \mathrm{Tr} \zeta \left(P_a^{x,(2)} \right) - \frac{1}{m^D} \mathrm{Tr} \zeta \left(P_a^x \right) \right| \leq \delta \text{ and } \left| \frac{1}{m^D} \mathrm{Tr} \zeta \left(Q_b^{y,(2)} \right) - \frac{1}{m^D} \mathrm{Tr} \zeta \left(Q_b^y \right) \right| \leq \delta.$$

Proof. Let $\alpha = 2^{-s_w} = \delta/(2m^D)$. For each x, y, σ , define $\widehat{P}_a^{x,(1)}(\sigma) = \lfloor \widehat{P}_a^x(\sigma)/\alpha \rfloor \alpha$. For each $x, \sigma \neq 0^D$, define integer $k_{x,\sigma}$ as

$$-\sum_a \widehat{P}_a^{x,(1)}(\sigma) = k_{x,\sigma} \cdot \alpha$$

and for $\sigma = 0^D$, define

$$1 - \sum_a \widehat{P}_a^{x,(1)}(\sigma) = k_{x,0^D} \cdot \alpha.$$

Let $t_{x,\sigma} = \left| \left\{ a \in \mathcal{A} : \widehat{P}_a^{x,(1)}(\sigma) \neq \widehat{P}_a^x(\sigma) \right\} \right|$, we can see that $0 \leq k_{x,\sigma} < t_{x,\sigma}$ always holds because $\sum_a P_a^x = \mathbf{1}$ and by the fact that $\widehat{P}_a^{x,(1)}(\sigma) > \widehat{P}_a^x(\sigma) - \alpha$. Let $S_{x,\sigma}$ be an arbitrary subset of $\left\{ a \in \mathcal{A} : \widehat{P}_a^{x,(1)}(\sigma) \neq \widehat{P}_a^x(\sigma) \right\}$ of size $k_{x,\sigma}$. Define $P_a^{x,(2)}$ as

$$\widehat{P}_a^{x,(2)}(\sigma) = \begin{cases} \widehat{P}_a^{x,(1)}(\sigma) & \text{if } a \notin S_{x,\sigma} \\ \widehat{P}_a^{x,(1)}(\sigma) + \alpha & \text{if } a \in S_{x,\sigma} \end{cases}$$

Then item 1 and item 2 hold for $P_a^{x,(2)}$. Also, since for $a \in S_{x,\sigma}$ we have $\widehat{P}_a^{x,(1)}(\sigma) < \widehat{P}_a^x(\sigma) \leq 1$, we have $\widehat{P}_a^{x,(2)}(\sigma) \leq 1 - \alpha$. So, it can be verified that $|\widehat{P}_a^{x,(2)}(\sigma)| \leq 1$ always holds, which implies that item 3 also holds. To prove the remaining items, we need

$$\left\| P_a^x - P_a^{x,(2)} \right\|_2 = \sqrt{\sum_{\sigma} \left(\widehat{P}_a^x(\sigma) - \widehat{P}_a^{x,(2)}(\sigma) \right)^2} < \sqrt{\sum_{\sigma} \alpha^2} \leq m^D \alpha.$$

We can apply the same operations to $\{Q_b^y\}$ and get $\{Q_b^{y,(2)}\}$. Then for all x, y, a, b ,

$$\begin{aligned} & \left| \text{Tr} \left(\left(P_a^{x,(2)} \otimes Q_b^{y,(2)} \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^x \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) \right| \\ & \leq \left| \text{Tr} \left(\left(P_a^{x,(2)} \otimes Q_b^{y,(2)} \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^{x,(2)} \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) \right| \\ & \quad + \left| \text{Tr} \left(\left(P_a^{x,(2)} \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^x \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) \right| \\ & = \left| \text{Tr} \left(\left(P_a^{x,(2)} \otimes \left(Q_b^{y,(2)} - Q_b^y \right) \right) \psi_{AB}^{\otimes n} \right) \right| + \left| \text{Tr} \left(\left(\left(P_a^{x,(2)} - P_a^x \right) \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) \right| \\ & \leq \left\| P_a^{x,(2)} \right\|_2 \left\| Q_b^{y,(2)} - Q_b^y \right\|_2 + \left\| P_a^{x,(2)} - P_a^x \right\|_2 \left\| Q_b^y \right\|_2 \leq 2m^D \alpha = \delta, \end{aligned}$$

and item 4 follows. Then item 5 follows from Fact 56. \blacktriangleleft

A.5 Lemmas for the Answer Reduction of MIP*

This section introduces several lemmas to prove the hardness of MIP*(poly, $O(1)$). We use the following notations for approximation in this section and Section 6.

- For complex numbers a and b , we write $a \approx_{\delta} b$ if $|a - b| \leq \delta$.
- With respect to a distribution D on \mathcal{X} and state $|\psi\rangle$, we write

$$A_a^x \approx_{\delta} B_a^x \quad \text{if} \quad \mathbb{E}_{x \sim D} \sum_{a \in \mathcal{A}} \left\| (A_a^x - B_a^x) |\psi\rangle \right\|^2 \leq \delta.$$

- With respect to a distribution D on \mathcal{X} and state $|\psi\rangle$, we write

$$A_a^x \simeq_{\delta} B_a^x \quad \text{if} \quad \mathbb{E}_{x \sim D} \sum_{a \in \mathcal{A}} \langle \psi | A_a^x \otimes B_a^x | \psi \rangle \geq 1 - \delta.$$

30:32 The Computational Advantage of MIP* Vanishes in the Presence of Noise

In the rest of the section, the distribution on \mathcal{X} is implicit.

► **Lemma 69** (Fact 4.13 of [39]). *Let $\{A_a^x\}$ and $\{B_a^x\}$ be POVM measurements. If $A_a^x \otimes \mathbf{1} \simeq_\delta \mathbf{1} \otimes B_a^x$, then $A_a^x \otimes \mathbf{1} \approx_{2\delta} \mathbf{1} \otimes B_a^x$.*

► **Lemma 70.** *Suppose $\{A_a^x\}$ and $\{B_a^x\}$ are two measurements such that one of them is projective, and that*

$$A_a^x \otimes \mathbf{1} \approx_\delta \mathbf{1} \otimes B_a^x$$

with respect to some distribution D of x and the quantum state $|\psi\rangle$. Then

$$\left| \mathbb{E}_x \sum_a \langle \psi | A_a^x \otimes \mathbf{1} - \mathbf{1} \otimes B_a^x | \psi \rangle \right| \leq 2\sqrt{\delta}.$$

Proof of Lemma 70. We assume $\{A_a^x\}$ is projective. Then

$$\mathbb{E}_x \sum_a \langle \psi | \mathbf{1} \otimes B_a^x | \psi \rangle \geq \mathbb{E}_x \sum_a \langle \psi | \mathbf{1} \otimes (B_a^x)^2 | \psi \rangle \geq 0,$$

which implies that

$$\left| \mathbb{E}_x \sum_a \langle \psi | A_a^x \otimes \mathbf{1} | \psi \rangle - \langle \psi | \mathbf{1} \otimes B_a^x | \psi \rangle \right| \leq \left| \mathbb{E}_x \sum_a \langle \psi | A_a^x \otimes \mathbf{1} | \psi \rangle - \langle \psi | \mathbf{1} \otimes (B_a^x)^2 | \psi \rangle \right|.$$

We can bound the second quantity in two steps.

$$\begin{aligned} & \left| \mathbb{E}_x \sum_a \langle \psi | A_a^x \otimes \mathbf{1} | \psi \rangle - \langle \psi | A_a^x \otimes B_a^x | \psi \rangle \right| \\ & \leq \sqrt{\mathbb{E}_x \sum_a \|A_a^x | \psi \rangle\|^2} \sqrt{\mathbb{E}_x \sum_a \|(A_a^x \otimes \mathbf{1} - \mathbf{1} \otimes B_a^x) | \psi \rangle\|^2} \leq \sqrt{\delta}, \end{aligned}$$

and similarly

$$\left| \mathbb{E}_x \sum_a \langle \psi | A_a^x \otimes B_a^x | \psi \rangle - \langle \psi | \mathbf{1} \otimes (B_a^x)^2 | \psi \rangle \right| \leq \sqrt{\delta}.$$

By the triangle inequality, the second quantity is at most $2\sqrt{\delta}$. So is the first one. ◀

► **Lemma 71** (Fact 4.14 of [39]). *Suppose $\{A_a^x\}$ and $\{B_a^x\}$ are two measurements such that $A_a^x \otimes \mathbf{1} \approx_\delta \mathbf{1} \otimes B_a^x$. Suppose that either A or B is a projective measurement and the other is a POVM measurement. Then $A_a^x \otimes \mathbf{1} \simeq_{\sqrt{\delta}} \mathbf{1} \otimes B_a^x$.*

► **Lemma 72** (Proposition 4.26 of [29]). *Let $\{C_{a,b}^x\} \subseteq \mathcal{L}(\mathcal{H})$ be a set of matrices such that $\sum_b (C_{a,b}^x)^\dagger C_{a,b}^x \leq \mathbf{1}$ for all x and a . Then*

$$A_a^x \approx_\delta B_a^x \quad \text{implies that} \quad C_{a,b}^x A_a^x \approx_\delta C_{a,b}^x B_a^x.$$

► **Lemma 73** (Proposition 4.28 of [29]). *Suppose $A_i = \{(A_i)_a^x\}$ be a set of matrices such that $(A_i)_a^x \approx_{\delta_i} (A_{i+1})_a^x$ for $i \in [k+1]$. Then*

$$(A_1)_a^x \approx_{k(\delta_1 + \dots + \delta_k)} (A_{k+1})_a^x.$$

► **Lemma 74** (Fact 4.33 of [39]). *Let $k \geq 0$ be a constant. Let $\{A_{a_1, \dots, a_k}^x\}$ be a projective measurement. For $1 \leq j \leq k$, let $\{(B_j)_{a_j}^x\}$ be a projective measurement, and suppose that*

$$A_{a_j}^x \otimes \mathbb{1} \approx_\delta \mathbb{1} \otimes (B_j)_{a_j}^x.$$

Define the POVM measurement $\{J_{a_1, \dots, a_k}^x\}$ as

$$J_{a_1, \dots, a_k}^x = (B_k)_{a_k}^x \cdots (B_2)_{a_2}^x (B_1)_{a_1}^x (B_2)_{a_2}^x \cdots (B_k)_{a_k}^x.$$

Then

$$A_{a_1, \dots, a_k}^x \otimes \mathbb{1} \approx_{(2k-1)^2 \delta} \mathbb{1} \otimes J_{a_1, \dots, a_k}^x.$$

Proof of Lemma 74. We start with

$$A_{a_1, \dots, a_k}^x = A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_k}^x.$$

Because $A_{a_k}^x \otimes \mathbb{1} \approx_\delta \mathbb{1} \otimes (B_k)_{a_k}^x$, To apply Lemma 72, we can set $C_{a,b}^x = A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_{k-1}}^x \otimes \mathbb{1}$ with $a = a_k$ and $b = (a_1, \dots, a_{k-1})$. Then $\sum_b (C_{a,b}^x)^\dagger C_{a,b}^x \leq \mathbb{1}$. Hence by Lemma 72

$$A_{a_1, \dots, a_k}^x \otimes \mathbb{1} \approx_\delta A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_{k-1}}^x \otimes (B_k)_{a_k}^x$$

We can apply Lemma 72 again with $C_{a,b}^x = A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_{k-2}}^x \otimes B_k^{(a_k)}$ with $a = a_{k-1}$ and $b = (a_1, \dots, a_{k-2}, a_k)$. Because $A_{a_{k-1}}^x \otimes \mathbb{1} \approx_\delta \mathbb{1} \otimes (B_{k-1})_{a_{k-1}}^{(a_{k-1})}$, we can get that

$$A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_{k-1}}^x \otimes (B_k)_{a_k}^x \approx_\delta A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x A_{a_2}^x \cdots A_{a_{k-2}}^x \otimes (B_k)_{a_k}^x (B_{k-1})_{a_{k-1}}^{(a_{k-1})}.$$

Continuing similarly, we can get that

$$A_{a_k}^x \cdots A_{a_2}^x A_{a_1}^x \otimes (B_k)_{a_k}^x \cdots (B_2)_{a_2}^x \approx_\delta A_{a_k}^x \cdots A_{a_2}^x \otimes (B_k)_{a_k}^x \cdots (B_1)_{a_1}^x.$$

With another $(k-2)$ steps we can get that

$$A_{a_k}^x \otimes (B_k)_{a_k}^x \cdots (B_2)_{a_2}^x (B_1)_{a_1}^x (B_2)_{a_2}^x \cdots (B_{k-1})_{a_{k-1}}^x \approx_\delta \mathbb{1} \otimes (B_k)_{a_k}^x \cdots (B_2)_{a_2}^x (B_1)_{a_1}^x (B_2)_{a_2}^x \cdots (B_k)_{a_k}^x.$$

Combining all the steps above with Lemma 73

$$A_{a_1, \dots, a_k}^x \otimes \mathbb{1} \approx_{(2k-1)^2 \delta} \mathbb{1} \otimes (B_k)_{a_k}^x \cdots (B_2)_{a_2}^x (B_1)_{a_1}^x (B_2)_{a_2}^x \cdots (B_k)_{a_k}^x,$$

which completes the proof. ◀

► **Lemma 75** (Fact 4.35 of [39]). *Let $k \geq 0$ be a constant. Let D be a distribution on questions (x, y_1, \dots, y_k) , where each $y_i \in \mathcal{Y}_i$. For each $1 \leq i \leq k$, let \mathcal{G}_i be a set of functions $g_i : \mathcal{Y}_i \rightarrow \mathcal{R}_i$, and let $\{(G_i)_g^x \mid g \in \mathcal{G}_i\}$ be a projective measurement. Suppose that the set \mathcal{G}_i has the following distance property: fix a question $z = (x, y_1, \dots, y_{i-1}, y_{i+1}, \dots, y_k)$, and let D_z be the distribution on y_i conditioned on z . Then for any two nonequal $g_i, g'_i \in \mathcal{G}_i$, the probability that $g_i(\mathbf{y}_i) = g'_i(\mathbf{y}_i)$, over a random $\mathbf{y}_i \sim D_z$, is at most ε .*

Let $\{A_{a_1, \dots, a_k}^{x, y_1, \dots, y_k}\}$ be a projective measurement with outcomes $a_i \in \mathcal{R}_i$. For each $1 \leq i \leq k$, suppose that

$$A_{a_i}^{x, y_1, \dots, y_k} \otimes \mathbb{1} \simeq_\delta \mathbb{1} \otimes (G_i)_{[g_i(\mathbf{y}_i)=a_i]}^x \quad (10)$$

$$(G_i)_{[g_i(\mathbf{y}_i)=a_i]}^x \otimes \mathbb{1} \simeq_\delta \mathbb{1} \otimes A_{a_i}^{x, y_1, \dots, y_k}. \quad (11)$$

30:34 The Computational Advantage of MIP* Vanishes in the Presence of Noise

Also suppose that

$$A_{a_i}^{x,y_1,\dots,y_k} \otimes \mathbb{1} \simeq_\delta \mathbb{1} \otimes A_{a_i}^{x,y_1,\dots,y_k}. \quad (12)$$

Define the POVM $\{J_{g_1,\dots,g_k}^x\}$ as

$$J_{g_1,\dots,g_k}^x := (G_k)_{g_k}^x \cdots (G_2)_{g_2}^x \cdot (G_1)_{g_1}^x \cdot (G_2)_{g_2}^x \cdots (G_k)_{g_k}^x.$$

Then

$$A_{a_1,\dots,a_k}^{x,y_1,\dots,y_k} \otimes \mathbb{1} \approx_{\text{poly}(\exp(k), \delta^{1/4k}, \varepsilon^{1/2k})} \mathbb{1} \otimes J_{[g_1(y_1),\dots,g_k(y_k)=a_1,\dots,a_k]}^x.$$

This proof is the same as the original one, but we rewrite it to keep better track of the approximation errors.

The original proof. We first show the $k = 2$ case. Notice that

$$J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2} = \sum_{g_2:g_2(y_2)=a_2} (G_2)_{g_2}^x \left(\sum_{g_1:g_1(y_1)=a_1} (G_1)_{g_1}^x \right) (G_2)_{g_2}^x.$$

Our goal is to bound

$$\begin{aligned} & \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2} | \psi \rangle \\ &= \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes \sum_{g_2:g_2(y_2)=a_2} (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x | \psi \rangle \\ &= \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x | \psi \rangle. \end{aligned}$$

First notice that

$$\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \approx_{2\sqrt{2\delta}} \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes \mathbb{1} | \psi \rangle = 1.$$

This is because

$$\begin{aligned} & \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x | \psi \rangle - \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \right| \\ &= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) | \psi \rangle \right| \\ &\leq \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \|A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x | \psi\|^2} \\ &\quad \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) A_{a_1,g_2(y_2)}^{x,y_1,y_2} (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) | \psi \rangle} \\ &\leq \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \|A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x | \psi\|^2} \\ &\quad \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1} \langle \psi | (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) \sum_{g_2} A_{a_1,g_2(y_2)}^{x,y_1,y_2} (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) | \psi \rangle} \\ &\leq 1 \cdot \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1} \| (A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} - \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x) | \psi \|^2} \\ &\leq \sqrt{2\delta} \end{aligned}$$

and

$$\begin{aligned}
& \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x | \psi \rangle - \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes \mathbf{1} | \psi \rangle \right| \\
&= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \cdot (\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) | \psi \rangle \right| \\
&\leq \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \|A_{a_1,a_2}^{x,y_1,y_2} | \psi \rangle\|^2} \\
&\sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | (\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) A_{a_1,a_2}^{x,y_1,y_2} (\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) | \psi \rangle} \\
&\leq \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \|A_{a_1,a_2}^{x,y_1,y_2} | \psi \rangle\|^2} \\
&\sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_2} \langle \psi | (\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) \sum_{a_1} A_{a_1,a_2}^{x,y_1,y_2} (\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) | \psi \rangle} \\
&\leq 1 \cdot \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_2} \|(\mathbf{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x - A_{a_2}^{x,y_1,y_2} \otimes \mathbf{1}) | \psi \rangle\|^2} \\
&\leq \sqrt{2\delta},
\end{aligned}$$

Hence, we focus on proving

$$\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \|\mathbf{1} \otimes \left((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x - (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x \right) | \psi \rangle\|^2 \leq C_1 \sqrt{\delta} + C_2 \varepsilon \quad (13)$$

for some constants C_1 and C_2 , which will imply that

$$\begin{aligned}
& \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x \left((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x - (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x \right) | \psi \rangle \right| \\
&\leq \sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \|A_{a_1,g_2(y_2)}^{x,y_1,y_2} \otimes (G_2)_{g_2}^x | \psi \rangle\|^2} \\
&\sqrt{\mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \|\langle \psi | \mathbf{1} \otimes \left((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x - (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x \right) | \psi \rangle\|^2} \\
&\leq \sqrt{C_1 \sqrt{\delta} + C_2 \varepsilon}
\end{aligned}$$

and

$$\left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2} | \psi \rangle - 1 \right| \leq 2\sqrt{2\delta} + \sqrt{C_1 \sqrt{\delta} + C_2 \varepsilon}.$$

To prove Equation (13), we start with Equation (10)

$$\mathbb{E}_{x,y_1,y_2} \sum_{a_i} \| (A_{a_i}^{x,y_1,y_2} \otimes \mathbf{1} - \mathbf{1} \otimes (G_i)_{[g_i(y_i)=a_i]}^x) | \psi \rangle \|^2 \leq 2\delta$$

for $i = 1, 2$. Then by Lemma 72

$$\begin{aligned}
 & \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \\
 & \approx_{2\delta} A_{a_2}^{x,y_1,y_2} \otimes (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle \\
 & \approx_{2\delta} A_{a_2}^{x,y_1,y_2} A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} |\psi\rangle \\
 & = A_{a_1}^{x,y_1,y_2} A_{a_1}^{x,y_1,y_2} \otimes \mathbb{1} |\psi\rangle \\
 & \approx_{2\delta} A_{a_2}^{x,y_1,y_2} \otimes (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \\
 & \approx_{2\delta} \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle.
 \end{aligned}$$

Chaining the inequalities together using Lemma 73 gives

$$\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \|\mathbb{1} \otimes ((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x - (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x) |\psi\rangle\|^2 \leq 32\delta.$$

Let

$$\begin{aligned}
 S_1 &= \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g} \|\mathbb{1} \otimes ((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x - (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x) |\psi\rangle\|^2 \\
 S_2 &= \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \|\mathbb{1} \otimes ((G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x - (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x) |\psi\rangle\|^2.
 \end{aligned}$$

We are going to show that S_1 is close to S_2 . Expanding $S_1 - S_2$, we get $|S_1 - S_2| \leq \Delta_1 + \Delta_2 + \Delta_3 + \Delta_4$, where

$$\begin{aligned}
 \Delta_1 &= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | \mathbb{1} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x |\psi\rangle \right. \\
 & \quad \left. - \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \right| \\
 \Delta_2 &= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle \right. \\
 & \quad \left. - \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle \right| \\
 \Delta_3 &= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | \mathbb{1} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle \right. \\
 & \quad \left. - \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x |\psi\rangle \right| \\
 \Delta_4 &= \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x |\psi\rangle \right. \\
 & \quad \left. - \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \right|.
 \end{aligned}$$

First of all

$$\Delta_1 = \left| 1 - \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \right|.$$

By Equation (11),

$$\mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x |\psi\rangle \approx_{18\delta} A_{a_1,a_2}^{x,y_1,y_2} \otimes \mathbb{1} |\psi\rangle,$$

then Lemma 70 implies that

$$|\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes \mathbb{1} | \psi \rangle - \langle \psi | \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x | \psi \rangle | \leq 6\sqrt{2\delta}.$$

Since $\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes A_{a_1,a_2}^{x,y_1,y_2} | \psi \rangle = 1$, $\Delta_1 \leq 6\sqrt{2\delta}$. Next, observe that $\Delta_2 = 0$ as $(G_2)_{g_2}^x$ and $(G_2)_{[g_2(y_2)=a_2]}^x$ are projective measurements. Lastly, observe that $\Delta_3 = \Delta_4$, so we focus on bounding Δ_3 . First notice that

$$\begin{aligned} & \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | \mathbb{1} \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \\ & \approx_{3\sqrt{2\delta}} \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | (G_1)_{[g_1(y_1)=a_1]}^x \otimes (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x | \psi \rangle \\ & \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | \mathbb{1} \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \\ & \approx_{3\sqrt{2\delta}} \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | (G_1)_{[g_1(y_1)=a_1]}^x \otimes (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x | \psi \rangle \end{aligned}$$

The reason why $\mathbb{1} \otimes (G_1)_{[g_1(y_1)=a_1]}^x \approx_{18\delta} (G_1)_{[g_1(y_1)=a_1]}^x \otimes \mathbb{1}$ is the following. Applying Lemma 69 to Equations (10) and (11) we get

$$\begin{aligned} & \mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_i} \| (A_{a_i}^{x,y_1,\dots,y_k} \otimes \mathbb{1} - \mathbb{1} \otimes (G_i)_{[g_i(y_i)=a_i]}^x) | \psi \rangle \|^2 \leq 2\delta \\ & \mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_i} \| ((G_i)_{[g_i(y_i)=a_i]}^x \otimes \mathbb{1} - \mathbb{1} \otimes A_{a_i}^{x,y_1,\dots,y_k}) | \psi \rangle \|^2 \leq 2\delta. \end{aligned}$$

Notice that for any $i \in [k]$,

$$\begin{aligned} & \mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_i} \langle \psi | A_{a_i}^{x,y_1,\dots,y_k} \otimes A_{a_i}^{x,y_1,\dots,y_k} | \psi \rangle \\ & \geq \mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_1,\dots,a_k} \langle \psi | A_{a_1,\dots,a_k}^{x,y_1,\dots,y_k} \otimes A_{a_1,\dots,a_k}^{x,y_1,\dots,y_k} | \psi \rangle \geq 1 - \delta \end{aligned}$$

because $A_{a_1,\dots,a_k}^{x,y_1,\dots,y_k} \otimes A_{b_1,\dots,b_k}^{x,y_1,\dots,y_k} \geq 0$ for any $a_1, \dots, a_k, b_1, \dots, b_k$. Then Lemma 69 also implies that

$$\mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_1} \| (A_{a_1}^{x,y_1,\dots,y_k} \otimes \mathbb{1} - \mathbb{1} \otimes A_{a_1}^{x,y_1,\dots,y_k}) | \psi \rangle \|^2 \leq 2\delta.$$

Hence, Lemma 73 implies that for all $i \in [k]$,

$$\mathbb{E}_{x,y_1,\dots,y_k} \sum_{a_i} \| ((G_i)_{[g_i(y_i)=a_i]}^x \otimes \mathbb{1} - \mathbb{1} \otimes (G_i)_{[g_i(y_i)=a_i]}^x) | \psi \rangle \|^2 \leq 18\delta.$$

Also, notice that

$$\begin{aligned} & |\mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | (G_2)_{[g_2(y_2)=a_2]}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{[g_2(y_2)=a_2]}^x \otimes (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \\ & - \mathbb{E}_{x,y_1,y_2} \sum_{a_1,g_2} \langle \psi | (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2}^x \otimes (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle | \\ & = |\mathbb{E}_{x,y_1,y_2} \sum_{a_1} \sum_{g_2,g_2'} \langle \psi | (G_2)_{g_2}^x (G_1)_{[g_1(y_1)=a_1]}^x (G_2)_{g_2'}^x \otimes (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle \mathbb{1}[g_2(y_2) = g_2'(y_2)]| \\ & \leq \varepsilon |\mathbb{E}_{x,y_1} \sum_{a_1} \langle \psi | (G_1)_{[g_1(y_1)=a_1]}^x \otimes (G_1)_{[g_1(y_1)=a_1]}^x | \psi \rangle | \\ & \leq \varepsilon. \end{aligned}$$

30:38 The Computational Advantage of MIP* Vanishes in the Presence of Noise

Therefore, $\Delta_3 = \Delta_4 \leq 6\sqrt{2\delta} + \varepsilon$, and

$$|S_1 - S_2| \leq \sum_{j=1}^4 \Delta_j \leq 18\sqrt{2\delta} + 2\varepsilon,$$

and

$$S_1 \leq 32\delta + 18\sqrt{2\delta} + 2\varepsilon.$$

In conclusion,

$$\begin{aligned} & \left| \mathbb{E}_{x,y_1,y_2} \sum_{a_1,a_2} \langle \psi | A_{a_1,a_2}^{x,y_1,y_2} \otimes J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2} | \psi \rangle - 1 \right| \\ & \leq 2\sqrt{2\delta} + \sqrt{32\delta + 18\sqrt{2\delta} + 2\varepsilon} \leq 11\delta^{1/4} + 2\sqrt{\varepsilon}, \end{aligned}$$

and equivalently

$$A_{a_1,a_2}^{x,y_1,y_2} \otimes \mathbb{1} \approx_{22\delta^{1/4} + 4\sqrt{\varepsilon}} \mathbb{1} \otimes J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2}.$$

Switching the roles of Alice and Bob, the same proof gives us that

$$J_{[g_1(y_1),g_2(y_2)=a_1,a_2]}^{x,y_1,y_2} \otimes \mathbb{1} \approx_{22\delta^{1/4} + 4\sqrt{\varepsilon}} \mathbb{1} \otimes A_{a_1,a_2}^{x,y_1,y_2}.$$

For the general case, assume

$$\begin{aligned} & A_{a_1,\dots,a_i}^{x,y_1,\dots,y_i} \otimes \mathbb{1} \approx_{f(i,\delta,\varepsilon)} \mathbb{1} \otimes J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^x \text{ and} \\ & \mathbb{1} \otimes A_{a_1,\dots,a_i}^{x,y_1,\dots,y_i} \approx_{f(i,\delta,\varepsilon)} J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^x \otimes \mathbb{1}, \end{aligned}$$

which imply that

$$\mathbb{1} \otimes J_{[g_1(y_1),\dots,g_i(y_i)]}^x \approx_{3(2\delta+2f(i,\delta,\varepsilon))} J_{[g_1(y_1),\dots,g_i(y_i)]}^x \otimes \mathbb{1}.$$

Since δ and ε are fixed, we write $f(i, \delta, \varepsilon)$ as $f(i)$ in the rest of the proof and proceed to the $i+1$ case. As in the base case, our goal is to bound

$$\begin{aligned} & \mathbb{E}_{x,y_1,\dots,y_{i+1}} \sum_{a_1,\dots,a_{i+1}} \langle \psi | A_{a_1,\dots,a_{i+1}}^{x,y_1,\dots,y_{i+1}} \otimes J_{[g_1(y_1),\dots,g_{i+1}(y_{i+1})=a_1,\dots,a_{i+1}]}^{x,y_1,\dots,y_{i+1}} | \psi \rangle \\ & = \mathbb{E}_{x,y_1,\dots,y_{i+1}} \sum_{a_1,\dots,a_i,g_{i+1}} \langle \psi | A_{a_1,\dots,a_i,g_{i+1}(y_{i+1})}^{x,y_1,\dots,y_{i+1}} \otimes (G_{i+1})_{g_{i+1}}^x J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^{x,y_1,\dots,y_i} (G_{i+1})_{g_{i+1}}^x | \psi \rangle. \end{aligned}$$

by relating it to

$$\begin{aligned} & \mathbb{E}_{x,y_1,\dots,y_{i+1}} \sum_{a_1,\dots,a_i,g_{i+1}} \langle \psi | A_{a_1,\dots,a_i,g_{i+1}(y_{i+1})}^{x,y_1,\dots,y_{i+1}} \otimes (G_{i+1})_{g_{i+1}}^x J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^{x,y_1,\dots,y_i} | \psi \rangle \\ & \approx_{\sqrt{2\delta} + \sqrt{f(i)}} \mathbb{E}_{x,y_1,\dots,y_{i+1}} \sum_{a_1,\dots,a_{i+1}} \langle \psi | A_{a_1,\dots,a_{i+1}}^{x,y_1,\dots,y_{i+1}} \otimes \mathbb{1} | \psi \rangle = 1. \end{aligned}$$

So the central step is bounding

$$\begin{aligned} & \mathbb{E}_{x,y_1,\dots,y_{i+1}} \sum_{a_1,\dots,a_i,g_{i+1}} \left\| \mathbb{1} \otimes \left(J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^{x,y_1,\dots,y_i} (G_{i+1})_{g_{i+1}}^x \right. \right. \\ & \quad \left. \left. - (G_{i+1})_{g_{i+1}}^x J_{[g_1(y_1),\dots,g_i(y_i)=a_1,\dots,a_i]}^{x,y_1,\dots,y_i} \right) | \psi \right\|^2. \end{aligned}$$

As in the base case, we can use similar arguments to show

$$\begin{aligned} & \mathbb{E}_{x, y_1, \dots, y_{i+1}} \sum_{a_1, \dots, a_i, g_{i+1}} \|\mathbb{1} \otimes \left(J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} (G_{i+1})_{[g_{i+1}(y_{i+1})=a_{i+1}]}^x \right. \\ & \quad \left. - (G_{i+1})_{[g_{i+1}(y_{i+1})=a_{i+1}]}^x J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} \right) |\psi\rangle\|^2 \\ & \leq 4(2f(i) + 4\delta), \end{aligned}$$

and

$$\begin{aligned} & \left| \mathbb{E}_{x, y_1, \dots, y_{i+1}} \sum_{a_1, \dots, a_i, g_{i+1}} \|\mathbb{1} \otimes \left(J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} (G_{i+1})_{g_{i+1}}^x \right. \right. \\ & \quad \left. \left. - (G_{i+1})_{g_{i+1}}^x J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} \right) |\psi\rangle\|^2 - \right. \\ & \quad \left. \mathbb{E}_{x, y_1, \dots, y_{i+1}} \sum_{a_1, \dots, a_i, g_{i+1}} \|\mathbb{1} \otimes \left(J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} (G_{i+1})_{[g_{i+1}(y_{i+1})=a_{i+1}]}^x \right. \right. \\ & \quad \left. \left. - (G_{i+1})_{[g_{i+1}(y_{i+1})=a_{i+1}]}^x J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} \right) |\psi\rangle\|^2 \right| \\ & \leq 2\sqrt{2f(i) + 4\delta} + 2\sqrt{6f(i) + 4\delta} + 2\varepsilon. \end{aligned}$$

Therefore,

$$\begin{aligned} & \mathbb{E}_{x, y_1, \dots, y_{i+1}} \sum_{a_1, \dots, a_i, g_{i+1}} \|\mathbb{1} \otimes \left(J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} (G_{i+1})_{g_{i+1}}^x \right. \\ & \quad \left. - (G_{i+1})_{g_{i+1}}^x J_{[g_1(y_1), \dots, g_i(y_i)=a_1, \dots, a_i]}^{x, y_1, \dots, y_i} \right) |\psi\rangle\|^2 \\ & \leq 4(2f(i) + 4\delta) + 2\sqrt{2f(i) + 4\delta} + 2\sqrt{6f(i) + 4\delta} + 2\varepsilon, \end{aligned}$$

and

$$\begin{aligned} & \left| \mathbb{E}_{x, y_1, \dots, y_{i+1}} \sum_{a_1, \dots, a_{i+1}} \langle \psi | A_{a_1, \dots, a_{i+1}}^{x, y_1, \dots, y_{i+1}} \otimes J_{[g_1(y_1), \dots, g_{i+1}(y_{i+1})=a_1, \dots, a_{i+1}]}^{x, y_1, \dots, y_{i+1}} |\psi\rangle - 1 \right| \\ & \leq \sqrt{2\delta} + \sqrt{f(i)} + \sqrt{16\sqrt{f(i)} + 24\sqrt{\delta} + 2\varepsilon} \end{aligned}$$

That is $f(i+1) = 5f(i)^{1/4} + 7\delta^{1/4} + \sqrt{2\varepsilon}$. Then the lemma follows. \blacktriangleleft

B Proofs of Theorems

B.1 Invariance Principle for Matrix Spaces

► **Fact 76** ([37, Remark 3.10]). *If \mathbf{x} is (p, q, η) -hypercontractive, then it is (p, q, η') -hypercontractive for any $0 < \eta' \leq \eta$.*

► **Lemma 77.** *Given $m, n \in \mathbb{Z}_{>0}$, $0 < \eta < 1$, a $(2, 4, \eta)$ -hypercontractive (m, n) ensemble \mathbf{x} , it holds that*

$$\mathbb{E} \left[\left(\sum_{i=1}^k (T_\eta p_i)(\mathbf{x})^2 \right)^2 \right] \leq \left(\mathbb{E} \left[\sum_{i=1}^k p_i(\mathbf{x})^2 \right] \right)^2,$$

for any multilinear polynomials p_1, \dots, p_k .

Proof of Lemma 77. Let $q_i = T_\eta p_i$. Then

$$\begin{aligned}
 \mathbb{E} \left[\left(\sum_{i=1}^k (T_\eta p_i)(\mathbf{x}) \right)^2 \right] &= \sum_{i,j} \mathbb{E} \left[q_i(\mathbf{x})^2 q_j(\mathbf{x})^2 \right] \\
 &\leq \sum_{i,j} \|q_i\|_4^2 \|q_j\|_4^2 \quad (\text{Cauchy-Schwarz inequality}) \\
 &\leq \sum_{i,j} \|p_i\|_2^2 \|p_j\|_2^2 \quad (\mathbf{x} \text{ is } (2, 4, \eta)\text{-hypercontractive}) \\
 &= \left(\sum_i \|p_i\|_2^2 \right)^2 = \left(\mathbb{E} \left[\sum_{i=1}^k p_i(\mathbf{x})^2 \right] \right)^2. \quad \blacktriangleleft
 \end{aligned}$$

The lemma below follows directly from Definition 10 and Fact 48.

► **Lemma 78.** Given $0 \leq \gamma \leq 1$, $h, n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, an (m^2, n) ensemble \mathbf{x} , and a random matrix

$$P(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{x}) \mathcal{B}_\sigma,$$

where $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$ is a standard orthonormal basis and p_σ is a real multilinear polynomial for all $\sigma \in [m^2]_{\geq 0}^h$, suppose that for all $\sigma \in [m^2]_{\geq 0}^h$, p_σ has an expansion

$$p_\sigma(\mathbf{x}) = \sum_{\tau \in [m^2]_{\geq 0}^n} \widehat{p}_\sigma(\tau) \mathbf{x}_\tau.$$

It holds that

$$\Gamma_\gamma(P(\mathbf{x})) = \sum_{\sigma \in [m^2]_{\geq 0}^h} \sum_{\tau \in [m^2]_{\geq 0}^n} \gamma^{|\sigma|+|\tau|} \widehat{p}_\sigma(\tau) \mathbf{x}_\tau \mathcal{B}_\sigma. \quad (14)$$

We need the hypercontractivity inequality for Hermitian matrices.

► **Fact 79** ([46, Lemma 8.3]). Given $h, n, m \in \mathbb{Z}_{>0}$, $m \geq 2$, $0 \leq \gamma \leq (9m)^{-1/4}$ and $P \in \mathcal{H}_m^{\otimes n}$, it holds that

$$\|\|\| \Delta_\gamma^{\otimes n}(P) \|\|\|_4 \leq \|P\|_2,$$

where $\Delta_\gamma(\cdot)$ is defined in Definition 47.

Proof of Theorem 13. Set $Q(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^h} (T_\gamma p_\sigma)(\mathbf{x}) \mathcal{B}_\sigma$. Then by the definition of Γ_γ ,

$$\Gamma_\gamma(P(\mathbf{x})) = \Delta_\gamma(Q(\mathbf{x})).$$

Using Fact 79,

$$\mathbb{E} \left[\|\|\| \Delta_\gamma(Q(\mathbf{x})) \|\|\|_4^4 \right] \leq \mathbb{E} \left[\|\|\| Q(\mathbf{x}) \|\|\|_2^4 \right]. \quad (15)$$

Denote $q_\sigma = T_\gamma p_\sigma$. Notice that

$$\begin{aligned}
 \mathbb{E} \left[\|\|\| Q(\mathbf{x}) \|\|\|_2^4 \right] &= m^{-2h} \mathbb{E} \left[\left(\sum_{\sigma \in [m^2]_{\geq 0}^h} q_\sigma(\mathbf{x}) \right)^2 \right]^2 \leq m^{-2h} \left(\mathbb{E} \left[\sum_{\sigma \in [m^2]_{\geq 0}^h} p_\sigma(\mathbf{x})^2 \right] \right)^2 \\
 &= \left(\mathbb{E} \left[\|\|\| P(\mathbf{x}) \|\|\|_2^2 \right] \right)^2,
 \end{aligned}$$

where the inequality follows from Fact 76 and Lemma 77. We conclude the result by combining it with Equation (15). \blacktriangleleft

Proof of Theorem 14. Suppose that for all $\sigma \in [m^2]_{\geq 0}^h$, p_σ has an expansion

$$p_\sigma(\mathbf{x}) = \sum_{\tau \in [m^2]_{\geq 0}^n} \widehat{p}_\sigma(\tau) \mathbf{x}_\tau.$$

Set

$$P^{=i}(\mathbf{x}) = \sum_{\substack{\sigma \in [m^2]_{\geq 0}^h, \tau \in [m^2]_{\geq 0}^n: \\ |\sigma| + |\tau| = i}} \widehat{p}_\sigma(\tau) \mathbf{x}_\tau \mathcal{B}_\sigma.$$

Set $\gamma = \min \left\{ \eta, (9m)^{-1/4} \right\}$. Applying Lemma 78 and Theorem 13,

$$\mathbb{E} \left[\left\| P(\mathbf{x}) \right\|_4^4 \right] = \mathbb{E} \left[\left\| \Gamma_\gamma \left(\sum_{i=1}^d \gamma^{-i} P^{=i}(\mathbf{x}) \right) \right\|_4^4 \right] \leq \left(\mathbb{E} \left[\left\| \sum_{i=1}^d \gamma^{-i} P^{=i}(\mathbf{x}) \right\|_2^2 \right] \right)^2$$

By the orthogonality of \mathbf{x} and \mathcal{B} , if $i \neq j$, we have

$$\mathbb{E} [\text{Tr } P^{=i}(\mathbf{x}) P^{=j}(\mathbf{x})] = 0.$$

Therefore,

$$\begin{aligned} \mathbb{E} \left[\left\| P(\mathbf{x}) \right\|_4^4 \right] &\leq \left(\sum_{i=1}^d \gamma^{-2i} \mathbb{E} \left[\left\| P^{=i}(\mathbf{x}) \right\|_2^2 \right] \right)^2 \\ &\leq \gamma^{-4d} \left(\sum_{i=1}^d \mathbb{E} \left[\left\| P^{=i}(\mathbf{x}) \right\|_2^2 \right] \right)^2 = \gamma^{-4d} \left(\mathbb{E} \left[\left\| P(\mathbf{x}) \right\|_2^2 \right] \right)^2. \quad \blacktriangleleft \end{aligned}$$

Proof of Theorem 15. Without loss of generality, we assume $\overline{H} = [n - h]$. We prove this by a hybrid argument. For any $0 \leq i \leq n - h$, define the hybrid basis elements and the hybrid random operators as follows.

$$\mathcal{X}_\sigma^{(i)} = \mathbf{x}_{\sigma_{\leq i}} \cdot \mathcal{B}_{\sigma_{> i}} \text{ for } \sigma \in [m^2]_{\geq 0}^n; \quad (16)$$

$$P^{(i)}(\mathbf{x}) = \sum_{\sigma \in [m^2]_{\geq 0}^n} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i)}, \quad (17)$$

where $\mathbf{x}_{\sigma_{\leq i}} = \mathbf{x}_{\sigma_1} \cdots \mathbf{x}_{\sigma_i}$ and $\mathcal{B}_{\sigma_{> i}} = \mathcal{B}_{\sigma_{i+1}} \otimes \cdots \otimes \mathcal{B}_{\sigma_n}$. Then $P = P^{(0)}(\mathbf{x})$ and $P^H(\mathbf{x}) = P^{(n-h)}(\mathbf{x})$. Note that

$$\begin{aligned} P^{(i)}(\mathbf{x}) &= \sum_{\sigma: \sigma_{i+1}=0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i)} + \sum_{\sigma: \sigma_{i+1} \neq 0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i)}, \\ P^{(i+1)}(\mathbf{x}) &= \sum_{\sigma: \sigma_{i+1}=0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i+1)} + \sum_{\sigma: \sigma_{i+1} \neq 0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i+1)}, \end{aligned}$$

Set

$$\begin{aligned} \mathbf{A} &= \sum_{\sigma: \sigma_{i+1}=0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i)}; & \mathbf{B} &= \sum_{\sigma: \sigma_{i+1} \neq 0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i)}; \\ \mathbf{C} &= \sum_{\sigma: \sigma_{i+1}=0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i+1)}; & \mathbf{D} &= \sum_{\sigma: \sigma_{i+1} \neq 0} \widehat{P}(\sigma) \mathcal{X}_\sigma^{(i+1)}. \end{aligned}$$

30:42 The Computational Advantage of MIP* Vanishes in the Presence of Noise

Then we have

$$P^{(i)}(\mathbf{x}) = \mathbf{A} + \mathbf{B}; \quad P^{(i+1)}(\mathbf{x}) = \mathbf{C} + \mathbf{D}.$$

Notice that $\mathbf{A} = \mathbf{1}_m \otimes \mathbf{C}$, where $\mathbf{1}_m$ is placed in the $(i+1)$ -th register. Thus,

$$\mathrm{Tr} \xi(\mathbf{A}) = m \cdot \mathrm{Tr} \xi(\mathbf{C}). \quad (18)$$

From Fact 54 and then Equation (18),

$$\begin{aligned} & \left| m^{i+1-n} \mathbb{E} \left[\mathrm{Tr} \xi \left(P^{(i+1)}(\mathbf{x}) \right) \right] - m^{i-n} \mathbb{E} \left[\mathrm{Tr} \xi \left(P^{(i)}(\mathbf{x}) \right) \right] \right| \\ &= \left| \mathbb{E} \left[\begin{aligned} & m^{i+1-n} \left(\mathrm{Tr} \xi(\mathbf{C}) + \mathrm{Tr} D\xi(\mathbf{C})[\mathbf{D}] + \frac{1}{2} \mathrm{Tr} D^2\xi(\mathbf{C})[\mathbf{D}] + \Delta_{3,\xi}(\mathbf{C}, \mathbf{D}) \right) - \\ & m^{i-n} \left(\mathrm{Tr} \xi(\mathbf{A}) + \mathrm{Tr} D\xi(\mathbf{A})[\mathbf{B}] + \frac{1}{2} \mathrm{Tr} D^2\xi(\mathbf{A})[\mathbf{B}] + \Delta_{3,\xi}(\mathbf{A}, \mathbf{B}) \right) \end{aligned} \right] \right| \\ &= \left| \mathbb{E} \left[\begin{aligned} & m^{i+1-n} \left(\mathrm{Tr} D\xi(\mathbf{C})[\mathbf{D}] + \frac{1}{2} \mathrm{Tr} D^2\xi(\mathbf{C})[\mathbf{D}] + \Delta_{3,\xi}(\mathbf{C}, \mathbf{D}) \right) - \\ & m^{i-n} \left(\mathrm{Tr} D\xi(\mathbf{A})[\mathbf{B}] + \frac{1}{2} \mathrm{Tr} D^2\xi(\mathbf{A})[\mathbf{B}] + \Delta_{3,\xi}(\mathbf{A}, \mathbf{B}) \right) \end{aligned} \right] \right| \end{aligned}$$

Both the first-order and second-order derivatives cancel out because of the following claim.

▷ **Claim 80.** It holds that

$$\mathbb{E}[\mathrm{Tr} D\xi(\mathbf{A})[\mathbf{B}]] = m \mathbb{E}[\mathrm{Tr} D\xi(\mathbf{C})[\mathbf{D}]];$$

$$\mathbb{E}[\mathrm{Tr} D^2\xi(\mathbf{A})[\mathbf{B}]] = m \mathbb{E}[\mathrm{Tr} D^2\xi(\mathbf{C})[\mathbf{D}]].$$

By Fact 54, there exists a universal constant $c_3 > 0$ such that

$$\begin{aligned} & \left| \mathbb{E} \left[m^{i+1-n} \mathrm{Tr} \xi \left(P^{(i+1)}(\mathbf{x}) \right) - m^{i-n} \mathrm{Tr} \xi \left(P^{(i)}(\mathbf{x}) \right) \right] \right| \\ &\leq c_3 B \left(\mathbb{E} \left[\|\mathbf{B}\|_3^3 \right] + \mathbb{E} \left[\|\mathbf{D}\|_3^3 \right] \right) \\ &\leq c_3 B \left(\mathbb{E} \left[\|\mathbf{B}\|_2 \|\mathbf{B}\|_4^2 \right] + \mathbb{E} \left[\|\mathbf{D}\|_2 \|\mathbf{D}\|_4^2 \right] \right) \quad (\text{H\"older's}) \\ &\leq c_3 B \left(\left(\mathbb{E} \left[\|\mathbf{B}\|_2^2 \right] \mathbb{E} \left[\|\mathbf{B}\|_4^4 \right] \right)^{1/2} + \left(\mathbb{E} \left[\|\mathbf{D}\|_2^2 \right] \mathbb{E} \left[\|\mathbf{D}\|_4^4 \right] \right)^{1/2} \right) \quad (\text{Cauchy-Schwartz}) \\ &\leq c_3 B \theta^d \left(\left(\mathbb{E} \left[\|\mathbf{B}\|_2^2 \right] \right)^{3/2} + \left(\mathbb{E} \left[\|\mathbf{D}\|_2^2 \right] \right)^{3/2} \right) \quad (\text{Theorem 14}), \end{aligned}$$

where $\theta = \max \{9m, 1/\eta^4\}$. Notice that

$$\mathbb{E} \left[\|\mathbf{B}\|_2^2 \right] = \mathbb{E} \left[\|\mathbf{D}\|_2^2 \right] = \sum_{\sigma: \sigma_{i+1} \neq 0} \left| \widehat{P}(\sigma)^2 \right| = \mathrm{Inf}_{i+1}(P).$$

Therefore,

$$\left| \mathbb{E} \left[m^{i+1-n} \mathrm{Tr} \xi \left(P^{(i+1)}(\mathbf{x}) \right) - m^{i-n} \mathrm{Tr} \xi \left(P^{(i)}(\mathbf{x}) \right) \right] \right| \leq 2c_3 B \theta^d \mathrm{Inf}_{i+1}(P)^{3/2}.$$

Summing over $i \in [n-h]_{\geq 0}$, we have

$$\begin{aligned} & \left| m^{-n} \mathrm{Tr} \xi(P) - m^{-h} \mathbb{E} \left[\mathrm{Tr} \xi(P^H(\mathbf{x})) \right] \right| \\ &\leq 2c_3 B \theta^d \sum_{i \notin H} \mathrm{Inf}_i(P)^{3/2} \\ &\leq 2c_3 B \theta^d \sqrt{\tau} \sum_{i \notin H} \mathrm{Inf}_i(P) \\ &\leq 2c_3 B \theta^d \sqrt{\tau} d \sum_{\sigma \neq 0} \widehat{P}(\sigma)^2 \\ &\leq 2c_3 B \theta^d \sqrt{\tau} d. \end{aligned}$$

◀

Proof of Claim 80. Note that $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and \mathbf{D} can be expressed as

$$\mathbf{A} = \mathbb{1}_m \otimes \mathbf{C}; \quad \mathbf{B} = \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathcal{B}_\sigma \otimes \mathbf{X}_\sigma; \quad \mathbf{D} = \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathbf{x}_{i+1, \sigma} \mathbf{X}_\sigma$$

for some random matrices \mathbf{X}_σ 's which are independent of $\mathbf{x}_{i+1, \sigma}$'s, where $\mathbb{1}_m$ and \mathbf{B}_σ 's are in the $(i+1)$ -th register.

Suppose that \mathbf{C} has a spectral decomposition

$$\mathbf{C} = \sum_{j=1}^{m'} \mathbf{a}_j \mathbf{\Pi}_j,$$

where m' is the dimension of \mathbf{C} , $\mathbf{a}_1 \geq \dots \geq \mathbf{a}_{m'}$, $\{\mathbf{\Pi}_i\}_{i \in [m']}$ are rank-one projectors satisfying that $\sum_{i=1}^{m'} \mathbf{\Pi}_i = \mathbb{1}$ and $\mathbf{\Pi}_i \mathbf{\Pi}_j = 0$ for all $i \neq j$.

By Fact 53, we have

$$\begin{aligned} & \mathbb{E}[\text{Tr } D\xi(\mathbf{A})[\mathbf{B}]] \\ &= \sum_{j, k \in [m']} \mathbb{E} \left[\xi^{[1]}(\mathbf{a}_j, \mathbf{a}_k) \text{Tr}((\mathbb{1} \otimes \mathbf{\Pi}_j) \mathbf{B} (\mathbb{1} \otimes \mathbf{\Pi}_k)) \right] \\ &= \sum_{j, k \in [m']} \mathbb{E} \left[\xi^{[1]}(\mathbf{a}_j, \mathbf{a}_k) \text{Tr}((\mathbb{1} \otimes \mathbf{\Pi}_j \mathbf{\Pi}_k) \mathbf{B}) \right] \\ &= \sum_{j \in [m']} \mathbb{E}[\xi'(\mathbf{a}_j) \text{Tr}((\mathbb{1} \otimes \mathbf{\Pi}_j) \mathbf{B})] \\ &= \mathbb{E}[\text{Tr } \xi'(\mathbf{A}) \mathbf{B}] \\ &= \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathbb{E}[\text{Tr}(\mathbb{1}_m \otimes \xi'(\mathbf{C})) (\mathcal{B}_\sigma \otimes \mathbf{X}_\sigma)] \\ &= \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathbb{E}[\text{Tr } \mathcal{B}_\sigma \cdot \text{Tr } \xi'(\mathbf{C}) \mathbf{X}_\sigma] = 0, \end{aligned}$$

where the last equality follows from the orthogonality of $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$.

$$\begin{aligned} \mathbb{E}[\text{Tr } D\xi(\mathbf{C})[\mathbf{D}]] &= \mathbb{E}[\text{Tr } \xi'(\mathbf{C}) \mathbf{D}] \\ &= \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathbb{E}[\mathbf{x}_{i+1, \sigma} \cdot \text{Tr } \xi'(\mathbf{C}) \mathbf{X}_\sigma] \\ &= \sum_{\sigma \in [m^2]_{\geq 0}: \sigma \neq 0} \mathbb{E}[\mathbf{x}_{i+1, \sigma}] \cdot \mathbb{E}[\text{Tr } \xi'(\mathbf{C}) \mathbf{X}_\sigma] = 0, \end{aligned}$$

where the last equality follows from the orthogonality of \mathbf{x} .

By Fact 53, we have

$$\begin{aligned} & \mathbb{E}[\text{Tr } D^2\xi(\mathbf{A})[\mathbf{B}]] \\ &= \sum_{j, k, \ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \text{Tr}((\mathbb{1} \otimes \mathbf{\Pi}_j) \mathbf{B} (\mathbb{1} \otimes \mathbf{\Pi}_k) \mathbf{B} (\mathbb{1} \otimes \mathbf{\Pi}_\ell)) \right] \\ &= \sum_{\sigma, \tau \neq 0} \sum_{j, k, \ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \text{Tr}(\mathcal{B}_\sigma \mathcal{B}_\tau) \cdot \text{Tr}(\mathbf{\Pi}_j \mathbf{X}_\sigma \mathbf{\Pi}_k \mathbf{X}_\tau \mathbf{\Pi}_\ell) \right] \\ &= \sum_{\sigma \neq 0} \sum_{j, k, \ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \text{Tr}(\mathbf{\Pi}_j \mathbf{X}_\sigma \mathbf{\Pi}_k \mathbf{X}_\sigma \mathbf{\Pi}_\ell) \right], \end{aligned}$$

30:44 The Computational Advantage of MIP* Vanishes in the Presence of Noise

where the last equality follows from the orthogonality of $\{\mathcal{B}_i\}_{i=0}^{m^2-1}$.

$$\begin{aligned}
& \mathbb{E}[\text{Tr } D^2 \xi(\mathbf{C}) [\mathbf{D}]] \\
&= \sum_{j,k,\ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \text{Tr}(\mathbf{\Pi}_j \mathbf{D} \mathbf{\Pi}_k \mathbf{D} \mathbf{\Pi}_\ell) \right] \\
&= \sum_{\sigma, \tau \neq 0} \sum_{j,k,\ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \mathbf{x}_{i+1,\sigma} \mathbf{x}_{i+1,\tau} \cdot \text{Tr}(\mathbf{\Pi}_j \mathbf{X}_\sigma \mathbf{\Pi}_k \mathbf{X}_\tau \mathbf{\Pi}_\ell) \right] \\
&= \sum_{\sigma, \tau \neq 0} \sum_{j,k,\ell \in [m']} \mathbb{E}[\mathbf{x}_{i+1,\sigma} \mathbf{x}_{i+1,\tau}] \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \cdot \text{Tr}(\mathbf{\Pi}_j \mathbf{X}_\sigma \mathbf{\Pi}_k \mathbf{X}_\tau \mathbf{\Pi}_\ell) \right] \\
&= \sum_{\sigma \neq 0} \sum_{j,k,\ell \in [m']} \mathbb{E} \left[\xi^{[2]}(\mathbf{a}_j, \mathbf{a}_k, \mathbf{a}_\ell) \text{Tr}(\mathbf{\Pi}_j \mathbf{X}_\sigma \mathbf{\Pi}_k \mathbf{X}_\sigma \mathbf{\Pi}_\ell) \right],
\end{aligned}$$

where the last equality follows from the orthogonality of \mathbf{x} . \triangleleft

Proof of Lemma 16. Let $\lambda > 0$ be determined later, and ζ_λ be defined as in Fact 57. By Theorem 15 and Fact 57,

$$\left| m^{-n} \text{Tr } \zeta_\lambda(P) - m^{-h} \mathbb{E}[\text{Tr } \zeta_\lambda(P^H(\mathbf{x}))] \right| \leq C B_3 \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d / \lambda,$$

where C, B_3 are universal constants. By Fact 57 we also have

$$\left| m^{-n} \text{Tr } \zeta(P) - m^{-n} \text{Tr } \zeta_\lambda(P) \right| \leq 2\lambda^2$$

and

$$\left| m^{-h} \mathbb{E}[\text{Tr } \zeta(P^H(\mathbf{x}))] - m^{-h} \mathbb{E}[\text{Tr } \zeta_\lambda(P^H(\mathbf{x}))] \right| \leq 2\lambda^2.$$

By the triangle inequality, we have

$$\left| m^{-n} \text{Tr } \zeta(P) - m^{-h} \mathbb{E}[\text{Tr } \zeta(P^H(\mathbf{x}))] \right| \leq 4\lambda^2 + C B_3 \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d / \lambda.$$

Choosing $\lambda = \left(C B_3 \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d / 8 \right)^{1/3}$, we have

$$\left| m^{-n} \text{Tr } \zeta(P) - m^{-h} \mathbb{E}[\text{Tr } \zeta(P^H(\mathbf{x}))] \right| \leq 3 \left(C B_3 \max\{9m, 1/\eta^4\}^d \sqrt{\tau} d \right)^{2/3}. \quad \blacktriangleleft$$

Proof of Theorem 19. Let $\lambda > 0$ be determined later and let ζ_λ be defined as in Fact 57. By Theorem 20 and Fact 57,

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}}[\text{Tr } \zeta_\lambda(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_f}[\text{Tr } \zeta_\lambda(\mathbf{P}(\mathbf{x}_f))] \right| \leq 4 C_1 B_4 \lambda^{-2} (9m)^d d \tau,$$

where C_1, B_4 are universal constants. By Fact 57 we also have

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}}[\text{Tr } \zeta(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{b}}[\text{Tr } \zeta_\lambda(P(\mathbf{b}))] \right| \leq 2\lambda^2$$

and

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_f}[\text{Tr } \zeta_\lambda(\mathbf{P}(\mathbf{x}_f))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_f}[\text{Tr } \zeta(\mathbf{P}(\mathbf{x}_f))] \right| \leq 2\lambda^2.$$

By the triangle inequality, we have

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}} [\text{Tr } \zeta(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_{\mathbf{f}}} [\text{Tr } \zeta(\mathbf{P}(\mathbf{x}_{\mathbf{f}}))] \right| \leq 4\lambda^2 + 4C_1 B_4 \lambda^{-2} (9m)^d d\tau.$$

Choosing $\lambda = (C_1 B_4 (9m)^d d\tau)^{1/4}$, we have

$$\left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}} [\text{Tr } \zeta(P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_{\mathbf{f}}} [\text{Tr } \zeta(\mathbf{P}(\mathbf{x}_{\mathbf{f}}))] \right| \leq 8 (C_1 B_4 (9m)^d d\tau)^{1/2}.$$

Let $C = 8\sqrt{C_1 B_4}$, we conclude the result. \blacktriangleleft

► **Lemma 81.** *Given $d, n \in \mathbb{Z}_{>0}$, and a random matrix*

$$P(\mathbf{b}) = \sum_{S \subseteq [n]: |S| \leq d} \mathbf{b}_S P_S,$$

where \mathbf{b} is a $2d$ -wise uniform random vector from $\{\pm 1\}^n$ and $\mathbb{E}_{\mathbf{b}} [\|P(\mathbf{b})\|_2^2] \leq 1$, it holds that

$$\sum_{i=1}^n \text{VarInf}_i(P(\mathbf{b})) \leq d.$$

Proof.

$$\begin{aligned} \sum_{i=1}^n \text{VarInf}_i(P(\mathbf{b})) &= \sum_{i=1}^n \sum_{S \ni i} \|P_S\|_2^2 \\ &= \sum_{S \subseteq [n]: |S| \leq d} |S| \|P_S\|_2^2 \\ &\leq d \sum_{S \subseteq [n]: |S| \leq d} \|P_S\|_2^2 \\ &= d \mathbb{E}_{\mathbf{b}} [\|P(\mathbf{b})\|_2^2] \leq d. \end{aligned} \quad \blacktriangleleft$$

The following lemma is crucial to our proof. The proof follows closely to the proof of [36, Lemma 5.4].

► **Lemma 82.** *Given $d, n, p \in \mathbb{Z}_{>0}$, and a random matrix*

$$P(\mathbf{b}) = \sum_{S \subseteq [n]: |S| \leq d} \mathbf{b}_S P_S,$$

satisfying $\mathbb{E}_{\mathbf{b}} [\|P(\mathbf{b})\|_2^2] \leq 1$, where \mathbf{b} is a $2d$ -wise uniform random vector drawn from $\{\pm 1\}^n$, let $\mathcal{F} = \{f: [n] \rightarrow [p]\}$ be a family of pairwise uniform hash functions. Then for $\mathbf{f} \sim_{\mathbf{u}} \mathcal{F}$,

$$\mathbb{E}_{\mathbf{f}} \left[\sum_{j=1}^p \text{VarInf}_{\mathbf{f}, j}(P(\mathbf{b}))^2 \right] \leq \sum_{i=1}^n \text{VarInf}_i(P(\mathbf{b}))^2 + \frac{d^2}{p}.$$

Proof. Fix $j \in [p]$ and for $1 \leq i \leq n$, let \mathbf{X}_i be the indicator variable that is 1 if $f(i) = j$ and 0 otherwise. For brevity, let $\tau_i = \text{VarInf}_i(P(\mathbf{b}))$ for $i \in [n]$. Now,

$$\begin{aligned} \text{VarInf}_{\mathbf{f}, j}(P(\mathbf{b})) &= \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \|P_S\|_2^2 \leq \sum_S \|P_S\|_2^2 \left(\sum_{i \in S} \mathbf{X}_i \right) \\ &= \sum_{i \in [n]} \mathbf{X}_i \sum_{S \ni i} \|P_S\|_2^2 = \sum_{i \in [n]} \mathbf{X}_i \tau_i \end{aligned}$$

30:46 The Computational Advantage of MIP* Vanishes in the Presence of Noise

Thus

$$\text{VarInf}_{\mathbf{f},j}(P(\mathbf{b}))^2 \leq \left(\sum_{i \in [n]} \mathbf{X}_i \tau_i \right)^2 = \sum_{i \in [n]} \mathbf{X}_i^2 \tau_i^2 + \sum_{i \neq k} \mathbf{X}_i \mathbf{X}_k \tau_i \tau_k.$$

Note that $\mathbb{E}[\mathbf{X}_i] = 1/p$ and for $i \neq k$, $\mathbb{E}[\mathbf{X}_i \mathbf{X}_k] = 1/p^2$. Thus

$$\mathbb{E} \left[\text{VarInf}_{\mathbf{f},j}(P(\mathbf{b}))^2 \right] \leq \frac{1}{p} \sum_i \tau_i^2 + \sum_{i \neq k} \tau_i \tau_k \frac{1}{p^2} \leq \frac{1}{p} \sum_i \tau_i^2 + \frac{1}{p^2} \left(\sum_i \tau_i \right)^2.$$

The lemma follows by using Lemma 81 and summing all $j \in [p]$. ◀

We are ready to prove Theorem 20.

Proof of Theorem 20. We prove this by a hybrid argument.

Denote $\mathbf{b}^{(0)} = \mathbf{b} = G(f, \mathbf{b}, \dots, \mathbf{b})$. For $j \in [p]$, define $\mathbf{b}^{(j)} = G(f, \mathbf{z}^1, \dots, \mathbf{z}^j, \mathbf{b}, \dots, \mathbf{b})$, i.e., substituting $\mathbf{b}^{(j-1)}|_{f^{-1}(j)}$ with $\mathbf{z}^j|_{f^{-1}(j)}$. Then $\mathbf{b}^{(p)} = \mathbf{x}_{\mathbf{f}}$, and

$$\begin{aligned} \mathbf{P}(\mathbf{b}^{(j-1)}) &= \sum_{S: S \cap f^{-1}(j) = \emptyset} \mathbf{b}_S^{(j-1)} P_S + \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \mathbf{b}_S^{(j-1)} P_S \\ \mathbf{P}(\mathbf{b}^{(j)}) &= \sum_{S: S \cap f^{-1}(j) = \emptyset} \mathbf{b}_S^{(j)} P_S + \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \mathbf{b}_S^{(j)} P_S. \end{aligned}$$

Note that for $S \cap f^{-1}(j) = \emptyset$, $\mathbf{b}_S^{(j-1)} = \mathbf{b}_S^{(j)}$. Denote

$$\mathbf{A} = \sum_{S: S \cap f^{-1}(j) = \emptyset} \mathbf{b}_S^{(j)} P_S, \quad \mathbf{B} = \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \mathbf{b}_S^{(j-1)} P_S, \quad \mathbf{C} = \sum_{S: S \cap f^{-1}(j) \neq \emptyset} \mathbf{b}_S^{(j)} P_S.$$

We have

$$\begin{aligned} & \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\text{Tr} \xi \left(\mathbf{P}(\mathbf{b}^{(j-1)}) \right) \right] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\text{Tr} \xi \left(\mathbf{P}(\mathbf{b}^{(j)}) \right) \right] \right| \\ &= \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\text{Tr} \xi (\mathbf{A} + \mathbf{B}) \right] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\text{Tr} \xi (\mathbf{A} + \mathbf{C}) \right] \right| \\ &= \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\sum_{k=0}^3 \frac{1}{k!} \text{Tr} D^k \xi(\mathbf{A})[\mathbf{B}] + \text{Tr} \Delta_{4,\xi}(\mathbf{A}, \mathbf{B}) \right] \right. \\ & \quad \left. - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\sum_{k=0}^3 \frac{1}{k!} \text{Tr} D^k \xi(\mathbf{A})[\mathbf{C}] + \text{Tr} \Delta_{4,\xi}(\mathbf{A}, \mathbf{C}) \right] \right| \end{aligned}$$

By Fact 53 and the fact that \mathbf{z}_j is $4d$ -wise uniform, we have for $k = 0, 1, 2, 3$,

$$\mathbb{E}_{\mathbf{b}^{(j-1)}} \left[\text{Tr} D^k \xi(\mathbf{A})[\mathbf{B}] \right] = \mathbb{E}_{\mathbf{b}^{(j)}} \left[\text{Tr} D^k \xi(\mathbf{A})[\mathbf{C}] \right].$$

Thus,

$$\begin{aligned} & \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\text{Tr} \xi \left(\mathbf{P}(\mathbf{b}^{(j-1)}) \right) \right] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\text{Tr} \xi \left(\mathbf{P}(\mathbf{b}^{(j)}) \right) \right] \right| \\ & \leq \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\left| \text{Tr} \Delta_{4,\xi}(\mathbf{A}, \mathbf{B}) \right| \right] + \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\left| \text{Tr} \Delta_{4,\xi}(\mathbf{A}, \mathbf{C}) \right| \right] \\ & \leq C_1 C_0 \left(\mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} \left[\left\| \mathbf{B} \right\|_4^4 \right] + \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} \left[\left\| \mathbf{C} \right\|_4^4 \right] \right), \end{aligned}$$

where the last inequality is from Fact 54, and C_1 is a universal constant. Because \mathbf{z}_j is $4d$ -wise uniform, we have $\mathbb{E}_{\mathbf{b}^{(j-1)}} [\|\mathbf{B}\|_4^4] = \mathbb{E}_{\mathbf{b}^{(j)}} [\|\mathbf{C}\|_4^4]$. Using Theorem 14 with $\eta \leftarrow 1/\sqrt{3}$,

$$\mathbb{E}_{\mathbf{b}^{(j-1)}} [\|\mathbf{B}\|_4^4] \leq (9m)^d \left(\mathbb{E}_{\mathbf{b}^{(j)}} [\|\mathbf{B}\|_2^2] \right)^2.$$

So we have

$$\begin{aligned} & \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j-1)}} [\text{Tr } \xi (\mathbf{P}(\mathbf{b}^{(j-1)}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{b}^{(j)}} [\text{Tr } \xi (\mathbf{P}(\mathbf{b}^{(j)}))] \right| \\ & \leq 2C_1 C_0 (9m)^d \mathbb{E}_{\mathbf{f}} \left[\left(\mathbb{E}_{\mathbf{b}^{(j-1)}} [\|\mathbf{B}\|_2^2] \right)^2 \right] \\ & = 2C_1 C_0 (9m)^d \mathbb{E}_{\mathbf{f}} [\text{VarInf}_{\mathbf{f}, j} (P(\mathbf{b}))^2]. \end{aligned}$$

Summing over $j \in [p]$ and by Lemma 82, we have

$$\begin{aligned} & \left| \frac{1}{m^h} \mathbb{E}_{\mathbf{b}} [\text{Tr } \zeta (P(\mathbf{b}))] - \frac{1}{m^h} \mathbb{E}_{\mathbf{f}, \mathbf{x}_{\mathbf{f}}} [\text{Tr } \zeta (\mathbf{P}(\mathbf{x}_{\mathbf{f}}))] \right| \\ & \leq 2C_1 C_0 (9m)^d \left(\sum_{i=1}^n \text{VarInf}_i (P(\mathbf{b}))^2 + \frac{d^2}{p} \right) \\ & \leq 2C_1 C_0 (9m)^d \left(\tau \sum_{i=1}^n \text{VarInf}_i (P(\mathbf{b})) + \frac{d^2}{p} \right) \\ & \leq 4C_1 C_0 (9m)^d d\tau, \end{aligned}$$

where the last inequality is by Lemma 81 and $p \geq d/\tau$. ◀

B.2 Positivity Tester for Low Degree Operators

Proof of Theorem 23. Consider the algorithm below

Input: Parameters given in Definition 22.

Algorithm:

1. **Regularization:** Compute $\tau = \delta^3 / (8 \cdot 3^{2d} m^d d^2)$. For each i , compute the influence

$$\text{Inf}_i (P) = \sum_{\sigma: \sigma_i \neq 0} \widehat{P}(\sigma)^2.$$

Let $H = \{i : \text{Inf}_i (P) > \tau\}$.

2. **Derandomized invariance principle:** Let p be the smallest power of 2 satisfying $p \geq d/\tau$. Let $n = (m^2 - 1)(D - |H|)$ and $\mathcal{F} = \{f : [n] \rightarrow [p]\}$ be a family of pairwise uniform hash functions. For any $i \in [p]$, let \mathbf{z}^i be $4d$ -wise uniform random variables of length n and (\mathbf{z}^i) 's be independent across $i \in [p]$. For any $f \in \mathcal{F}$, set $\mathbf{x}_{\mathbf{f}} = G(f, \mathbf{z}^1, \dots, \mathbf{z}^p)$ as defined in Theorem 19. Define the random operator

$$P'(f, \mathbf{z}) = \sum_{\sigma \in [m^2]_{\geq 0}^D: |\sigma| \leq d} \widehat{P}(\sigma) \mathbf{x}_{\mathbf{f}, \sigma_H} \mathcal{B}_{\sigma_H}, \quad (19)$$

where $\mathbf{x}_{\mathbf{f}, \sigma_H} = \prod_{i \notin H} (\mathbf{x}_{\mathbf{f}})_{(m^2-1)(i-1)+\sigma_i}$ and $\mathcal{B}_{\sigma_H} = \bigotimes_{i \in H} \mathcal{B}_{\sigma_i}$.

3. Compute the distance to PSD: For each f, \mathbf{z} , compute

$$\delta_{f,\mathbf{z}} = m^{-|H|} \operatorname{Tr} \zeta(P'(f, \mathbf{z})).$$

4. Accept if

$$\mathbb{E}_{f,\mathbf{z}} [\delta_{f,\mathbf{z}}] < \beta.$$

Time complexity

- Given that each computation of $\operatorname{Inf}_i(P)$ entails calculating a sum of products of Fourier coefficients, the time required can be expressed as $\sum_{i=0}^d \binom{D}{i} (m^2 - 1)^i \leq dm^{2d} D^d$. In addition, the time needed to determine the set H is at most D .
- When fixing f and \mathbf{z} , computing $\delta_{f,\mathbf{z}}$ takes time

$$\exp(|H|) = \exp(|d/\tau|) = \exp(\operatorname{poly}(m^d, 1/\delta)).$$

- By Lemma 60 and Corollary 61, the enumeration over \mathcal{F} and \mathbf{z} takes time polynomial in D , thus computing the expectation of $\delta_{f,\mathbf{z}}$ also takes time polynomial in D .

Correctness

By the choice of τ , it holds that

$$\left(3^d m^{d/2} \sqrt{\tau} d\right)^{2/3} \leq \delta/2, \quad (20)$$

$$C \sqrt{(9m)^d d \tau} \leq \delta/2. \quad (21)$$

Let $\mathbf{b} \in \{-1, 1\}^n$ be uniformly distributed. Consider the operator $P^{(1)}$ obtained by replacing the basis outside of H by random bits. That is,

$$P^{(1)}(\mathbf{b}) = \sum_{\sigma \in [m^2]_{\geq 0}^D: |\sigma| \leq d} \widehat{P}(\sigma) \mathbf{b}_{\sigma_{\bar{H}}} \mathcal{B}_{\sigma_H},$$

where $\mathbf{b}_{\sigma_{\bar{H}}} = \prod_{i \notin H} \mathbf{b}_{(m^2-1)(i-1)+\sigma_i}$ and $\mathcal{B}_{\sigma_H} = \otimes_{i \in H} \mathcal{B}_{\sigma_i}$.

By Eq. (20) and Lemma 16, we have

$$\left| \frac{1}{m^{|H|}} \mathbb{E}_{\mathbf{b}} \left[\operatorname{Tr} \zeta(P^{(1)}(\mathbf{b})) \right] - \frac{1}{m^D} \operatorname{Tr} \zeta(P) \right| \leq \delta/2.$$

Then we define $P^{(2)}$ to be the operator obtained by replacing \mathbf{b} with $\mathbf{x}_{f,\mathbf{z}}$, which is the operator in Equation (19). By Eq. (21) and Theorem 19,

$$\left| \frac{1}{m^{|H|}} \mathbb{E}_{\mathbf{b}} \left[\operatorname{Tr} \zeta(P^{(2)}(\mathbf{x}_{f,\mathbf{z}})) \right] - \frac{1}{m^{|H|}} \mathbb{E}_{f,\mathbf{z}} \left[\operatorname{Tr} \zeta(P^{(1)}(\mathbf{b})) \right] \right| \leq \delta/2.$$

Thus by triangle inequality, we have

$$\left| \frac{1}{m^{|H|}} \mathbb{E}_{f,\mathbf{z}} \left[\operatorname{Tr} \zeta(P^{(2)}(\mathbf{x}_{f,\mathbf{z}})) \right] - \frac{1}{m^D} \operatorname{Tr} \zeta(P) \right| \leq \delta. \quad (22)$$

The algorithm computes $m^{-|H|} \mathbb{E}_{f,\mathbf{z}} \left[\operatorname{Tr} \zeta(P^{(2)}(\mathbf{x}_{f,\mathbf{z}})) \right]$. By Eq.(22), the value is smaller than β if $m^{-D} \operatorname{Tr} \zeta(P) < \beta - \delta$; or greater than β if $m^{-D} \operatorname{Tr} \zeta(P) > \beta + \delta$. Therefore, the algorithm distinguishes the two cases correctly. ◀

B.3 Noisy Nonlocal Games are NP-complete

B.3.1 The nondeterministic algorithm

First, we prove an upper bound on the Number of Noisy MES's for Nonlocal Games. The proof follows closely to that of [46]. The major difference is that in the proof of [46], each pair of questions (x, y) is treated independently. Then, a union bound is applied to all possible questions. To improve the upper bound, we take into account the distribution of the questions, combined with a better Gaussian dimension reduction in [47]. Then our new upper bound below only depends polynomially on the size of the question set whereas the previous one has an exponential dependence.

Gaussian Dimension Reduction

The following lemma is a simplified version of [47, Lemma 5.13], with the questions and answers being classical. In the proof of Theorem 28, we will use this lemma, after we replace the low-influence registers by Gaussian random variables, to further reduce the dimension of the Gaussian space. The only difference is in Item 3 of Lemma 83, where we preserve the expectation of the ζ function value over the random variable \mathbf{M} . In the previous version (Item 2 of [47, Lemma 5.13]), we used Markov's inequality on the expectation value. As the notations are considerably different, we include a new proof for completeness.

► **Lemma 83** ([47, Lemma 5.13]). *Given parameters $\rho \in [0, 1]$, $\delta > 0$, $d, n, h \in \mathbb{Z}_{>0}$, $m \geq 2$, an m -dimensional noisy MES ψ_{AB} with the maximal correlation $\rho = \rho(\psi_{AB})$, and degree- d multilinear joint random matrices*

$$(P(\mathbf{g}), Q(\mathbf{h})) = \left(\sum_{S \subseteq [n]} \mathbf{g}_S P_S, \sum_{S \subseteq [n]} \mathbf{h}_S Q_S \right)_{(\mathbf{g}, \mathbf{h}) \sim \mathcal{G}_\rho^{\otimes n}},$$

where $\mathbf{g}_S = \prod_{i \in S} \mathbf{g}_i$, $\mathbf{h}_S = \prod_{i \in S} \mathbf{h}_i$ and $P_S, Q_S \in \mathcal{H}_m^{\otimes h}$ for all $S \subseteq [n]$, satisfying

$$\mathbb{E}_{\mathbf{g}} \left[\|\| P(\mathbf{g}) \|\|_2^2 \right] \leq 1 \text{ and } \mathbb{E}_{\mathbf{h}} \left[\|\| Q(\mathbf{h}) \|\|_2^2 \right] \leq 1.$$

Let $L^2(\mathcal{H}_m^{\otimes h}, \gamma_n)$ be the space of random operators whose Fourier coefficients are square-integrable with respect to the measure γ_n . Then there exists an explicitly computable $n_0 = n_0(d, \delta)$ and maps $f_M, g_M : L^2(\mathcal{H}_m^{\otimes h}, \gamma_n) \rightarrow L^2(\mathcal{H}_m^{\otimes h}, \gamma_n)$ for $M \in \mathbb{R}^{n \times n_0}$ and joint random operators $(P(M\tilde{\mathbf{x}}), Q(M\tilde{\mathbf{y}})) = (f_M(P(\mathbf{g})), g_M(Q(\mathbf{h})))$:

$$(P(M\tilde{\mathbf{x}}), Q(M\tilde{\mathbf{y}})) = \left(\sum_{S \subseteq [n]} \mathbf{u}_S P_S, \sum_{S \subseteq [n]} \mathbf{v}_S Q_S \right)_{(\mathbf{x}, \mathbf{y}) \sim \mathcal{G}_\rho^{\otimes n_0}},$$

where $\tilde{\mathbf{x}} = \mathbf{x}/\|\mathbf{x}\|_2$, $\tilde{\mathbf{y}} = \mathbf{y}/\|\mathbf{y}\|_2$, $\mathbf{u}_S = \prod_{i \in S} \langle m_i, \tilde{\mathbf{x}} \rangle$, $\mathbf{v}_S = \prod_{i \in S} \langle m_i, \tilde{\mathbf{y}} \rangle$, $\langle \cdot, \cdot \rangle$ denotes the standard inner product over \mathbb{R}^{n_0} and m_i denotes the i 'th row of M , such that if we sample $\mathbf{M} \sim \gamma_{n \times n_0}$, then the following hold:

1. With probability at least $1 - 2\delta$, we have

$$\mathbb{E}_{\tilde{\mathbf{x}}} \left[\|\| P(\mathbf{M}\tilde{\mathbf{x}}) \|\|_2^2 \right] \leq 1 + \delta \quad \text{and} \quad \mathbb{E}_{\tilde{\mathbf{y}}} \left[\|\| Q(\mathbf{M}\tilde{\mathbf{y}}) \|\|_2^2 \right] \leq 1 + \delta.$$

2. With probability at least $1 - \delta$, we have

$$\left| \mathbb{E}_{\tilde{\mathbf{x}}, \tilde{\mathbf{y}}} \left[\text{Tr} \left((P(\mathbf{M}\tilde{\mathbf{x}}) \otimes Q(\mathbf{M}\tilde{\mathbf{y}})) \psi_{AB}^{\otimes h} \right) \right] - \mathbb{E}_{\mathbf{g}, \mathbf{h}} \left[\text{Tr} \left((P(\mathbf{g}) \otimes Q(\mathbf{h})) \psi_{AB}^{\otimes h} \right) \right] \right| \leq \delta.$$

30:50 The Computational Advantage of MIP* Vanishes in the Presence of Noise

$$3. \mathbb{E}_{\mathbf{g}}[\text{Tr } \zeta(P(\mathbf{g}))] = \mathbb{E}_{\mathbf{M}, \mathbf{x}}[\text{Tr } \zeta(P(\mathbf{M}\tilde{\mathbf{x}}))] \quad \text{and} \quad \mathbb{E}_{\mathbf{h}}[\text{Tr } \zeta(Q(\mathbf{h}))] = \mathbb{E}_{\mathbf{M}, \mathbf{y}}[\text{Tr } \zeta(Q(\mathbf{M}\tilde{\mathbf{y}}))].$$

4. the maps f_M, g_M are linear and unital for any nonzero $M \in \mathbb{R}^{n \times n_0}$.

In particular, one may take $n_0 = \frac{d^{O(d)}}{\delta^6}$.

For $M \in \mathbb{R}^{n \times n_0}$, denote $F(M) = \mathbb{E}_{\mathbf{x}, \mathbf{y}}[\text{Tr}((P(\mathbf{M}\tilde{\mathbf{x}}) \otimes Q(\mathbf{M}\tilde{\mathbf{y}})) \psi_{AB}^{\otimes h})]$. To prove Lemma 83 item 2, we need the following lemma.

► **Lemma 84.** *In the setting of Lemma 83, given $d \in \mathbb{Z}_{>0}$, $\delta > 0$, there exists $n_0 = \frac{d^{O(d)}}{\delta^2}$ such that the following holds: For $\mathbf{M} \sim \gamma_{n \times n_0}$,*

$$\left| \mathbb{E}[F(\mathbf{M})] - \mathbb{E}_{\mathbf{g}, \mathbf{h}}[\text{Tr}((P(\mathbf{g}) \otimes Q(\mathbf{h})) \psi_{AB}^{\otimes h})] \right| \leq \delta,$$

$$\text{Var}[F(\mathbf{M})] \leq \delta.$$

We use the following lemma to prove Lemma 84.

► **Lemma 85** ([20, Lemma A.8, A.9]). *Given parameters d and δ , there exists an explicitly computable $n_0(d, \delta)$ such that the followings hold:*

■ *For any subsets $S, T \subseteq [n]$ satisfying $|S|, |T| \leq d$, it holds that*

$$\text{if } S \neq T : \quad \mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}}[\mathbf{u}_S \mathbf{v}_T] = 0,$$

$$\text{if } S = T : \quad \left| \mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}}[\mathbf{u}_S \mathbf{v}_T] - \rho^{|S|} \right| \leq \delta.$$

■ *Let $(\mathbf{x}', \mathbf{y}') \sim \mathcal{G}_\rho^{\otimes n_0}$ be independent of (\mathbf{x}, \mathbf{y}) , and let $\mathbf{u}'_S = \prod_{i \in S} \langle m_i, \frac{\mathbf{x}'_i}{\|\mathbf{x}'_i\|_2} \rangle$, $\mathbf{v}'_S = \prod_{i \in S} \langle m_i, \frac{\mathbf{y}'_i}{\|\mathbf{y}'_i\|_2} \rangle$. For any subsets $S, T, S', T' \subseteq [n]$ satisfying $|S|, |T|, |S'|, |T'| \leq d$, it holds that*

$$\text{if } S \triangle T \triangle S' \triangle T' \neq \emptyset :$$

$$\left| \mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}, \mathbf{x}', \mathbf{y}'}[\mathbf{u}_S \mathbf{v}_T \mathbf{u}'_{S'} \mathbf{v}'_{T'}] - \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}}[\mathbf{u}_S \mathbf{v}_T] \right) \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}', \mathbf{y}'}[\mathbf{u}'_{S'} \mathbf{v}'_{T'}] \right) \right| = 0,$$

$$\text{if } S \triangle T \triangle S' \triangle T' = \emptyset :$$

$$\left| \mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}, \mathbf{x}', \mathbf{y}'}[\mathbf{u}_S \mathbf{v}_T \mathbf{u}'_{S'} \mathbf{v}'_{T'}] - \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}}[\mathbf{u}_S \mathbf{v}_T] \right) \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}', \mathbf{y}'}[\mathbf{u}'_{S'} \mathbf{v}'_{T'}] \right) \right| \leq \delta.$$

Here, $S \triangle T \triangle S' \triangle T'$ is the symmetric difference of the sets S, T, S', T' , equivalently, the set of all $i \in [n]$ which appear an odd number of times in the multiset $S \sqcup T \sqcup S' \sqcup T'$.

In particular, one may take $n_0 = \frac{d^{O(d)}}{\delta^2}$.

Proof of Lemma 84. Use Lemma 85 with parameters d and δ , we have

$$\begin{aligned}
& \left| \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] - \mathbb{E}_{\mathbf{g}, \mathbf{h}} [\text{Tr}((P(\mathbf{g}) \otimes Q(\mathbf{h})) \psi_{AB}^{\otimes h})] \right| \\
&= \left| \sum_{S, T \subseteq [n]} \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}} [\mathbf{u}_S \mathbf{v}_T] - \mathbb{E}_{\mathbf{g}, \mathbf{h}} [\mathbf{g}_S \mathbf{h}_T] \right) \text{Tr}((P_S \otimes Q_T) \psi_{AB}^{\otimes h}) \right| \\
&= \left| \sum_{S \subseteq [n]} \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}} [\mathbf{u}_S \mathbf{v}_S] - \rho^{|S|} \right) \text{Tr}((P_S \otimes Q_S) \psi_{AB}^{\otimes h}) \right| \\
&\leq \delta \sum_{S \subseteq [n]} |\text{Tr}((P_S \otimes Q_S) \psi_{AB}^{\otimes h})| \quad (\text{Lemma 85}) \\
&\leq \delta \sum_{S \subseteq [n]} \|P_S\|_2 \|Q_S\|_2 \quad (\text{Fact 66}) \\
&\leq \delta \sqrt{\sum_{S \subseteq [n]} \|P_S\|_2^2 \cdot \sum_{S \subseteq [n]} \|Q_S\|_2^2} \\
&= \delta \left(\mathbb{E}_{\mathbf{g}} [\|P(\mathbf{g})\|_2^2] \mathbb{E}_{\mathbf{h}} [\|Q(\mathbf{h})\|_2^2] \right)^{1/2} \leq \delta.
\end{aligned}$$

Use Lemma 85 with parameters d and $\delta \leftarrow \delta/9^d$, we have

$$\begin{aligned}
& \text{Var} [F(\mathbf{M})] \\
&= \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})^2] - \left(\mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] \right)^2 \\
&\leq \sum_{S, T, S', T' \subseteq [n]} \left| \mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}, \mathbf{x}', \mathbf{y}'} [\mathbf{u}_S \mathbf{v}_T \mathbf{u}'_{S'} \mathbf{v}'_{T'}] - \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}, \mathbf{y}} [\mathbf{u}_S \mathbf{v}_T] \right) \left(\mathbb{E}_{\mathbf{M}, \mathbf{x}', \mathbf{y}'} [\mathbf{u}'_{S'} \mathbf{v}'_{T'}] \right) \right| \\
&\quad \left| \text{Tr}((P_S \otimes Q_S) \psi_{AB}^{\otimes h}) \text{Tr}((P_{S'} \otimes Q_{S'}) \psi_{AB}^{\otimes h}) \right| \\
&\leq \frac{\delta}{9^d} \sum_{\substack{S, T, S', T' \subseteq [n] \\ S \Delta T \Delta S' \Delta T' = \emptyset}} \|P_S\|_2 \|Q_T\|_2 \|P_{S'}\|_2 \|Q_{T'}\|_2
\end{aligned}$$

To finish the proof, we will show that,

$$\sum_{\substack{S, T, S', T' \subseteq [n] \\ S \Delta T \Delta S' \Delta T' = \emptyset}} \|P_S\|_2 \|Q_T\|_2 \|P_{S'}\|_2 \|Q_{T'}\|_2 \leq 9^d \mathbb{E}_{\mathbf{g}} [\|P(\mathbf{g})\|_2^2] \mathbb{E}_{\mathbf{h}} [\|Q(\mathbf{h})\|_2^2]$$

Define functions $f, g : \{1, -1\}^n \rightarrow \mathbb{R}$ over the boolean hypercube as,

$$f(x) = \sum_{\substack{S \subseteq [n] \\ |S| \leq d}} \|P_S\|_2 \chi_S(x) \quad \text{and} \quad g(x) = \sum_{\substack{T \subseteq [n] \\ |T| \leq d}} \|Q_T\|_2 \chi_T(x)$$

By the hypercontractivity inequality over the boolean hypercube [42, Page 240]

$$\mathbb{E}_x [f(x)^4] \leq 9^d \left(\mathbb{E}_x [f(x)^2] \right)^2 \quad \text{and} \quad \mathbb{E}_x [g(x)^4] \leq 9^d \left(\mathbb{E}_x [g(x)^2] \right)^2,$$

30:52 The Computational Advantage of MIP* Vanishes in the Presence of Noise

we have

$$\begin{aligned}
& \sum_{\substack{S,T,S',T' \subseteq [n] \\ S \Delta T \Delta S' \Delta T' = \emptyset}} \|P_S\|_2 \|Q_T\|_2 \|P_{S'}\|_2 \|Q_{T'}\|_2 \\
&= \mathbb{E}_x [f(x)^2 g(x)^2] \\
&\leq \sqrt{\mathbb{E}_x [f(x)^4] \mathbb{E}_x [g(x)^4]} \\
&\leq 9^d \mathbb{E}_x [f(x)^2] \mathbb{E}_x [g(x)^2] \\
&= 9^d \sum_{S \subseteq [n]} \|P_S\|_2^2 \sum_{S \subseteq [n]} \|Q_S\|_2^2 \\
&= 9^d \mathbb{E}_{\mathbf{g}} [\|P(\mathbf{g})\|_2^2] \mathbb{E}_{\mathbf{h}} [\|Q(\mathbf{h})\|_2^2] \leq 9^d.
\end{aligned}$$

Thus $\text{Var}[F(\mathbf{M})] \leq \delta$. ◀

To prove Lemma 83 Item 1, we need the following lemma whose proof is similar to that of Lemma 84. We omit the proof here.

► **Lemma 86.** *In the setting of Lemma 83, given $d \in \mathbb{Z}_{>0}$, $\delta > 0$, there exists $n_0 = \frac{d^{O(d)}}{\delta^2}$ such that the following holds: For $\mathbf{M} \sim \gamma_{n \times n_0}$,*

$$\begin{aligned}
& \left| \mathbb{E}_{\mathbf{M}, \mathbf{x}} [\|P(\mathbf{M}\tilde{\mathbf{x}})\|_2^2] - \mathbb{E}_{\mathbf{g}} [\|P(\mathbf{g})\|_2^2] \right| \leq \delta, \\
& \text{Var}_{\mathbf{x}} \left[\mathbb{E}_{\mathbf{M}} [\|P(\mathbf{M}\tilde{\mathbf{x}})\|_2^2] \right] \leq \delta, \\
& \left| \mathbb{E}_{\mathbf{M}, \mathbf{y}} [\|Q(\mathbf{M}\tilde{\mathbf{y}})\|_2^2] - \mathbb{E}_{\mathbf{h}} [\|Q(\mathbf{h})\|_2^2] \right| \leq \delta, \\
& \text{Var}_{\mathbf{y}} \left[\mathbb{E}_{\mathbf{M}} [\|Q(\mathbf{M}\tilde{\mathbf{y}})\|_2^2] \right] \leq \delta.
\end{aligned}$$

Proof of Lemma 83. For item 2, we invoke Lemma 84 with parameters d and $\delta \leftarrow \delta^3/2$. Using Chebyshev's inequality, we have that for any $\eta > 0$,

$$\Pr_{\mathbf{M}} \left[\left| F(\mathbf{M}) - \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] \right| > \eta \right] \leq \frac{\delta^3}{2\eta^2}.$$

Using the triangle inequality, we get

$$\begin{aligned}
& \Pr_{\mathbf{M}} \left[\left| F(\mathbf{M}) - \mathbb{E}_{\mathbf{g}, \mathbf{h}} [\text{Tr}((P(\mathbf{g}) \otimes Q(\mathbf{h})) \psi_{AB}^{\otimes h})] \right| > \delta \right] \\
&\leq \Pr_{\mathbf{M}} \left[\left| F(\mathbf{M}) - \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] \right| + \left| \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] - \mathbb{E}_{\mathbf{g}, \mathbf{h}} [\text{Tr}((P(\mathbf{g}) \otimes Q(\mathbf{h})) \psi_{AB}^{\otimes h})] \right| > \delta \right] \\
&\leq \Pr_{\mathbf{M}} \left[\left| F(\mathbf{M}) - \mathbb{E}_{\mathbf{M}} [F(\mathbf{M})] \right| > \delta - \delta^3/2 \right] \leq \delta.
\end{aligned}$$

By Lemma 86, we can similarly argue for item 1. For item 3, note that for any fixed $x \in \mathbb{R}^{n_0}$, the distribution of $\mathbf{M}x/\|x\|_2$ is identical to γ_n . It is easy to verify Item 4. ◀

We are now ready to prove Theorem 28.

Proof of Theorem 28. The proof follows that in [46] step by step, except that the Gaussian dimension reduction step in the original proof is replaced by Lemma 83. Here, we include the proof for completeness.

Suppose the players use the strategy $\left(\left\{P_a^{x,(0)}\right\}_{a \in \mathcal{A}}^{x \in \mathcal{X}}, \left\{Q_b^{y,(0)}\right\}_{b \in \mathcal{B}}^{y \in \mathcal{Y}}\right)$ to achieve the highest winning probability when sharing n copies of ψ_{AB} , where $P_a^{x,(0)}$ is the POVM element of Alice corresponding to the answer a upon receiving the question x , and $Q_b^{y,(0)}$ is the POVM element of Bob corresponding to the answer b upon receiving the question y . Then for all $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, $P_a^{x,(0)} \geq 0$, $Q_b^{y,(0)} \geq 0$, $\sum_a P_a^{x,(0)} = \mathbb{1}$, $\sum_b Q_b^{y,(0)} = \mathbb{1}$, and $\omega_n(\mathfrak{G}, \psi_{AB}) = \text{val}_n\left(\left\{P_a^{x,(0)}\right\}, \left\{Q_b^{y,(0)}\right\}\right)$.

Let δ, τ be parameters which are chosen later. The proof is composed of several steps.

- **Smoothing.** This step allows us to restrict ourselves to strategies with low-degree POVMs.

More specifically, for any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, we apply the map $f^{(1)}$ implied by Lemma 62 to $P_a^{x,(0)}$ and $Q_b^{y,(0)}$ to get $P_a^{x,(1)}$ and $Q_b^{y,(1)}$, respectively. Note that for all x, y, a, b , $\left\|P_a^{x,(0)}\right\|_2^2 \leq 1$ and $\left\|Q_b^{y,(0)}\right\|_2^2 \leq 1$. Let $d = \frac{C \log^2 \frac{1}{\delta}}{\delta(1-\rho)}$, by Lemma 62 Item 3 and Item 4,

$$\left| \text{Tr} \left(\left(P_a^{x,(1)} \otimes Q_b^{y,(1)} \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^{x,(0)} \otimes Q_b^{y,(0)} \right) \psi_{AB}^{\otimes n} \right) \right| \leq \delta$$

and

$$\frac{1}{m^n} \text{Tr} \zeta(P_a^{x,(1)}) \leq \delta, \quad \frac{1}{m^n} \text{Tr} \zeta(Q_b^{y,(1)}) \leq \delta.$$

By Lemma 67 and Lemma 62 items 1, 2 and 5, the following hold.

1. For any x, y, a, b , $P_a^{x,(1)}$ and $Q_b^{y,(1)}$ are of degree at most d .
2. For any x, y, a, b , $\left\|P_a^{x,(1)}\right\|_2 \leq 1$ and $\left\|Q_b^{y,(1)}\right\|_2 \leq 1$.
3. $\left| \text{val}_n \left(\left\{P_a^{x,(1)}\right\}, \left\{Q_b^{y,(1)}\right\} \right) - \text{val}_n \left(\left\{P_a^{x,(0)}\right\}, \left\{Q_b^{y,(0)}\right\} \right) \right| \leq \delta t^2$,
4. $\frac{1}{m^n} \sum_{x,a} \mu_A(x) \text{Tr} \zeta \left(P_a^{x,(1)} \right) \leq \delta t$ and $\frac{1}{m^n} \sum_{y,b} \mu_B(y) \text{Tr} \zeta \left(Q_b^{y,(1)} \right) \leq \delta t$.
5. For any x, y , $\sum_{a \in \mathcal{A}} P_a^{x,(1)} = \sum_{b \in \mathcal{B}} Q_b^{y,(1)} = \mathbb{1}$.

- **Regularization.** In this step, we identify the set H of high-influence registers for all POVM elements.

For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, we apply Lemma 64 to $P_a^{x,(1)}$ and $Q_b^{y,(1)}$ to get sets $H_{x,a}$ and $H_{y,b}$ of size at most d/τ , respectively, such that

$$(\forall i \notin H_{x,a}) \text{Inf}_i \left(P_a^{x,(1)} \right) \leq \tau \quad \text{and} \quad (\forall i \notin H_{y,b}) \text{Inf}_i \left(Q_b^{y,(1)} \right) \leq \tau.$$

Set $H = \left(\bigcup_{x,a} H_{x,a} \right) \cup \left(\bigcup_{y,b} H_{y,b} \right)$, then $h = |H| \leq \frac{2std}{\tau}$, and

$$(\forall i \notin H) \text{Inf}_i \left(P_a^{x,(1)} \right) \leq \tau \quad \text{and} \quad \text{Inf}_i \left(Q_b^{y,(1)} \right) \leq \tau.$$

- **Invariance from $\mathcal{H}_m^{\otimes n}$ to $L^2(\mathcal{H}_m^{\otimes h}, \gamma_{(m^2-1)(n-h)})$.** In this step, we only keep the quantum registers in H and replace the rest of the quantum registers by Gaussian random variables. Hence, the number of quantum registers is reduced from n to $h = |H| = d/\tau$. For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, applying [46, Lemma 10.5] to $P_a^{x,(1)}$, $Q_b^{y,(1)}$ and H , we obtain joint random matrices

$$\left(P_a^{x,(2)}(\mathbf{g}), Q_b^{y,(2)}(\mathbf{h}) \right) \in L^2(\mathcal{H}_m^{\otimes h}, \gamma_{(m^2-1)(n-h)}) \times L^2(\mathcal{H}_m^{\otimes h}, \gamma_{(m^2-1)(n-h)}),$$

where $(\mathbf{g}, \mathbf{h}) \sim \mathcal{G}_\rho^{\otimes 2(m^2-1)(n-h)}$, such that the following hold.

1. For any x, y, a, b , $\mathbb{E}_{\mathbf{g}} \left[\left\| P_a^{x,(2)}(\mathbf{g}) \right\|_2^2 \right] \leq 1$ and $\mathbb{E}_{\mathbf{h}} \left[\left\| Q_b^{y,(2)}(\mathbf{h}) \right\|_2^2 \right] \leq 1$.
 2. $\mathbb{E}_{\mathbf{g}, \mathbf{h}} \left[\text{val}_h \left(\left\{ P_a^{x,(2)}(\mathbf{g}) \right\}, \left\{ Q_b^{y,(2)}(\mathbf{h}) \right\} \right) \right] = \text{val}_n \left(\left\{ P_a^{x,(1)} \right\}, \left\{ Q_b^{y,(1)} \right\} \right)$.
 3. $\sum_{x,a} \mu_A(x) \left| \frac{1}{m^h} \mathbb{E} \left[\text{Tr } \zeta \left(P_a^{x,(2)}(\mathbf{g}) \right) \right] - \frac{1}{m^n} \text{Tr } \zeta \left(P_a^{x,(1)} \right) \right| \leq O \left(t \left(3^d m^{d/2} \sqrt{\tau} d \right)^{2/3} \right)$ and $\sum_{y,b} \mu_B(y) \left| \frac{1}{m^h} \mathbb{E} \left[\text{Tr } \zeta \left(Q_b^{y,(2)}(\mathbf{h}) \right) \right] - \frac{1}{m^n} \text{Tr } \zeta \left(Q_b^{y,(1)} \right) \right| \leq O \left(t \left(3^d m^{d/2} \sqrt{\tau} d \right)^{2/3} \right)$.
 4. For any x, y , $\sum_{a \in \mathcal{A}} P_a^{x,(2)}(\mathbf{g}) = \sum_{b \in \mathcal{B}} Q_b^{y,(2)}(\mathbf{h}) = \mathbf{1}$.
- **Gaussian dimension reduction.** In this step, we apply Lemma 83 to further reduce the number of Gaussian random variables. This is the only part different from the proof in [46].

Let n_0 be determined later. For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$ and $M \in \mathbb{R}^{n \times n_0}$, applying Lemma 83 to $P_a^{x,(2)}(\mathbf{g})$ and $Q_b^{y,(2)}(\mathbf{h})$ with $\delta \leftarrow \delta / (2s^2 t^2)$, $d \leftarrow d$, $n \leftarrow 2(m^2 - 1)(n - h)$, we get joint random matrices $P_a^{x,(3)}(M\tilde{\mathbf{x}})$ and $Q_b^{y,(3)}(M\tilde{\mathbf{y}})$. If we sample $\mathbf{M} \sim \gamma_{n \times n_0}$, by Lemma 83 item 3 we have

$$\sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{M}, \tilde{\mathbf{x}}} \left[\text{Tr } \zeta \left(P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right) \right] = \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{g}} \left[\text{Tr } \zeta \left(P_a^{x,(2)}(\mathbf{g}) \right) \right]$$

and

$$\sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{M}, \tilde{\mathbf{y}}} \left[\text{Tr } \zeta \left(Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right) \right] = \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{h}} \left[\text{Tr } \zeta \left(Q_b^{y,(2)}(\mathbf{h}) \right) \right].$$

Then by Markov's inequality, with probability each at most $1/6$,

$$\sum_{x,a} \mu_A(x) \mathbb{E}_{\tilde{\mathbf{x}}} \left[\text{Tr } \zeta \left(P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right) \right] > 6 \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{g}} \left[\text{Tr } \zeta \left(P_a^{x,(2)}(\mathbf{g}) \right) \right]$$

and

$$\sum_{y,b} \mu_B(y) \mathbb{E}_{\tilde{\mathbf{y}}} \left[\text{Tr } \zeta \left(Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right) \right] > 6 \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{h}} \left[\text{Tr } \zeta \left(Q_b^{y,(2)}(\mathbf{h}) \right) \right].$$

By Lemma 83 item 1, 2, and using a union bound, with probability at least $2/3 - \delta$ the following hold:

1. For any x, y, a, b , $\mathbb{E}_{\tilde{\mathbf{x}}} \left[\left\| P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right\|_2^2 \right] \leq 2$ and $\mathbb{E}_{\tilde{\mathbf{y}}} \left[\left\| Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right\|_2^2 \right] \leq 2$.
2. $\left| \mathbb{E}_{\tilde{\mathbf{x}}, \tilde{\mathbf{y}}} \left[\text{val}_h \left(\left\{ P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right\}, \left\{ Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right\} \right) \right] - \mathbb{E}_{\mathbf{g}, \mathbf{h}} \left[\text{val}_h \left(\left\{ P_a^{x,(2)}(\mathbf{g}) \right\}, \left\{ Q_b^{y,(2)}(\mathbf{h}) \right\} \right) \right] \right| \leq \delta t^2$.

3. $\sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{x}} \left[\text{Tr} \zeta \left(P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right) \right] \leq 6 \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{g}} \left[\text{Tr} \zeta \left(P_a^{x,(2)}(\mathbf{g}) \right) \right]$ and
 $\sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{y}} \left[\text{Tr} \zeta \left(Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right) \right] \leq 6 \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{h}} \left[\text{Tr} \zeta \left(Q_b^{y,(2)}(\mathbf{h}) \right) \right]$.
4. For any x, y , $\sum_{\substack{a \in \mathcal{A} \\ b \in \mathcal{B}}} P_a^{x,(3)}(M\tilde{\mathbf{x}}) = \sum_{b \in \mathcal{B}} Q_b^{y,(3)}(M\tilde{\mathbf{y}}) = \mathbf{1}$.

Here $n_0 = \frac{d^{O(d)} s^{12} t^{12}}{\delta^6}$.

- **Smoothing random matrices.** In this step, we reduce $\deg(P_a^{x,(3)})$ and $\deg(Q_b^{y,(3)})$ for any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$. We apply [46, Lemma 12.1] to $P_a^{x,(3)}(M\tilde{\mathbf{x}})$ and $Q_b^{y,(3)}(M\tilde{\mathbf{y}})$ with $\delta \leftarrow \delta$, $h \leftarrow h$, $n \leftarrow n_0$ and obtain joint random matrices $P_a^{x,(4)}(\mathbf{x}), Q_b^{y,(4)}(\mathbf{y}) \in L^2(\mathcal{H}_m^{\otimes h}, \gamma_{n_0})$ such that the following holds.

1. For any x, y, a, b , the entries of $P_a^{x,(4)}(\mathbf{x})$ and $Q_b^{y,(4)}(\mathbf{y})$ are polynomials of degree at most d .
2. For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, $\mathbb{E}_{\mathbf{x}} \left[\left\| P_a^{x,(4)}(\mathbf{x}) \right\|_2^2 \right] \leq 2$ and $\mathbb{E}_{\mathbf{y}} \left[\left\| Q_b^{y,(4)}(\mathbf{y}) \right\|_2^2 \right] \leq 2$.
3. $\left| \mathbb{E}_{\mathbf{x}, \mathbf{y}} \left[\text{val}_h \left(\left\{ P_a^{x,(4)}(\mathbf{x}) \right\}, \left\{ Q_b^{y,(4)}(\mathbf{y}) \right\} \right) \right] - \mathbb{E}_{\mathbf{x}, \mathbf{y}} \left[\text{val}_h \left(\left\{ P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right\}, \left\{ Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right\} \right) \right] \right| \leq \delta t^2$.
4. $\left| \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{x}} \left[\text{Tr} \zeta \left(P_a^{x,(4)}(\mathbf{x}) \right) \right] - \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{x}} \left[\text{Tr} \zeta \left(P_a^{x,(3)}(M\tilde{\mathbf{x}}) \right) \right] \right| \leq \delta t$ and
 $\left| \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{y}} \left[\text{Tr} \zeta \left(Q_b^{y,(4)}(\mathbf{y}) \right) \right] - \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{y}} \left[\text{Tr} \zeta \left(Q_b^{y,(3)}(M\tilde{\mathbf{y}}) \right) \right] \right| \leq \delta t$.
5. For any x, y , $\sum_{a \in \mathcal{A}} P_a^{x,(4)}(\mathbf{x}) = \sum_{b \in \mathcal{B}} Q_b^{y,(4)}(\mathbf{y}) = \mathbf{1}$.

- **Multilinearization.** For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, we apply [46, Lemma 13.1] to $P_a^{x,(4)}(\mathbf{x})$ and $Q_b^{y,(4)}(\mathbf{y})$ with $d \leftarrow d$, $\delta \leftarrow \tau$, $h \leftarrow h$, $n \leftarrow n_0$ and obtain joint random matrices $P_a^{x,(5)}(\mathbf{x}), Q_b^{y,(5)}(\mathbf{y}) \in L^2(\mathcal{H}_m^{\otimes h}, \gamma_{n_0 n_1})$ such that the following holds.

1. For any x, y, a, b , the entries of $P_a^{x,(5)}(\mathbf{x})$ and $Q_b^{y,(5)}(\mathbf{y})$ are multilinear polynomials of degree at most d , and every variable in $P_a^{x,(5)}(\mathbf{x})$ and $Q_b^{y,(5)}(\mathbf{y})$ has influence at most τ .
2. For any x, y, a, b , $\mathbb{E}_{\mathbf{x}} \left[\left\| P_a^{x,(5)}(\mathbf{x}) \right\|_2^2 \right] \leq 2$ and $\mathbb{E}_{\mathbf{y}} \left[\left\| Q_b^{y,(5)}(\mathbf{y}) \right\|_2^2 \right] \leq 2$.
3. $\left| \mathbb{E}_{\mathbf{x}, \mathbf{y}} \left[\text{val}_h \left(\left\{ P_a^{x,(5)}(\mathbf{x}) \right\}, \left\{ Q_b^{y,(5)}(\mathbf{y}) \right\} \right) \right] - \mathbb{E}_{\mathbf{x}, \mathbf{y}} \left[\text{val}_h \left(\left\{ P_a^{x,(4)}(\mathbf{x}) \right\}, \left\{ Q_b^{y,(4)}(\mathbf{y}) \right\} \right) \right] \right| \leq \tau t^2$.
4. $\left| \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{x}} \left[\text{Tr} \zeta \left(P_a^{x,(5)}(\mathbf{x}) \right) \right] - \sum_{x,a} \mu_A(x) \mathbb{E}_{\mathbf{x}} \left[\text{Tr} \zeta \left(P_a^{x,(4)}(\mathbf{x}) \right) \right] \right| \leq \tau t$ and
 $\left| \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{y}} \left[\text{Tr} \zeta \left(Q_b^{y,(5)}(\mathbf{y}) \right) \right] - \sum_{y,b} \mu_B(y) \mathbb{E}_{\mathbf{y}} \left[\text{Tr} \zeta \left(Q_b^{y,(4)}(\mathbf{y}) \right) \right] \right| \leq \tau t$.
5. For any x, y , $\sum_{a \in \mathcal{A}} P_a^{x,(5)}(\mathbf{x}) = \sum_{b \in \mathcal{B}} Q_b^{y,(5)}(\mathbf{y}) = \mathbf{1}$.
 Here $n_1 = O\left(\frac{d^2}{\tau^2}\right)$.

- **Invariance from $L^2(\mathcal{H}_m^{\otimes h}, \gamma_{n_0 n_1})$ to $\mathcal{H}_m^{\otimes h+n_0 n_1}$.** In this step, we transform all the random matrices from the previous step to matrices without any classical randomness. In particular, we replace all the Gaussian random variables with $n_0 n_1$ quantum registers, so after this step, the number of quantum registers is $h + n_0 n_1$.

For any $(x, y, a, b) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B}$, applying [46, Lemma 10.11] to $P_a^{x,(5)}(\mathbf{x}), Q_b^{y,(5)}(\mathbf{y})$ with $n \leftarrow n_0 n_1$, $h \leftarrow h$, $d \leftarrow 2d$, $\tau \leftarrow \tau$ to get $P_a^{x,(6)}, Q_b^{y,(6)} \in \mathcal{H}_m^{\otimes h+n_0 n_1}$ satisfying the following.

1. For any x, y, a, b , $\left\| \left\| P_a^{x,(6)} \right\| \right\|_2^2 \leq 2$ and $\left\| \left\| Q_b^{y,(6)} \right\| \right\|_2^2 \leq 2$.
 2. $\text{val}_{h+n_0n_1} \left(\left\{ P_a^{x,(6)} \right\}, \left\{ Q_b^{y,(6)} \right\} \right) = \mathbb{E}_{\mathbf{x}, \mathbf{y}} \left[\text{val}_h \left(\left\{ P_a^{x,(5)}(\mathbf{x}) \right\}, \left\{ Q_b^{y,(5)}(\mathbf{y}) \right\} \right) \right]$.
 3. $\sum_{x,a} \mu_A(x) \left| \frac{1}{m^{h+n_0n_1}} \text{Tr } \zeta \left(P_a^{x,(6)} \right) - \frac{1}{m^h} \mathbb{E} \left[\text{Tr } \zeta \left(P_a^{x,(5)}(\mathbf{x}) \right) \right] \right| \leq O \left(t \left(9^d m^d \sqrt{\tau} d \right)^{2/3} \right)$ and $\sum_{y,b} \mu_B(y) \left| \frac{1}{m^{h+n_0n_1}} \text{Tr } \zeta \left(Q_b^{y,(6)} \right) - \frac{1}{m^h} \mathbb{E} \left[\text{Tr } \zeta \left(Q_b^{y,(5)}(\mathbf{y}) \right) \right] \right| \leq O \left(t \left(9^d m^d \sqrt{\tau} d \right)^{2/3} \right)$.
 4. For any x, y , $\sum_{a \in \mathcal{A}} P_a^{x,(6)} = \sum_{b \in \mathcal{B}} Q_b^{y,(6)} = \mathbf{1}$.
- **Rounding.** Note that the matrices from the previous step may not form valid POVMs, so in this step we round them to close POVMs. In this step, the number of quantum registers remains the same as $h + n_0n_1$.
By Lemma 65 there exist operators $\left\{ P_a^{x,(7)} \right\}$ and $\left\{ Q_b^{y,(7)} \right\}$ satisfying for all x

$$\begin{aligned} \sum_a \left\| \left\| P_a^{x,(7)} - P_a^{x,(6)} \right\| \right\|_2^2 &\leq \frac{3(t+1)}{m^D} \sum_a \text{Tr } \zeta \left(P_a^{x,(6)} \right) + 6\sqrt{t} \left(\frac{1}{m^D} \sum_a \text{Tr } \zeta \left(P_a^{x,(6)} \right) \right)^{1/2} \\ &\leq 10t \left(\frac{1}{m^D} \sum_a \text{Tr } \zeta \left(P_a^{x,(6)} \right) \right)^{1/2}. \end{aligned} \quad (23)$$

Similarly, for all y , we have

$$\sum_a \left\| \left\| Q_b^{y,(7)} - Q_b^{y,(6)} \right\| \right\|_2^2 \leq 10t \left(\frac{1}{m^D} \sum_b \text{Tr } \zeta \left(Q_b^{y,(6)} \right) \right)^{1/2}. \quad (24)$$

Then

$$\begin{aligned} &\left| \text{val}_D \left(\left\{ P_a^{x,(7)} \right\}, \left\{ Q_b^{y,(7)} \right\} \right) - \text{val}_D \left(\left\{ P_a^{x,(6)} \right\}, \left\{ Q_b^{y,(6)} \right\} \right) \right| \\ &\leq \left| \text{val}_D \left(\left\{ P_a^{x,(7)} - P_a^{x,(6)} \right\}, \left\{ Q_b^{y,(7)} \right\} \right) \right| + \left| \text{val}_D \left(\left\{ P_a^{x,(6)} \right\}, \left\{ Q_b^{y,(7)} - Q_b^{y,(6)} \right\} \right) \right| \\ &\leq \sum_{x,y,a,b} \mu(x,y) \left(\left\| \left\| P_a^{x,(7)} - P_a^{x,(6)} \right\| \right\|_2 \left\| \left\| Q_b^{y,(7)} \right\| \right\|_2 + \left\| \left\| P_a^{x,(6)} \right\| \right\|_2 \left\| \left\| Q_b^{y,(7)} - Q_b^{y,(6)} \right\| \right\|_2 \right) \\ &\leq \left(\sum_b \sum_{x,a} \mu_A(x) \left\| \left\| P_a^{x,(7)} - P_a^{x,(6)} \right\| \right\|_2^2 \right)^{1/2} \left(\sum_a \sum_{y,b} \mu_B(y) \left\| \left\| Q_b^{y,(7)} \right\| \right\|_2^2 \right)^{1/2} \\ &\quad + \left(\sum_b \sum_{x,a} \mu_A(x) \left\| \left\| P_a^{x,(6)} \right\| \right\|_2^2 \right)^{1/2} \left(\sum_a \sum_{y,b} \mu_B(y) \left\| \left\| Q_b^{y,(7)} - Q_b^{y,(6)} \right\| \right\|_2^2 \right)^{1/2} \\ &\quad \text{(Cauchy Schwarz)} \\ &\leq \sqrt{10}t^2 \left(\sum_x \mu_A(x) \left(\frac{1}{m^D} \sum_a \text{Tr } \zeta \left(P_a^{x,(6)} \right) \right)^{1/2} \right)^{1/2} \\ &\quad + 2\sqrt{5}t^2 \left(\sum_y \mu_B(y) \left(\frac{1}{m^D} \sum_b \text{Tr } \zeta \left(Q_b^{y,(6)} \right) \right)^{1/2} \right)^{1/2} \\ &\leq \sqrt{10}t^2 \left(\frac{1}{m^D} \sum_{x,a} \mu_A(x) \text{Tr } \zeta \left(P_a^{x,(6)} \right) \right)^{1/4} + 2\sqrt{5}t^2 \left(\frac{1}{m^D} \sum_{y,b} \mu_B(y) \text{Tr } \zeta \left(Q_b^{y,(6)} \right) \right)^{1/4}, \end{aligned}$$

where in the second last inequality, we use $\left\| \left\| P_a^{x,(6)} \right\| \right\|_2 \leq 2$, $\left\| \left\| Q_b^{y,(7)} \right\| \right\|_2 \leq 1$, and Equations (23) and (24). The last inequality follows from concavity of the function $x \mapsto \sqrt{x}$.

Keeping track of the parameters in the construction, we can upper bound $\frac{1}{m^D} \sum_{x,a} \mu_A(x) \text{Tr} \zeta \left(P_a^{x,(6)} \right)$ and $\frac{1}{m^D} \sum_{y,b} \mu_B(y) \text{Tr} \zeta \left(Q_b^{y,(6)} \right)$. We choose

$$\delta = \frac{\epsilon^4}{300t^9}, \tau = \frac{\epsilon^{12}}{t^{27}} \exp \left(-\frac{300t^9 \log m}{\epsilon^4(1-\rho)} \log^2 \left(\frac{t}{\epsilon} \right) \right) \quad (25)$$

such that the difference in the game value at the final step matches that of the previous steps, remaining on the order of $O(\delta t^2)$. We conclude that the number of quantum registers is

$$\begin{aligned} D = h + n_0 n_1 &= \frac{d}{\tau} + \frac{d^{O(d)} s^{12} t^{12}}{\delta^6} \cdot O \left(\frac{d^2}{\tau^2} \right) \\ &= O \left(\frac{s^{12} t^{120}}{\epsilon^{48}} \exp \left(\frac{600t^9 \log m}{\epsilon^4(1-\rho)} \log^2 \left(\frac{t}{\epsilon(1-\rho)} \right) \right) \right), \end{aligned}$$

which completes the proof. \blacktriangleleft

Next, we give the non-deterministic algorithm with the following parameters.

$$C_{pt} = 300$$

$$\varepsilon_{rd} = \varepsilon^2 / (4t^3)$$

$$\delta = \frac{\varepsilon_{rd}^2}{C_{pt} t (t+1)}$$

$$d = \frac{C_{sm} \log^2 \frac{1}{\delta}}{\delta \log(1/\rho)} \text{ as in Lemma 62.}$$

$$s_w = D \log m + \log \left(\frac{2}{\delta} \right) \text{ as in Lemma 68.}$$

D = is the polynomial specified in Theorem 28 with $\varepsilon \leftarrow \varepsilon/2$.

Proof of Proposition 26. Consider the algorithm below, with the parameters above.

Input: Parameters in Definition 24.

Certificate: Let $\{(\mathcal{A}_i, \mathcal{B}_i)\}_{i=0}^{m^2-1}$ be a pair of standard orthonormal basis satisfying Fact 46. A tuple of real numbers of width s_w , which are non-zero Fourier coefficients of a degree- d pseudo-strategy on D copies of ψ_{AB} . For each $x \in \mathcal{X}, a \in \mathcal{A}$ and $\sigma \in [m^2]_{\geq 0}^D$ satisfying $|\sigma| \leq d$, the certificate should contain the coefficient $\hat{P}_a^x(\sigma)$. Similarly, for $y \in \mathcal{Y}, b \in \mathcal{B}$ and σ , the certificate should contain the coefficient $\hat{Q}_b^y(\sigma)$. Then the degree- d pseudo-strategy can be written as P_a^x and Q_b^y satisfying

$$P_a^x = \sum_{|\sigma| \leq d} \hat{P}_a^x(\sigma) \mathcal{A}_\sigma \text{ and } Q_b^y = \sum_{|\sigma| \leq d} \hat{Q}_b^y(\sigma) \mathcal{B}_\sigma.$$

Algorithm:

1. Compute the winning probability on the pseudo-strategy, which is

$$\text{val}_D(\{P_a^x\}, \{Q_b^y\}) = \sum_{x,y,a,b} \mu(x,y) \cdot V(x,y,a,b) \sum_{\sigma \in [m^2]_{\geq 0}^D} c_\sigma \hat{P}_a^x(\sigma) \cdot \hat{Q}_b^y(\sigma),$$

where $c_\sigma = c_{\sigma_1} \cdots c_{\sigma_D}$, and $\{c_i\}_{i=0}^{m^2-1}$ is given in Fact 46. Reject if

$$\text{val}_D(\{P_a^x\}, \{Q_b^y\}) < \beta.$$

2. Check if the operators sum up to the identity by checking
 - For all x, y and $\sigma \neq 0^D$, it should hold that

$$\sum_a \widehat{P}_a^x(\sigma) = \sum_b \widehat{Q}_b^y(\sigma) = 0.$$

- For all x, y , and $\sigma = 0^D$, it should hold that

$$\sum_a \widehat{P}_a^x(\sigma) = \sum_b \widehat{Q}_b^y(\sigma) = 1.$$

Reject if any of the above equalities fails.

3. For each x, y, a, b , run the positivity testing algorithm described in Section 4 on P_a^x and Q_b^y with parameters $\beta \leftarrow 4\delta$ and $\delta \leftarrow 2\delta$. Reject if any of the positivity testings fails.
4. Accept.

Time complexity

We upper bound the time complexity of each step.

1. Certificate length: The certificate contains the non-zero Fourier coefficients of degree- d operators acting on D qudits. Each degree- d operator consists of

$$\sum_{d=0}^d \binom{D}{d} \cdot (m^2 - 1)^d \leq d(m^2 - 1)^d D^d$$

coefficients, each s_w bits. Hence, the length of the certificate is $O(stdm^{2d}D^d s_w)$.

2. To compute the game value, we need to enumerate over all x, y, a, b, σ and compute a sum of products. This takes time

$$s^2 t^2 (m^2 - 1)^d D^d.$$

3. Checking if the operators sum up to the identity takes linear time in certificate length as it involves only summation over Fourier coefficients.
4. Each positivity testing takes time as specified in Theorem 23, which is

$$\exp(\text{poly}(m^d, 1/\delta)) \cdot D^{O(d)}.$$

By the choices of parameters, the overall running time is upper bounded by

$$\text{poly}\left(s, \text{eexp}\left(t, \log\left(\frac{1}{\rho}\right), \frac{1}{\varepsilon}\right)\right).$$

Completeness

Suppose $\omega^*(G, \psi_{AB}) \geq \beta + \varepsilon$. Then by Theorem 28, there exists a strategy (P_a^x, Q_b^y) that uses D copies of ψ_{AB} with game value $\text{val}_D(\{P_a^x\}, \{Q_b^y\}) \geq \beta + \varepsilon/2$. Let f be the smoothing map in Lemma 62, and let $P_a^{x,(1)} = f(P_a^x)$ and $Q_b^{y,(1)} = f(Q_b^y)$. Then $\{P_a^{x,(1)}\}, \{Q_b^{y,(1)}\}$ are of degree at most d and satisfy

1. For all x, y , we have $\sum_a P_a^{x,(1)} = \sum_b Q_b^{y,(1)} = \mathbb{1}$ (since f is linear and unital)
2. For all x, y, a, b , $\|P_a^{x,(1)}\|_2 \leq 1$ and $\|Q_b^{y,(1)}\|_2 \leq 1$.
3. For all x, y, a, b , $\left| \text{Tr} \left(\left(P_a^{x,(1)} \otimes Q_b^{y,(1)} \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^x \otimes Q_b^y \right) \psi_{AB}^{\otimes n} \right) \right| \leq \delta$.
4. For all x, y, a, b , $m^{-D} \text{Tr} \zeta \left(P_a^{x,(1)} \right) \leq \delta$ and $m^{-D} \text{Tr} \zeta \left(Q_b^{y,(1)} \right) \leq \delta$.

We observe that Lemma 62 also guarantees the Fourier coefficients of $P_a^{x,(1)}$ and $Q_b^{y,(1)}$ have absolute values bounded by 1. This allows us to truncate the strategy. For each Fourier coefficient we preserve s_w digits and by Lemma 68 get $\{P_a^{x,(2)}\}, \{Q_b^{y,(2)}\}$ satisfying

1. For all x, y , $\sum_a P_a^{x,(2)} = \sum_b Q_b^{y,(2)} = \mathbb{1}$.
2. For all x, y, a, b , $\|P_a^{x,(2)}\|_2 \leq 1$ and $\|Q_b^{y,(2)}\|_2 \leq 1$;
3. For all x, y, a, b , $\left| \text{Tr} \left(\left(P_a^{x,(2)} \otimes Q_b^{y,(2)} \right) \psi_{AB}^{\otimes n} \right) - \text{Tr} \left(\left(P_a^{x,(1)} \otimes Q_b^{y,(1)} \right) \psi_{AB}^{\otimes n} \right) \right| \leq \delta$,
4. For all x, y, a, b , $m^{-D} \text{Tr} \zeta \left(P_a^{x,(2)} \right) \leq 2\delta$ and $m^{-D} \text{Tr} \zeta \left(Q_b^{y,(2)} \right) \leq 2\delta$.

This pseudo-strategy is the certificate. Specifically, by Lemma 67 the game value is

$$\text{val}_D \left(\{P_a^{x,(2)}\}, \{Q_b^{y,(2)}\} \right) \geq \beta + \varepsilon/2 - 2\delta t^2 = \beta + \varepsilon/2 - \frac{\varepsilon^2}{2tC_{pt}} \geq \beta,$$

and the first check is passed. Also, by item 4, the positivity testings can also be passed.

Soundness

Suppose that there exists a certificate that passes all the testings, then there exists a degree- d pseudo-strategy $\{P_a^{x,(1)}\}, \{Q_b^{y,(1)}\}$ satisfying

- By the game value testing,

$$\text{val}_D \left(\{P_a^{x,(1)}\}, \{Q_b^{y,(1)}\} \right) \geq \beta.$$

- By “summing up to the identity” testings, for all x, y

$$\sum_a P_a^{x,(1)} = \mathbb{1}, \text{ and } \sum_b Q_b^{y,(1)} = \mathbb{1}.$$

- By the positivity testings, for all x, y, a, b

$$\frac{1}{m^D} \text{Tr} \zeta \left(P_a^{x,(1)} \right) \leq 6\delta, \text{ and } \frac{1}{m^D} \text{Tr} \zeta \left(Q_b^{y,(1)} \right) \leq 6\delta.$$

We then apply Lemma 65 to get a strategy $\{P_a^{x,(2)}\}$ and $\{Q_b^{y,(2)}\}$. It holds that for each $x \in \mathcal{X}$

$$\begin{aligned} \sum_{a \in \mathcal{A}} \|P_a^{x,(2)} - P_a^{x,(1)}\|_2^2 &\leq 3(t+1) \left(\sum_{a \in \mathcal{A}} \frac{1}{m^D} \text{Tr} \zeta \left(P_a^{x,(1)} \right) \right) + 6\sqrt{t} \left(\sum_{a \in \mathcal{A}} \frac{1}{m^D} \text{Tr} \zeta \left(P_a^{x,(1)} \right) \right)^{1/2} \\ &\leq 18t(t+1)\delta + 6\sqrt{6}t\sqrt{\delta} \leq \frac{18\varepsilon_{rd}^2}{C_{pt}} + \frac{6\sqrt{6}\varepsilon_{rd}}{\sqrt{C_{pt}}} \leq \frac{18 + 6\sqrt{6}C_{pt}}{C_{pt}} \varepsilon_{rd} \leq \varepsilon_{rd}. \end{aligned}$$

Similarly, for each $y \in \mathcal{Y}$ we have

$$\sum_{b \in \mathcal{B}} \|Q_b^{y,(2)} - Q_b^{y,(1)}\|_2^2 \leq \varepsilon_{rd}.$$

We get a strategy $\{P_a^{x,(2)}\}$ and $\{Q_b^{y,(2)}\}$ with game value

$$\begin{aligned}
 & \left| \text{val}_D \left(\{P_a^{x,(2)}\}, \{Q_b^{y,(2)}\} \right) - \text{val}_D \left(\{P_a^{x,(1)}\}, \{Q_b^{y,(1)}\} \right) \right| \\
 & \leq \left| \text{val}_D \left(\{P_a^{x,(2)} - P_a^{x,(1)}\}, \{Q_b^{y,(2)}\} \right) \right| + \left| \text{val}_D \left(\{P_a^{x,(1)}\}, \{Q_b^{y,(2)} - Q_b^{y,(1)}\} \right) \right| \\
 & \leq \sum_{x,y,a,b} \mu(x,y) \left(\left\| P_a^{x,(2)} - P_a^{x,(1)} \right\|_2 \left\| Q_b^{y,(2)} \right\|_2 + \left\| P_a^{x,(1)} \right\|_2 \left\| Q_b^{y,(2)} - Q_b^{y,(1)} \right\|_2 \right) \\
 & \leq \left(\sum_b \sum_{x,a} \mu_A(x) \left\| P_a^{x,(2)} - P_a^{x,(1)} \right\|_2^2 \right)^{1/2} \left(\sum_a \sum_{y,b} \mu_B(y) \left\| Q_b^{y,(2)} \right\|_2^2 \right)^{1/2} \\
 & \quad + \left(\sum_b \sum_{x,a} \mu_A(x) \left\| P_a^{x,(1)} \right\|_2^2 \right)^{1/2} \left(\sum_a \sum_{y,b} \mu_B(y) \left\| Q_b^{y,(2)} - Q_b^{y,(1)} \right\|_2^2 \right)^{1/2} \\
 & \quad \text{(Cauchy-Schwarz)} \\
 & \leq 2t\sqrt{t\varepsilon_{rd}}.
 \end{aligned}$$

Thus there exists a strategy with game value

$$\text{val}_D \left(\{P_a^{x,(2)}\}, \{Q_b^{y,(2)}\} \right) > \beta - 2t\sqrt{t\varepsilon_{rd}} = \beta - \varepsilon. \quad \blacktriangleleft$$

B.3.2 NP-hardness

Proof of Proposition 29. The noisy MIP* verifier V^* from an MIP verifier $V = (\text{Alg}_Q, \text{Alg}_V)$

Setup: Flip two unbiased coins $\mathbf{b}, \mathbf{c} \sim \{0, 1\}$. Sample questions $(\mathbf{x}, \mathbf{y}) \sim \text{Alg}_Q(\text{input})$.

With probability 1/2 each, perform one of the following ten tests.

Verify: Distribute the questions as follows

- Player \mathbf{b} : give \mathbf{x} ; receive \mathbf{a} .
 - Player $\bar{\mathbf{b}}$: give \mathbf{y} ; receive \mathbf{b}
- Accept if $V(\text{input}, \mathbf{x}, \mathbf{y})$ accepts on \mathbf{a}, \mathbf{b} .

Consistency: Distribute the questions as follows: if $\mathbf{c} = 0$

- Player \mathbf{b} : give \mathbf{x} ; receive \mathbf{a} ,
 - Player $\bar{\mathbf{b}}$: give \mathbf{x} ; receive \mathbf{b} ,
- otherwise
- Player \mathbf{b} : give \mathbf{y} ; receive \mathbf{a} ,
 - Player $\bar{\mathbf{b}}$: give \mathbf{y} ; receive \mathbf{b} ,
- Accept if $\mathbf{a} = \mathbf{b}$.

Completeness. If input is satisfiable, the value-1 strategy for V is also a value-1 strategy for V^* .

Soundness. In the consistency test, with probability 1/2 both provers get a question x . Hence the probability for the provers to pass the consistency test of x is at least $1 - 4\epsilon$. If Alice and Bob sharing n copies of a noisy m -dimensional MES ψ_{AB} , it means that

$$\mathbb{E}_x \sum_{a \in \mathcal{A}} \text{Tr} \left((P_a^x \otimes P_a^x) \psi_{AB}^{\otimes n} \right) \geq 1 - 4\epsilon.$$

Using the Fourier expansion of $P_a^x = \sum_{\sigma} \widehat{P}_a^x(\sigma) \mathcal{P}_{\sigma}$, the condition above is equivalent to

$$\mathbb{E}_x \sum_a \sum_{\sigma} \rho^{|\sigma|} \widehat{P}_a^x(\sigma)^2 \geq 1 - 4\epsilon.$$

Notice that $\|P_a^x\|_2^2 = \sum_{\sigma} \widehat{P}_a^x(\sigma)^2$, and for all x , $\sum_a \|P_a^x\|_2^2 \leq 1$. Hence

$$\begin{aligned} \mathbb{E}_x \sum_a \sum_{\sigma} \rho^{|\sigma|} \widehat{P}_a^x(\sigma)^2 &\leq \mathbb{E}_x \sum_a \left[\widehat{P}_a^x(\emptyset)^2 + \rho \sum_{\sigma \neq \emptyset} \widehat{P}_a^x(\sigma)^2 \right] \\ &\leq \mathbb{E}_x \sum_a \left[\widehat{P}_a^x(\emptyset)^2 + \rho (\|P_a^x\|_2^2 - \widehat{P}_a^x(\emptyset)^2) \right] \\ &\leq \rho + (1 - \rho) \mathbb{E}_x \sum_a \widehat{P}_a^x(\emptyset)^2. \end{aligned}$$

Therefore,

$$\mathbb{E}_x \sum_a \widehat{P}_a^x(\emptyset)^2 \geq 1 - \frac{4\epsilon}{1 - \rho}. \quad (26)$$

On the other hand, for all x , $\sum_a \widehat{P}_a^x(\emptyset) = 1$. For each x , let a_x be the answer that maximizes $\widehat{P}_a^x(\emptyset)$. Then $\sum_a \widehat{P}_a^x(\emptyset)^2 \leq \widehat{P}_{a_x}^x(\emptyset) \sum_a \widehat{P}_a^x(\emptyset) = \widehat{P}_{a_x}^x(\emptyset)$, and

$$\mathbb{E}_x \widehat{P}_{a_x}^x(\emptyset) \geq 1 - \frac{4\epsilon}{1 - \rho}.$$

Similarly, for each y , let b_y be the answer that maximizes $\widehat{Q}_b^y(\emptyset)$, and then

$$\mathbb{E}_{x,y} \widehat{Q}_{b_y}^y(\emptyset) \geq 1 - \frac{4\epsilon}{1 - \rho}.$$

In the deterministic strategy, Alice answers a_x for question x and Bob answers b_y for question y . The difference in the probability of satisfying V between the original strategy and the deterministic strategy is

$$\begin{aligned} & \left| \mathbb{E}_{x,y} \sum_{a,b} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] V(x, y, a, b) - \mathbb{E}_{x,y} V(x, y, a_x, b_y) \right| \\ &= \mathbb{E}_{x,y} \left(1 - \text{Tr} [(P_{a_x}^x \otimes Q_{b_y}^y) \psi_{AB}^{\otimes n}] \right) V(x, y, a_x, b_y) \\ &\quad + \mathbb{E}_{x,y} \sum_{\substack{a \neq a_x \text{ or} \\ b \neq b_y}} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] V(x, y, a, b) \\ &\leq \mathbb{E}_{x,y} \left(1 - \text{Tr} [(P_{a_x}^x \otimes Q_{b_y}^y) \psi_{AB}^{\otimes n}] \right) + \mathbb{E}_{x,y} \sum_{\substack{a \neq a_x \text{ or} \\ b \neq b_y}} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] \end{aligned}$$

where we use the fact that $V(x, y, a, b) \leq 1$ for all x, y, a, b .

Writing $1 = \sum_{a,b} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}]$, we get that the expression above equals

$$\begin{aligned} & 2 \mathbb{E}_{x,y} \sum_{\substack{a \neq a_x \text{ or} \\ b \neq b_y}} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] \\ &= 2 \mathbb{E}_{x,y} \sum_{a \neq a_x, b} \text{Tr} [(P_a^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] + 2 \mathbb{E}_{x,y} \sum_{b \neq b_y} \text{Tr} [(P_{a_x}^x \otimes Q_b^y) \psi_{AB}^{\otimes n}] \\ &\leq 2 \mathbb{E}_{x,y} \left[\sum_{a \neq a_x} \widehat{P}_a^x(\emptyset) + \sum_{b \neq b_y} \widehat{Q}_b^y(\emptyset) \right] \\ &\leq \frac{16\epsilon}{1 - \rho}. \end{aligned}$$

The probability for the original strategy to satisfy V is at least $1 - 2\varepsilon$, so the probability for the deterministic strategy to satisfy V is at least $1 - 2\varepsilon - 16\varepsilon/(1 - \rho)$. ◀

B.4 MIP* Protocol for RE with $O(1)$ -size Answers

Subset tester for the Hadamard code

Let $k \leq n$ and D be a distribution on the subsets of \mathbb{F}_2^n with size k . Flip an unbiased coin $\mathbf{b} \sim \{0, 1\}$. Sample $\mathbf{F} = \{x_1, \dots, x_k\} \sim D$ and a uniformly random $\mathbf{y} \in \mathbb{F}_2^n$. Perform one of the following three subtests with equal probability.

Subtest 1: Perform one of the following checks with equal probability.

Check 1: Distribute the question as follows:

- Player \mathbf{b} : give \mathbf{F} and \mathbf{y} ; receive $(\mathbf{a}_1, \dots, \mathbf{a}_k, \mathbf{c}, \mathbf{a}'_1, \dots, \mathbf{a}'_k) \in \mathbb{F}_2^{2k+1}$.
 - Player $\bar{\mathbf{b}}$: give \mathbf{F} , receive $(\mathbf{d}_1, \dots, \mathbf{d}_k) \in \mathbb{F}_2^k$.
- Accept if $\mathbf{a}_i + \mathbf{c} = \mathbf{a}'_i$ and $\mathbf{a}_i = \mathbf{d}_i$ for all i .

Check 2: Distribute the question as follows:

- Player \mathbf{b} : give \mathbf{F} and \mathbf{y} ; receive $(\mathbf{a}_1, \dots, \mathbf{a}_k, \mathbf{c}, \mathbf{a}'_1, \dots, \mathbf{a}'_k) \in \mathbb{F}_2^{2k+1}$.
 - Player $\bar{\mathbf{b}}$: give \mathbf{y} , receive $\mathbf{e} \in \mathbb{F}_2$.
- Accept if $\mathbf{a}_i + \mathbf{c} = \mathbf{a}'_i$ for all i , and $\mathbf{e} = \mathbf{c}$.

Check 3: Distribute the question as follows:

- Player \mathbf{b} : give \mathbf{F} and \mathbf{y} ; receive $(\mathbf{a}_1, \dots, \mathbf{a}_k, \mathbf{c}, \mathbf{a}'_1, \dots, \mathbf{a}'_k) \in \mathbb{F}_2^{2k+1}$.
 - Player $\bar{\mathbf{b}}$: give $\mathbf{F} + \mathbf{y} = \{\mathbf{x}_1 + \mathbf{y}, \dots, \mathbf{x}_k + \mathbf{y}\}$, receive $(\mathbf{d}_1, \dots, \mathbf{d}_k) \in \mathbb{F}_2^k$.
- Accept if $\mathbf{a}_i + \mathbf{c} = \mathbf{a}'_i$ and $\mathbf{a}'_i = \mathbf{d}_i$ for all i .

Subtest 2: Distribute the question as follows:

- Player \mathbf{b} : give $\mathbf{F} + \mathbf{y} = \{\mathbf{x}_1 + \mathbf{y}, \dots, \mathbf{x}_k + \mathbf{y}\}$; receive $(\mathbf{a}_1, \dots, \mathbf{a}_k)$.
 - Player $\bar{\mathbf{b}}$: give $\mathbf{x}_i + \mathbf{y}$ for a random i , receive \mathbf{d} .
- Accept if $\mathbf{a}_i = \mathbf{d}$.

Subtest 3: Perform one of the following checks with equal probability

Check 1: Distribute the question as follows:

- Player \mathbf{b} : give \mathbf{F} ; receive $(\mathbf{a}_1, \dots, \mathbf{a}_k)$.
 - Player $\bar{\mathbf{b}}$: give \mathbf{F} ; receive $(\mathbf{d}_1, \dots, \mathbf{d}_k)$.
- Accept if $\mathbf{a}_i = \mathbf{d}_i$ for all i .

Check 2: Distribute the question as follows:

- Player \mathbf{b} : give $\mathbf{x}_i + \mathbf{y}$ for a random i ; receive \mathbf{a} .
 - Player $\bar{\mathbf{b}}$: give $\mathbf{x}_i + \mathbf{y}$ for a random i ; receive \mathbf{d} .
- Accept if $\mathbf{a} = \mathbf{d}$.

Proof of Proposition 31. Let $\mathbf{F} + \mathbf{y} = (x_1 + y, \dots, x_k + y)$. Let

$$\Omega = \{(a, c, a') \mid a_i + c = a'_i \text{ for all } i \in [k]\}.$$

The set Ω is the set of valid answer tuples for Alice in **Subtest 1**; we also use Ω to denote the *event* that Alice's answers are valid. Winning the subset tester with probability $1 - \varepsilon$ implies that winning each subtest with a probability of at least $1 - 3\varepsilon$. Furthermore, winning **Subtest 1** with a probability of at least $1 - 3\varepsilon$ implies that when Alice gets question (\mathbf{F}, \mathbf{y}) and Bob gets Player 1's questions:

$$\begin{aligned}
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[a_1 = b_1 \wedge \dots \wedge a_k = b_k \wedge \Omega \mid q_A = (F, y), q_B = F] \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[c = d \wedge \Omega \mid q_A = (F, y), q_B = y] \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[a'_1 = b_1 \wedge \dots \wedge a'_k = b_k \wedge \Omega \mid q_A = (F, y), q_B = F + y] \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[\Omega \mid q_A = (F, y)] \geq 1 - 6\varepsilon,
\end{aligned}$$

for all $i \in [k]$; winning **Subtest 2** with a probability of at least $1 - 3\varepsilon$ implies that when Alice gets Player 0's question and Bob gets Player 1's question

$$\mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[a_i = d \mid q_A = F + y, q_B = x_i + y] \geq 1 - 6k\varepsilon;$$

and winning **Subtest 3** with a probability of at least $1 - 3\varepsilon$ implies that when Alice gets Player 0's question and Bob gets Player 1's question

$$\begin{aligned}
& \mathbb{E}_{F \sim D} \Pr[a_1 = b_1 \wedge \dots \wedge a_k = b_k \mid q_A = q_B = F] \geq 1 - 12\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \sim D_{\text{Unif}}} \Pr[a = b \mid q_A = q_B = x_i + y] \geq 1 - 12k\varepsilon \quad \text{for all } i.
\end{aligned}$$

In terms of the measurements and the state $|\psi\rangle$, these conditions are equivalent to

$$\begin{aligned}
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{\substack{a, c, a': \\ a_i + c = a'_i \forall i}} \langle \psi \mid M_{a, c, a'}^{F, y} \otimes N_a^F \mid \psi \rangle \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{\substack{a, c, a': \\ a_i + c = a'_i \forall i}} \langle \psi \mid M_{a, c, a'}^{F, y} \otimes N_c^y \mid \psi \rangle \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{\substack{a, c, a': \\ a_i + c = a'_i \forall i}} \langle \psi \mid M_{a, c, a'}^{F, y} \otimes N_{a'}^{F+y} \mid \psi \rangle \geq 1 - 18\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{\substack{a, c, a': \\ a_i + c = a'_i \forall i}} \langle \psi \mid M_{a, c, a'}^{F, y} \otimes \mathbb{1}_B \mid \psi \rangle \geq 1 - 6\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{a \in \mathbb{F}_2^k} \langle \psi \mid N_a^{F+y} \otimes N_{a_i}^{x_i+y} \mid \psi \rangle \geq 1 - 6k\varepsilon \quad \text{for all } i \\
& \mathbb{E}_{F \sim D} \sum_{a \in \mathbb{F}_2^k} \langle \psi \mid N_a^F \otimes N_a^F \mid \psi \rangle \geq 1 - 12\varepsilon \\
& \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{a \in \mathbb{F}_2} \langle \psi \mid N_a^{x_i+y} \otimes N_a^{x_i+y} \mid \psi \rangle \geq 1 - 12k\varepsilon \quad \text{for all } i.
\end{aligned}$$

We define binary observables

$$\begin{aligned}
M^{x_i | F, y} &= \sum_{a, c, a'} (-1)^{a_i} M_{a, c, a'}^{F, y} & M^{y | F, y} &= \sum_{a, c, a'} (-1)^c M_{a, c, a'}^{F, y} & M^{x_i + y | F, y} &= \sum_{a, c, a'} (-1)^{a'_i} M_{a, c, a'}^{F, y} \\
N^{x_i | F} &= \sum_b (-1)^{b_i} N_b^F & N^y &= N_0^y - N_1^y & N^{x_i + y | F + y} &= \sum_b (-1)^{b_i} N_b^{F+y}.
\end{aligned}$$

We can prove

$$\begin{aligned}
 & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | M^{x_i|F,y} \otimes N^{x_i|F} | \psi \rangle \\
 = & \mathbb{E}_{x \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \left[\Pr[a_i = b_i \wedge \Omega \mid q_A = (F, y), q_B = F] \right. \\
 & \left. - (\Pr[a_i \neq b_i \mid q_A = (F, y), q_B = F] - \Pr[a_i = b_i \wedge \bar{\Omega} \mid q_A = (F, y), q_B = F]) \right] \\
 \geq & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \left[\Pr[a_i = b_i \wedge \Omega \mid q_A = (F, y), q_B = F] \right. \\
 & \left. - (1 - \Pr[a_i = b_i \wedge \Omega \mid q_A = (F, y), q_B = F]) \right] \\
 = & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \left[2\Pr[a_i = b_i \wedge \Omega \mid q_A = (F, y), q_B = F] - 1 \right] \\
 \geq & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \left[2\Pr[a_1 = b_1 \wedge \dots \wedge a_k = b_k \wedge \Omega \mid q_A = (F, y), q_B = F] - 1 \right] \\
 \geq & 1 - 36\varepsilon,
 \end{aligned}$$

which implies that $\mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|M^{x_i|F,y} \otimes \mathbb{1}_B | \psi \rangle - \mathbb{1}_A \otimes N^{x_i|F} | \psi \rangle\|^2 \leq 72\varepsilon$ by expanding the vector norm. Similarly, from the two other checks of **Subtest 1**,

$$\begin{aligned}
 & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|M^{y|F,y} \otimes \mathbb{1}_B | \psi \rangle - \mathbb{1}_A \otimes N^y | \psi \rangle\|^2 \leq 72\varepsilon \\
 & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|M^{x_i+y|F,y} \otimes \mathbb{1}_B | \psi \rangle - \mathbb{1}_A \otimes N^{x_i+y|F+y} | \psi \rangle\|^2 \leq 72\varepsilon.
 \end{aligned}$$

Applying a similar argument to the probability of the event Ω , we can also show

$$\begin{aligned}
 & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | M^{x_i|F,y} M^{y|F,y} M^{x_i+y|F,y} \otimes \mathbb{1}_B | \psi \rangle \\
 = & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \sum_{a,c,a'} (-1)^{a_i+c+a'_i} \langle \psi | M_{a,c,a'}^{F,y} \otimes \mathbb{1}_B | \psi \rangle \\
 = & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} 2\Pr[a_i + c = a'_i \mid q_A = (F, y)] - 1 \\
 \geq & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} 2\Pr[\Omega \mid q_A = (F, y)] - 1 \geq 1 - 12\varepsilon.
 \end{aligned}$$

Next, we would like to replace $M^{x_i|F,y}$ by $N^{x_i|F}$, $M^{y|F,y}$ by N^y and $M^{x_i+y|F,y}$ by $N^{x_i+y|F+y}$ and show

$$\left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | \mathbb{1}_A \otimes N^{x_i+y|F+y} N^y N^{x_i|F} | \psi \rangle - 1 \right| \leq 38\sqrt{\varepsilon}. \quad (27)$$

In the first step

$$\begin{aligned}
 & \left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | M^{x_i|F,y} M^{y|F,y} (M^{x_i+y|F,y} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes N^{x_i+y|F+y}) | \psi \rangle \right| \\
 \leq & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|M^{y|F,y} M^{x_i|F,y} \otimes \mathbb{1}_B | \psi \rangle\| \cdot \|(M^{x_i+y|F,y} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes N^{x_i+y|F+y}) | \psi \rangle\| \\
 & \quad \text{(Cauchy-Schwarz)} \\
 = & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|(M^{x_i+y|F,y} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes N^{x_i+y|F+y}) | \psi \rangle\| \\
 \leq & \sqrt{\mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|(M^{x_i+y|F,y} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes N^{x_i+y|F+y}) | \psi \rangle\|^2} \quad \text{(Jensen)} \\
 \leq & 6\sqrt{2\varepsilon}.
 \end{aligned}$$

Similarly,

$$\begin{aligned} & \left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | M^{x_i|F,y} \otimes N^{x_i+y|F,y} \cdot (M^{y|F,y} \otimes \mathbf{1}_B - \mathbf{1}_A \otimes N^y) | \psi \rangle \right| \leq 6\sqrt{2\varepsilon} \\ & \left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | \mathbf{1}_A \otimes N^{x_i+y|F+y} N^y \cdot (M^{x_i|F,y} \otimes \mathbf{1}_B - \mathbf{1}_A \otimes N^{x_i|F}) | \psi \rangle \right| \leq 6\sqrt{2\varepsilon}. \end{aligned}$$

Hence

$$\left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | \mathbf{1}_A \otimes N^{x_i+y|F+y} N^y N^{x_i|F} | \psi \rangle - 1 \right| \leq 18\sqrt{2\varepsilon} + 12\varepsilon \leq 38\sqrt{\varepsilon}.$$

On the other hand, from **Subtest 2**, we have that for all $i \in [k]$

$$\begin{aligned} & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | N^{x_i+y|F+y} \otimes N^{x_i+y} | \psi \rangle \\ & = 2 \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \Pr[a_i = b \mid q_A = F + y, q_B = x_i + y] - 1 \geq 1 - 12k\varepsilon, \end{aligned}$$

which implies that

$$\mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|(N^{x_i+y|F+y} \otimes \mathbf{1}_B - \mathbf{1}_A \otimes N^{x_i+y}) | \psi \rangle\|^2 \leq 24k\varepsilon.$$

From **Subtest 3**, with similar reasoning we know

$$\begin{aligned} & \mathbb{E}_{F \sim D} \|(N^{x_i|F} \otimes \mathbf{1}_B - \mathbf{1}_A \otimes N^{x_i|F}) | \psi \rangle\|^2 \leq 48\varepsilon \\ & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \|(N^{x_i+y} \otimes \mathbf{1}_B - \mathbf{1}_A \otimes N^{x_i+y}) | \psi \rangle\|^2 \leq 48k\varepsilon \quad \text{for all } i. \end{aligned}$$

Then

$$\begin{aligned} & \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | \mathbf{1}_A \otimes N^{x_i+y|F+y} N^y N^{x_i|F} | \psi \rangle \\ & \approx \sqrt{24k\varepsilon} \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | N^{x_i+y} \otimes N^y N^{x_i|F} | \psi \rangle \\ & \approx \sqrt{48\varepsilon} \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | N^{x_i+y} N^{x_i|F} \otimes N^y | \psi \rangle \\ & \approx \sqrt{48k\varepsilon} \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | N^{x_i|F} \otimes N^{x_i+y} N^y | \psi \rangle \end{aligned}$$

Hence Equation (27) implies that

$$\left| \mathbb{E}_{F \sim D} \mathbb{E}_{y \in D_{\text{Unif}}} \langle \psi | N^{x_i|F} \otimes N^{x_i+y} N^y | \psi \rangle - 1 \right| \leq (45 + 12\sqrt{k})\sqrt{\varepsilon}. \quad (28)$$

Let $C_1 := 45 + 12\sqrt{k}$. Let $\tilde{N}_u = \frac{1}{2^n} \sum_{y \in \mathbb{F}_2^n} (-1)^{u \cdot y} N^y$ and $G_u = (\tilde{N}_u)^2$. Since each N^y is a binary observable, $\{G_u\}$ is a PÖVM. It can be checked that $N^y = \sum_{u \in \mathbb{F}_2^n} (-1)^{u \cdot y} \tilde{N}_u$. Averaging over $F \sim D$, the consistency between $\{N_0^{x_i|F}, N_1^{x_i|F}\}$ and $\{\sum_{u:u \cdot x_i=0} G_u, \sum_{u:u \cdot x_i=1} G_u\}$ is

$$\begin{aligned} & \mathbb{E}_{F \sim D} \frac{1}{2} (1 + \langle \psi | \sum_u (-1)^{u \cdot x_i} N^{x_i|F} \otimes G_u | \psi \rangle) \\ & = \frac{1}{2} + \frac{1}{2} \langle \psi | \mathbb{E}_{F \sim D} \mathbb{E}_{y,z \in D_{\text{Unif}}} \sum_u (-1)^{u \cdot (x_i+y+z)} N^{x_i|F} \otimes N^y N^z | \psi \rangle \\ & = \frac{1}{2} + \frac{1}{2} \langle \psi | \mathbb{E}_{F \sim D} \mathbb{E}_{z \in D_{\text{Unif}}} N^{x_i|F} \otimes N^{x_i+z} N^z | \psi \rangle \approx \frac{C_1}{2} \sqrt{\varepsilon} \cdot 1, \end{aligned}$$

which follows Equation (28). We consider the Naimark's dialation of $\{G_u\}$ on $\mathcal{H} \otimes \mathcal{H}'$ denoted by $\{\hat{G}_u\}$, which is a projective measurement. There exists $|aux\rangle \in \mathcal{H}'$ such that averaging over $F \sim D$, the consistency between $\{N_0^{x_i|F} \otimes \mathbb{1}_{\mathcal{H}'}, N_1^{x_i|F} \otimes \mathbb{1}_{\mathcal{H}'}\}$ and $\{\sum_{u:u \cdot x_i=0} \hat{G}_u, \sum_{u:u \cdot x_i=1} \hat{G}_u\}$ with respect to $|\psi'\rangle = |\psi\rangle \otimes |aux\rangle \otimes |aux\rangle$ is

$$\begin{aligned} & \mathbb{E}_{F \sim D} \sum_{a=0,1} \langle \psi' | (N_a^{x_i|F} \otimes \mathbb{1}_{\mathcal{H}'}) \otimes \left(\sum_{u:u \cdot x_i=a} \hat{G}_u \right) | \psi' \rangle \\ &= \mathbb{E}_{F \sim D} \sum_{a=0,1} \langle \psi | N_a^{x_i|F} \otimes \left(\sum_{u:u \cdot x_i=a} (\mathbb{1} \otimes \langle aux |) \hat{G}_u (\mathbb{1} \otimes |aux\rangle) \right) | \psi \rangle \\ &= \mathbb{E}_{F \sim D} \sum_{a=0,1} \langle \psi | N_a^{x_i|F} \otimes \left(\sum_{u:u \cdot x_i=a} G_u \right) | \psi \rangle \\ &\approx_{C_1/2\sqrt{\varepsilon}} \mathbb{1}. \end{aligned}$$

Since both $\{N_a^{x_i|F} \otimes \mathbb{1}_{\mathcal{H}'}\}$ and $\{\sum_{u:u \cdot x_i=a} \hat{G}_u\}$ are projective measurements, their consistency implies that

$$\mathbb{E}_{F \sim D} \sum_{d=0,1} \|N_d^{x_i|F} \otimes \mathbb{1}_{\mathcal{H}'} |\psi'\rangle - \sum_{u:u \cdot x_i=d} \hat{G}_u |\psi'\rangle\|^2 \leq C_1 \sqrt{\varepsilon}.$$

Next, notice that

$$N_a^F = N_{a_k}^{x_k|F} \dots N_{a_1}^{x_1|F} \text{ and } \sum_{\substack{u:u \cdot x_i=a_i \\ \forall i \in [k]}} \hat{G}_u = \left(\sum_{u:u \cdot x_k=a_k} \hat{G}_u \right) \dots \left(\sum_{u:u \cdot x_1=a_1} \hat{G}_u \right) \dots \left(\sum_{u:u \cdot x_k=a_k} \hat{G}_u \right)$$

Then by Lemma 74

$$\mathbb{E}_{F \sim D} \sum_{a \in \mathbb{F}_2^k} \|N_a^F \otimes \mathbb{1}_{\mathcal{H}'} \otimes \mathbb{1}_B |\tilde{\psi}\rangle - \mathbb{1}_A \otimes \sum_{\substack{u:u \cdot x_i=a_i \\ \forall i \in [k]}} \hat{G}_u |\tilde{\psi}\rangle\|^2 \leq (2k-1)^2 C_1 \sqrt{\varepsilon},$$

which completes the proof. ◀

The answer reduced verifier V^{AR}

Setup Flip two unbiased coins $\mathbf{b}, \mathbf{c} \sim \{0, 1\}$. Sample questions $(\mathbf{x}_0, \mathbf{x}_1) \sim \text{Alg}_Q(\text{input})$.

Sample a view $\mathbf{I}_0, \mathbf{I}_1, \mathbf{J} \sim V_{\text{PCPP}}(\text{input}, \mathbf{x}_0, \mathbf{x}_1)$. Set $\mathbf{J}' = \mu_{\ell_2}(\mathbf{J})$. Randomly select $\mathbf{I}'_0, \mathbf{I}'_1 \subseteq [2^{\ell_1}]$ and $\mathbf{J}'' \subseteq [2^{\ell_2}]$ such that $|\mathbf{I}'_0| = |\mathbf{I}'_1| = |\mathbf{J}''| = \kappa$, which is a sufficiently large constant. Details about how to choose κ can be found in the proof below. Set $\mathbf{T}_0 = \mathbf{I}_0 \cup \mathbf{I}'_0$, $\mathbf{T}_1 = \mathbf{I}_1 \cup \mathbf{I}'_1$ and $\mathbf{U} = \mathbf{J}' \cup \mathbf{J}''$.

With probability 1/10 each, perform one of the following ten tests.

Verify : Distribute the questions as follows:

- Player \mathbf{b} : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{T}_0, \mathbf{T}_1, \mathbf{U}$; receive $\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2$.
- Accept if $V_{\text{PCPP}}(\text{input}, \mathbf{x}_0, \mathbf{x}_1)$ accepts on $\mathbf{a}_0|_{\mathbf{I}_0}$, $\mathbf{a}_1|_{\mathbf{I}_1}$ and $\mathbf{a}_2|_{\mathbf{J}'}$.

Cross check:

Consistency test: Distribute the questions as follows:

- Player \mathbf{b} : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{T}_0, \mathbf{T}_1, \mathbf{U}$; receive $\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2$.

- Player \bar{b} : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{T}_0, \mathbf{T}_1, \mathbf{U}$; receive $\mathbf{a}'_0, \mathbf{a}'_1, \mathbf{a}'_2$

Accept if $\mathbf{a}_0 = \mathbf{a}'_0, \mathbf{a}_1 = \mathbf{a}'_1$ and $\mathbf{a}_2 = \mathbf{a}'_2$.

Answer cross-check: Distributed the questions as follows:

- Player b : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{T}_0, \mathbf{T}_1, \mathbf{U}$; receive $\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2$.
- Player \bar{b} : give $\mathbf{x}_c, \mathbf{T}_c$; receive \mathbf{a}'_c

Accept if $\mathbf{a}_c = \mathbf{a}'_c$.

Answer consistency check: Distributed the questions as follows:

- Player b : give $\mathbf{x}_c, \mathbf{T}_c$; receive \mathbf{a}_c .
- Player \bar{b} : give $\mathbf{x}_c, \mathbf{T}_c$; receive \mathbf{a}'_c

Accept if $\mathbf{a}_c = \mathbf{a}'_c$.

Proof cross-check: Distribute the questions as follows:

- Player b : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{T}_0, \mathbf{T}_1, \mathbf{U}$; receive $\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2$.
- Player \bar{b} : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{U}$; receive \mathbf{a}'_2

Accept if $\mathbf{a}_2 = \mathbf{a}'_2$.

Code checks :

Answer code check: Sample questions $(\mathbf{w}_0, \mathbf{w}_1) \sim G_{\ell_1}(\mathbf{T}_c)$. Distributed the questions as follows:

- Player b : give $\mathbf{x}_c, \mathbf{w}_0$; receive \mathbf{a}_0 .
- Player \bar{b} : give $\mathbf{x}_c, \mathbf{w}_1$; receive \mathbf{a}_1 .

Accept if $G_{\ell_1}(\mathbf{T}_c)$ accepts on \mathbf{a}_0 and \mathbf{a}_1 .

Proof code check: Sample questions $(\mathbf{w}_0, \mathbf{w}_1) \sim G_{\ell_2}(\mathbf{U})$. Distribute the questions as follows:

- Player b : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{w}_0$; receive \mathbf{a}_0 .
- Player \bar{b} : give $(\mathbf{x}_0, \mathbf{x}_1), \mathbf{w}_1$; receive \mathbf{a}_1 .

Accept if $G_{\ell_2}(\mathbf{U})$ accepts on \mathbf{a}_0 and \mathbf{a}_1 .

Proof of Theorem 35.

Completeness. This follows the same proof of the completeness part of [39, Theorem 17.10].

Soundness. The constant K_1 depends on the parameter $\kappa = |\mathbf{I}'_0|$, so we should set κ to be a sufficiently large constant so that $1 - K_1 - K_2\varepsilon^{1/8}$ is greater than the soundness of V . Operationally, the views are augmented by κ uniformly randomly chosen coordinates. The purpose of this is to drive the distance of the Hadamard code up from $1/2$ to $1 - 1/2^\kappa$, which will be needed for Lemma 75.

Suppose input is not in L . Let $(|\psi\rangle, M)$ be a strategy that passes with probability $1 - \varepsilon$. This strategy can pass each **Answer code check** with probability $1 - 10\varepsilon$. Given values c and x_c , write $1 - \varepsilon_{c, x_c}$ for the probability the code check passes conditioned on these values. Then with probability at least $1 - 10\varepsilon^{1/2}$, $\varepsilon_{c, x_c} \leq \varepsilon^{1/2}$. When this occurs, we can apply Proposition 31 to $G_{\ell_1}(\mathbf{T}_c)$ where the distribution of \mathbf{T}_c is determined by c and x_c . Proposition 31 implies that there exists Hilbert spaces $\mathcal{H}_{x_c}, |aux_{x_c}\rangle \in \mathcal{H}_{x_c} \otimes \mathcal{H}_{x_c}$ and projective measurement $\{G_u^{x_c}\}$ on \mathcal{H}^{x_c} such that

$$\mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(M_a^{x_c, T_c} \otimes \mathbb{1}_{\mathcal{H}_{A, x_c}} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes G_{[w|T_c=a]}^{x_c}) |\psi\rangle \otimes |aux_{x_c}\rangle\|^2 \leq O(\sqrt{\varepsilon_{c, x_c}})$$

where we use the fact that k is a constant and $\mathbb{1}_A = \mathbb{1}_{\mathcal{H}_A \otimes \mathcal{H}_{A, x_c}}$ and similar for $\mathbb{1}_B$. When this does not occur, we can still assume such Hilbert spaces and projective measurements so that

$$\mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(M_a^{x_c, T_c} \otimes \mathbb{1}_{\mathcal{H}_{A, x_c}} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes G_{[w|T_c=a]}^{x_c}) |\psi\rangle \otimes |aux_{x_c}\rangle\|^2 \leq O(1).$$

When averaging over c and x_c ,

$$\mathbb{E}_{c, x_c} \mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(M_a^{x_c, T_c} \otimes \mathbb{1}_{A, x_c} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes G_{[w|T_c=a]}^{x_c}) |\psi\rangle \otimes |aux_{x_c}\rangle\|^2 \leq O(\varepsilon^{1/4}).$$

Passing the **Proof code check** implies that there exists Hilbert spaces \mathcal{H}_{x_0, x_1} , states $|aux_{x_0, x_1}\rangle \in \mathcal{H}_{x_0, x_1} \otimes \mathcal{H}_{x_0, x_1}$ and projective measurements $\{H_w^{x_0, x_1}\}$ on $\mathcal{H} \otimes \mathcal{H}_{x_0, x_1}$ such that

$$\mathbb{E}_{x_0, x_1} \mathbb{E}_{U \sim D(x_0, x_1)} \sum_{a \in \mathbb{F}_2^k} \|(M_a^{x_0, x_1, U} \otimes \mathbb{1}_{\mathcal{H}_{A, x_0, x_1}} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes H_{[w|U=a]}^{x_0, x_1}) |\psi\rangle \otimes |aux_{x_0, x_1}\rangle\|^2 \leq O(\varepsilon^{1/4}).$$

The next step is ensuring the G and H measurements act on the same Hilbert space. Let

$$|\tilde{\psi}\rangle = |\psi\rangle \otimes (\otimes_x |aux_x\rangle) \otimes (\otimes_{x_0, x_1} |aux_{x_0, x_1}\rangle)$$

and

$$\begin{aligned} \tilde{G}_u^{x_c} &= G_u^{x_c} \otimes (\otimes_{x \neq x_c} \mathbb{1}_{\mathcal{H}_x}) \otimes (\otimes_{x_0, x_1} \mathbb{1}_{\mathcal{H}_{x_0, x_1}}) \\ \tilde{H}_u^{x_0, x_1} &= H_u^{x_0, x_1} \otimes (\otimes_x \mathbb{1}_{\mathcal{H}_x}) \otimes (\otimes_{(z_0, z_1) \neq (x_0, x_1)} \mathbb{1}_{\mathcal{H}_{z_0, z_1}}), \end{aligned}$$

and, let

$$\begin{aligned} N_{a_c}^{x_c, T_c} &= M_{a_c}^{x_c, T_c} \otimes (\otimes_x \mathbb{1}_{\mathcal{H}_x}) \otimes (\otimes_{x_0, x_1} \mathbb{1}_{\mathcal{H}_{x_0, x_1}}) \\ N_{a_2}^{x_0, x_1, U} &= M_{a_2}^{x_0, x_1, U} \otimes (\otimes_x \mathbb{1}_{\mathcal{H}_x}) \otimes (\otimes_{x_0, x_1} \mathbb{1}_{\mathcal{H}_{x_0, x_1}}) \\ N_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U} &= M_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U} \otimes (\otimes_x \mathbb{1}_{\mathcal{H}_x}) \otimes (\otimes_{x_0, x_1} \mathbb{1}_{\mathcal{H}_{x_0, x_1}}). \end{aligned}$$

Note that we omit the permutation of the Hilbert spaces in the definitions above. Then for all x_c

$$\begin{aligned} &\mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(N_{a_c}^{x_c, T_c} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes \tilde{G}_{[w|T_c=a]}^{x_c}) |\tilde{\psi}\rangle\|^2 \\ &= \mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(M_{a_c}^{x_c, T_c} \otimes \mathbb{1}_{\mathcal{H}_{A, x_c}} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes G_{[w|T_c=a]}^{x_c}) |\psi\rangle \otimes |aux_{x_c}\rangle\|^2. \end{aligned}$$

Thus

$$\mathbb{E}_{c, x_c} \mathbb{E}_{T_c \sim D_{x_c}} \sum_{a \in \mathbb{F}_2^k} \|(N_{a_c}^{x_c, T_c} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes \tilde{G}_{[w|T_c=a]}^{x_c}) |\tilde{\psi}\rangle\|^2 \leq O(\varepsilon^{1/4}), \quad (29)$$

and

$$\mathbb{E}_{x_0, x_1} \mathbb{E}_{U \sim D(x_0, x_1)} \sum_{a \in \mathbb{F}_2^k} \|(N_a^{x_0, x_1, U} \otimes \mathbb{1}_B - \mathbb{1}_A \otimes \tilde{H}_{[w|U=a]}^{x_0, x_1}) |\tilde{\psi}\rangle\|^2 \leq O(\varepsilon^{1/4}). \quad (30)$$

Note these relations also hold with the two systems flipped.

Passing the **Cross Checks** implies that

$$N_{a_0}^{x_0, x_1, T_0, T_1, U} \otimes \mathbb{1}_B \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_0}^{x_0, T_0} \quad (31)$$

$$N_{a_1}^{x_0, x_1, T_0, T_1, U} \otimes \mathbb{1}_B \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_1}^{x_1, T_1} \quad (32)$$

$$N_{a_0}^{x_0, T_0} \otimes \mathbb{1} \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_0}^{x_0, T_0} \quad (33)$$

$$N_{a_1}^{x_1, T_1} \otimes \mathbb{1} \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_1}^{x_1, T_1} \quad (34)$$

$$N_{a_2}^{x_0, x_1, T_0, T_1, U} \otimes \mathbb{1}_B \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_2}^{x_0, x_1, U} \quad (35)$$

$$N_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U} \otimes \mathbb{1}_B \approx_{O(\varepsilon)} \mathbb{1}_A \otimes N_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U}, \quad (36)$$

with respect to $|\tilde{\psi}\rangle$. These equations combined with Equations (29) and (30) imply the measurements $\{N_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U}\}$, $\{\tilde{G}_u^x\}$ and $\{\tilde{H}_w^{x_0, x_1}\}$ satisfy conditions of Lemma 75 with respect to $|\tilde{\psi}\rangle$. Let

$$\{\Lambda_{u_0, u_1, w}^{x_0, x_1} := \tilde{G}_{u_0}^{x_0} \cdot \tilde{G}_{u_1}^{x_1} \cdot \tilde{H}_w^{x_0, x_1} \cdot \tilde{G}_{u_1}^{x_1} \cdot \tilde{G}_{u_0}^{x_0}\}$$

be a POVM constructed following Lemma 75. Recall that T_c and U has κ independent coordinates, so two different codewords agree on T_c or U with a probability at most $\eta_H^\kappa = 1/2^\kappa$. Hence we can applying Lemma 75 to this POVM with $k = 3$, $\delta := \varepsilon^{1/4}$ and $\varepsilon := 1/2^\kappa$, and get that

$$N_{a_0, a_1, a_2}^{x_0, x_1, T_0, T_1, U} \otimes \mathbb{1}_B \approx_{O(\varepsilon^{1/48} + 1/2^{\kappa/6})} \mathbb{1}_A \otimes \Lambda_{[u_0|_{T_0}, u_1|_{T_1}, w|_U = a_0, a_1, a_2]}^{x_0, x_1} \quad (37)$$

with respect to $|\tilde{\psi}\rangle$, where $[u_0|_{T_0}, u_1|_{T_1}, w|_U = a_0, a_1, a_2]$ means that $\text{Enc}_{\ell_1}(u_0)|_{T_0} = a_0$ and etc.. Passing **Verify** with a probability at least $1 - 10\varepsilon$ along with Equation (37) and Lemma 70 implies that $\{\Lambda_{u_0, u_1, w}^{x_0, x_1}\}$ can be used to pass the verify test with probability $1 - 10\varepsilon - O(\varepsilon^{1/96} + (1/2)^\kappa/6)$. The player would measure $\mathbb{1}_A \otimes \Lambda$ on $|\tilde{\psi}\rangle$ and return the local views of the measurement outcomes according to the questions.

Consider the measurements $\{\Lambda_{u_0, u_1}^{x_0, x_1} := \sum_w \Lambda_{u_0, u_1, w}^{x_0, x_1}\}$ Let

$$p := \mathbb{E}_{x_0, x_1} \sum_{u_0, u_1: V(\text{input}, x_0, x_1, u_0, u_1)=1} \langle \tilde{\psi} | \mathbb{1}_A \otimes \Lambda_{u_0, u_1}^{x_0, x_1} | \tilde{\psi} \rangle,$$

which is the probability that measuring with $\Lambda_{u_0, u_1}^{x_0, x_1}$ gives answers u_0 and u_1 accepted by the verifier V when the questions are x_0 and x_1 . Then

$$\begin{aligned} p &= \mathbb{E}_{x_0, x_1} \sum_{u_0, u_1: V(\text{input}, x_0, x_1, u_0, u_1)=1} \sum_w \langle \tilde{\psi} | \mathbb{1}_A \otimes \Lambda_{u_0, u_1, w}^{x_0, x_1} | \tilde{\psi} \rangle \\ &\geq \mathbb{E}_{x_0, x_1} \sum_{u_0, u_1: V(\text{input}, x_0, x_1, u_0, u_1)=1} \sum_w \langle \tilde{\psi} | \mathbb{1}_A \otimes \Lambda_{u_0, u_1, w}^{x_0, x_1} | \tilde{\psi} \rangle \cdot \Pr_R[V_{\text{PCPP}}^{u_0, u_1, w}(\text{input}, x_0, x_1, 2 \cdot 2^{\ell_1}; R) = 1] \\ &= \Pr[(|\tilde{\psi}\rangle, \Lambda) \text{ pass } \mathbf{verify} \text{ check}] \\ &\quad - \sum_{u_0, u_1: V(\text{input}, x_0, x_1, u_0, u_1)=0} \sum_w \langle \tilde{\psi} | \mathbb{1}_A \otimes \Lambda_{u_0, u_1, w}^{x_0, x_1} | \tilde{\psi} \rangle \cdot \Pr_R[V_{\text{PCPP}}^{u_0, u_1, w}(\text{input}, x_0, x_1, 2 \cdot 2^{\ell_1}; R) = 1] \\ &\geq 1 - 10\varepsilon - O(\varepsilon^{1/96} + (1/2)^\kappa/6) \\ &\quad - \sum_{u_0, u_1: V(\text{input}, x_0, x_1, u_0, u_1)=0} \sum_w \langle \tilde{\psi} | \mathbb{1}_A \otimes \Lambda_{u_0, u_1, w}^{x_0, x_1} | \tilde{\psi} \rangle \cdot \Pr_R[V_{\text{PCPP}}^{u_0, u_1, w}(\text{input}, x_0, x_1, 2 \cdot 2^{\ell_1}; R) = 1] \\ &\geq 1 - 10\varepsilon - O(\varepsilon^{1/8} + 1/2^\kappa) - (1 - p)s, \end{aligned}$$

30:70 The Computational Advantage of MIP* Vanishes in the Presence of Noise

where s is the soundness of V_{PCPP} . In the derivation above, $\Pr_R[V_{\text{PCPP}}^{u_0, u_1, w}(\text{input}, x_0, x_1, 2 \cdot 2^{\ell_1}; R) = 1]$ is the probability that V_{PCPP} accepts input. For any x_0, x_1, u_0, u_1 not accepted by V , this probability is below s by [39, Proposition 17.8]. Hence

$$p \geq \frac{1 - 10\varepsilon - O(\varepsilon^{1/96} + (1/2)^{\kappa/6}) - s}{1 - s} = 1 - \frac{10\varepsilon + O(\varepsilon^{1/96} + (1/2)^{\kappa/6})}{1 - s}.$$

In the end, we use $(\{\tilde{G}_u^{x_0}\}, |\tilde{\psi}\rangle)$ as a strategy for V . Applying Lemma 73 to Equations (29), (33), and (34), we get that

$$\tilde{G}_{u|T_0=a}^{x_0} \otimes \mathbf{1} \approx_{O(\varepsilon^{1/4})} \mathbf{1} \otimes \tilde{G}_{u|T_0=a}^{x_0}$$

with respect to the distribution of x_0 and the distribution of T_0 determined by x_0 on the state $|\tilde{\psi}\rangle$. Since $\{\tilde{G}_u^{x_0}\}$ is a projective measurement, we know

$$\mathbb{E}_{x_0} \mathbb{E}_{T_0 \sim D_{x_0}} \sum_a \langle \tilde{\psi} | \tilde{G}_{u|T_0=a}^{x_0} \otimes \tilde{G}_{u|T_0=a}^{x_0} | \tilde{\psi} \rangle \geq 1 - O(\varepsilon^{1/4}).$$

On the other hand

$$\begin{aligned} & \mathbb{E}_{x_0} \mathbb{E}_{T_0 \sim D_{x_0}} \sum_a \langle \tilde{\psi} | \tilde{G}_{u|T_0=a}^{x_0} \otimes \tilde{G}_{u|T_0=a}^{x_0} | \tilde{\psi} \rangle \\ &= \mathbb{E}_{x_0} \sum_u \langle \tilde{\psi} | \tilde{G}_u^{x_0} \otimes \tilde{G}_u^{x_0} | \tilde{\psi} \rangle + \mathbb{E}_{x_0} \mathbb{E}_{T_0 \sim D_{x_0}} \sum_{u \neq u': u|_{T_0} = u'|_{T_0}} \langle \tilde{\psi} | \tilde{G}_u^{x_0} \otimes \tilde{G}_{u'}^{x_0} | \tilde{\psi} \rangle \\ &= \mathbb{E}_{x_0} \sum_u \langle \tilde{\psi} | \tilde{G}_u^{x_0} \otimes \tilde{G}_u^{x_0} | \tilde{\psi} \rangle + \mathbb{E}_{x_0} \mathbb{E}_{T_0 \sim D_{x_0}} \sum_{u \neq u'} \mathbf{1}[u|_{T_0} = u'|_{T_0}] \langle \tilde{\psi} | \tilde{G}_u^{x_0} \otimes \tilde{G}_{u'}^{x_0} | \tilde{\psi} \rangle. \end{aligned}$$

Since for all x_0 and $u \neq u'$, $\mathbb{E}_{T_0 \sim D_{x_0}} \mathbf{1}[u|_{T_0} = u'|_{T_0}] \leq 1/2^\kappa$, we know

$$\mathbb{E}_{x_0} \sum_u \langle \tilde{\psi} | \tilde{G}_u^{x_0} \otimes \tilde{G}_u^{x_0} | \tilde{\psi} \rangle \geq 1 - 1/2^\kappa - O(\varepsilon^{1/4}).$$

Again, because $\{\tilde{G}_u^{x_0}\}$ is a projective measurement

$$\mathbb{E}_{x_0} \sum_u \|(\tilde{G}_u^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_u^{x_0}) | \tilde{\psi} \rangle\|^2 \leq \frac{1}{2^{\kappa-1}} + O(\varepsilon^{1/4}).$$

Let $S(x_0, x_1) = \{(a_0, a_1) \mid V(x_0, x_1, a_0, a_1) = 1\}$. We can calculate

$$\begin{aligned} & \left| \mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | \tilde{G}_{a_0}^{x_0} \otimes \tilde{G}_{a_1}^{x_1} | \tilde{\psi} \rangle - \langle \tilde{\psi} | \tilde{G}_{a_0}^{x_0} \otimes \tilde{G}_{a_1}^{x_1} \tilde{G}_{a_0}^{x_0} | \tilde{\psi} \rangle \right| \\ & \leq \sqrt{\mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \|\tilde{G}_{a_0}^{x_0} \otimes \tilde{G}_{a_1}^{x_1} | \tilde{\psi} \rangle\|^2} \\ & \quad \cdot \sqrt{\mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | (\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0})(\mathbf{1} \otimes \tilde{G}_{a_1}^{x_1})(\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0}) | \tilde{\psi} \rangle} \\ & \leq 1 \cdot \sqrt{\mathbb{E}_{x_0} \sum_{a_0} \|(\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0}) | \tilde{\psi} \rangle\|^2} \\ & \leq O\left(\frac{1}{2^{\kappa/2}} + \varepsilon^{1/8}\right), \end{aligned}$$

and

$$\begin{aligned}
& \left| \mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0} \tilde{G}_{a_1}^{x_1} \tilde{G}_{a_0}^{x_0} | \tilde{\psi} \rangle - \langle \tilde{\psi} | \tilde{G}_{a_0}^{x_0} \otimes \tilde{G}_{a_1}^{x_1} \tilde{G}_{a_0}^{x_0} | \tilde{\psi} \rangle \right| \\
& \leq \sqrt{\mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \|\mathbf{1} \otimes \tilde{G}_{a_1}^{x_1} \tilde{G}_{a_0}^{x_0} | \tilde{\psi} \rangle\|^2} \\
& \quad \cdot \sqrt{\mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | (\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0})(\mathbf{1} \otimes \tilde{G}_{a_1}^{x_1})(\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0}) | \tilde{\psi} \rangle} \\
& \leq 1 \cdot \sqrt{\mathbb{E}_{x_0} \sum_{a_0} \|(\tilde{G}_{a_0}^{x_0} \otimes \mathbf{1} - \mathbf{1} \otimes \tilde{G}_{a_0}^{x_0}) | \tilde{\psi} \rangle\|^2} \\
& \leq O\left(\frac{1}{2^{\kappa/2}} + \varepsilon^{1/8}\right).
\end{aligned}$$

Note that $\tilde{G}_{a_0}^{x_0} \tilde{G}_{a_1}^{x_1} \tilde{G}_{a_0}^{x_0} = \Lambda_{a_0, a_1}^{x_0, x_1}$. Therefore,

$$\left| \mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | (\tilde{G}_{a_0}^{x_0} \otimes \tilde{G}_{a_1}^{x_1} - \mathbf{1} \otimes \Lambda_{a_0, a_1}^{x_0, x_1}) | \tilde{\psi} \rangle \right| \leq O\left(\frac{1}{2^{\kappa/2}} + \varepsilon^{1/8}\right).$$

On the other hand, we have shown

$$\mathbb{E}_{x_0, x_1} \sum_{(a_0, a_1) \in S} \langle \tilde{\psi} | \mathbf{1} \otimes \Lambda_{a_0, a_1}^{x_0, x_1} | \tilde{\psi} \rangle = p \geq 1 - O(\varepsilon^{1/96} + (1/2)^{\kappa/6}).$$

Hence, the winning probability of the strategy $(\{\tilde{G}_u^x\}, |\tilde{\psi}\rangle)$ is at least $1 - \frac{C_1}{2^{\kappa/6}} - C_2 \varepsilon^{1/96}$ for some constants C_1 and C_2 . Hence, $K_1 = \frac{C_1}{2^{\kappa/6}}$ and $K_2 = C_2$ in the soundness statement. ◀

Proof of Theorem 36. We first oracularize the MIP* protocol for the Halting problem from [28]. Denote the oracularized verifier by V . For inputs of size n , the verifier's running time for sampling questions and checking answers is $O(\text{poly}(n))$. The sizes of the questions and answers are also $O(\text{poly}(n))$. The oracularized protocol maintains completeness 1 and a constant soundness.

Define the language L_{Enc} as in Definition 33 for V . Then $L_{\text{Enc}} \in \text{DTIME}(2^{\text{poly}(n)})$ because the most costly step of the decider of L_{Enc} is running $\text{Dec}_{\text{poly}(n)}$ which takes $O(2^{\text{poly}(n)})$ time. By Definition 32, the PCPP verifier V_{PCPP} for L_{Enc} has randomness complexity $O(\text{poly}(n))$, query complexity $O(1)$, and verification time $O(\text{poly}(n))$.

Next, we apply the answer reduction technique of this section to V to get verifier V^{AR} . The sampling time of V^{AR} is the sum of the sampling time of V , the sampling time of V_{PCPP} , and the sampling time of the additional constantly many independent coordinates, so it is $O(\text{poly}(n))$. Since the question sizes of V and V_{PCPP} are both $O(\text{poly}(n))$, the question size of V^{AR} is also $O(\text{poly}(n))$. The answers expected by V^{AR} are constantly many bits, so the answer size is $O(1)$. Lastly, the verification time of V^{AR} is determined by the verification time of V_{PCPP} , so it is also $O(\text{poly}(n))$. The completeness and soundness of V^{AR} follow from Theorem 35. Then the theorem statement follows from the Halting problem is RE-complete. ◀

Low-Depth Algebraic Circuit Lower Bounds over Any Field

Michael A. Forbes 

Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA

Abstract

The recent breakthrough of Limaye, Srinivasan and Tavenas [15] (LST) gave the first super-polynomial lower bounds against low-depth algebraic circuits, for any field of zero (or sufficiently large) characteristic. It was an open question to extend this result to small-characteristic ([8, 9, 16]), which in particular is relevant for an approach to prove superpolynomial $AC^0[p]$ -Frege lower bounds ([9]).

In this work, we prove super-polynomial algebraic circuit lower bounds against low-depth algebraic circuits over any field, with the same parameters as LST (or even matching the improved parameters of Bhargav, Dutta, and Saxena [3]). We give two proofs. The first is *logical*, showing that even though the proof of LST naively fails in small characteristic, the proof is sufficiently algebraic that generic transfer results imply the result over characteristic zero implies the result over all fields. Motivated by this indirect proof, we then proceed to give a second *constructive* proof, replacing the field-dependent set-multilinearization result of LST with a set-multilinearization that works over any field, by using the Binet-Minc identity [17].

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases algebraic circuits, lower bounds, low-depth circuits, positive characteristic

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.31

Funding *Michael A. Forbes*: Supported by NSF CAREER award 2047310.

1 Introduction

Algebraic complexity asks for the best methods to compute multivariate polynomials $f(x_1, \dots, x_n)$ from the primitive operations of addition and multiplication, using constants from the underlying field \mathbb{F} for free. These computations are organized as an *algebraic circuit*, a directed acyclic graph of these primitive operations from the input variables x_1, \dots, x_n . The complexity of this object is primarily governed by its *size*, which is the number of operations in the circuit. Another complexity measure is the *product depth*, which is the maximum number of product operations on any input to output path. In a recent breakthrough, Limaye, Srinivasan and Tavenas [15] (LST) gave the first superpolynomial lower bounds for explicit polynomials to be computed by low-product-depth algebraic circuits, in *large characteristic*.¹

The main motivation for our work was to better understand the breakthrough work of Limaye, Srinivasan and Tavenas [15] (LST), as it left several open problems. In particular, Limaye, Srinivasan and Tavenas noted in their survey [16] the challenge of obtaining their lower bound over arbitrary fields; a problem this paper resolves. We briefly review the motivation for this challenge.

Algebraic Circuit Lower Bounds

Obtaining circuit lower bounds over *any* field is a fundamental aspect of algebraic complexity theory, as both the large and small characteristic settings are fundamentally interesting. These two regimes are somewhat incomparable in their difficulty. However there is the general

¹ For ease of exposition, we will write “large characteristic” to mean “large (or zero) characteristic”.

intuition that lower bounds in small characteristic “should be” easier, as there are “fewer” efficient algorithms to rule out. Following this intuition, a small characteristic analogue of LST should be achievable, as done in this paper.

To justify this intuition, here are examples of efficient algebraic algorithms known to exist in large characteristic which currently lack analogues in small characteristic. First, consider the Newton identities, which relate the elementary symmetric polynomials to the power-sum polynomials, but only in large characteristic. These identities are crucial in various algebraic complexity settings. For example, for constructing non-trivially small depth-4 formulas for the elementary symmetric polynomials ([20]) (as used by Limaye, Srinivasan and Tavenas [15]). Another example is a folklore construction of a polylogarithmic-depth polynomial-size algebraic circuit for the determinant (essentially the Faddeev–LeVerrier algorithm, expressed as an algebraic circuit), which uses traces of matrix powers to compute the power-sums of eigenvalues, and then computes the product of the eigenvalues (the determinant) from these power-sums using a small formula constructed from the Newton identities.

Another example is Fischer’s identity [5], which shows how to compute the monomial $x_1 \dots x_n$ as a homogeneous sum of powers of linear forms, but only in large characteristic. This identity was crucial to the celebrated depth-reduction of algebraic circuits to depth-3 ([13]).

A more computational example is the fundamental result in algebraic complexity that small algebraic circuits can be factored efficiently ([14]); however in characteristic p the result only produces a factorization up to p -th powers. The current inability (despite recent progress of Andrews [2]) to take p -th roots of algebraic circuits in characteristic p in particular leads to weaker hardness versus randomness trade-offs.

On the other hand, there *are* efficient algorithms that arise only in small characteristic and as such the above intuition is not quite correct. In particular, in characteristic p we have the identity $(x + y)^p = x^p + y^p$. In characteristic 2, we can compute the permanent efficiently (as $\det = \text{perm}$ over such fields) and also have a tighter connection between boolean and algebraic circuits. One can also view $n \times n$ Hadamard matrices as examples of this phenomenon, as in large characteristic they are full rank (and as such, “hard”), but in characteristic two then can have rank $O(\log n)$ (and as such, “easy”). Additionally, lifted Reed-Solomon codes ([12]) and William’s algorithm for k -path ([22]) are other techniques that only work in small characteristic.

Proof Complexity

Aside from interest in algebraic circuit lower bounds in small characteristic for their own sake, there are significant applications of such lower bounds.

A long-standing open question in proof-complexity is to prove superpolynomial lower bounds over constant-depth reasoning using modular gates, in particular the $\text{AC}^0[p]$ -Frege system. This challenge is notable as the boolean circuit complexity version has been solved by Razborov [19] and Smolensky [21], but the techniques have thus far not been successfully exported into the proof-complexity setting. In the context of this challenge, Grochow and Pitassi [11] showed that their Ideal Proof System (IPS) can efficiently simulate $\text{AC}^0[p]$ -Frege when IPS is over a field of characteristic p , and hence in particular that superpolynomial lower bounds for constant-depth IPS refutations of CNFs in characteristic p would give superpolynomial lower bounds against $\text{AC}^0[p]$ -Frege.

Toward this goal, Govindasamy, Hakoniemi, and Tzameret [9] showed how the lower bound of Limaye, Srinivasan and Tavenas [15] can yield a superpolynomial lower bound against (multilinear) constant-depth IPS refutations for (a variant of) the subset-sum problem, in large characteristic. While the subset-sum problem they use is *easy* in small characteristic

and hence our lower bounds cannot as is improve their results, our results eliminate one barrier to progress toward constant-depth IPS lower bounds in characteristic p , and hence $AC^0[p]$ -Frege lower bounds.

Polynomial Identity Testing

For applications within algebraic complexity itself, Limaye, Srinivasan and Tavenas [15] showed how their lower bound yields a deterministic subexponential time polynomial identity testing (PIT) algorithm for constant-depth algebraic circuits, in large characteristic. The restriction on characteristic comes from two places, the lower bound itself, as well as the known relations between algebraic hardness and derandomization. By removing one of these restrictions, our result hence makes progress toward obtaining corresponding PIT algorithms in small characteristic.

2 Our results

The lower bound of LST has two components. The first is a new set-multilinearization result for algebraic circuits, showing that algebraic circuits can be non-trivially set-multilinearized in a particular regime of parameters. The second component is a new lower bound for set-multilinear formulas. While the second component holds over any field, the first requires large characteristic. As such, our work focuses on this first component, which we now discuss more in depth.

Recall that for a partition of variables $\bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$, a *set-multilinear* monomial is one of the form $x_{1,i_1} \cdots x_{d,i_d}$. A set-multilinear polynomial is a linear combination of set-multilinear monomials. Many important polynomials are set-multilinear with respect to natural partitions of the variables. For example, the permanent of a matrix is set-multilinear with respect to the partition of the matrix into rows. For algebraic circuits computing set-multilinear polynomials it is natural to ask that the circuit respects the set-multilinear structure. In particular, we say that an algebraic circuit is (*syntactically*) *set-multilinear* if all product gates $f = f_1 \cdots f_k$ in the circuit have that the f_i are on disjoint parts of the variable partition. It follows then that set-multilinear circuits can only compute set-multilinear polynomials.

The first step of LST is a set-multilinearization result which transforms a low-depth algebraic circuit on n variables computing a degree d set-multilinear polynomial to one computing the same polynomial, where now the computation is itself set-multilinear. One pays for imposing this structure by increasing the circuit size. Crucially, the size only increases by a function of the degree d , *not* in the number of variables n .

► **Theorem 1** ([15]). *Let \mathbb{F} be a field of characteristic $\text{char}(\mathbb{F}) > d$ (or zero), where d is a parameter. Let f a set-multilinear polynomial of degree d , computed by a product-depth Δ circuit of size s . Then f is computed by a $(d^d \cdot s)^{O(1)}$ -size set-multilinear circuit of product-depth 2Δ .*

This result is proven in two steps. The first is to use an efficient low-depth homogenization transformation, that will double the product depth and increase the circuit size by $2^{O(\sqrt{d})}$. This construction uses that the characteristic is large, and is a generalization of the following result.

► **Theorem 2** ([20]). *Let \mathbb{F} be a field of characteristic $> d$, or zero. Then the elementary symmetric polynomial $\text{esym}_{n,d} = \sum_{S \in \binom{[n]}{d}} \prod_{i \in S} x_i$ has a homogeneous depth-4 sums of products of sums of powers $(\sum \prod \sum \wedge)$ formula of size $\text{poly}(n, 2^{\sqrt{d}})$.*

The standard connection between elementary symmetric polynomials and computing homogeneous parts allows the above theorem to homogenize depth-3 circuits into homogeneous depth-5 circuits with a $2^{O(\sqrt{d})}$ -blowup in circuit size. LST generalized Theorem 2 to *weighted* elementary symmetric polynomials to allow the idea to succeed in higher depths.

After converting product-depth Δ to homogeneous product-depth 2Δ , LST then convert the circuit to be set-multilinear while preserving the product-depth. This conversion will work over any field, and is a simple gate-simulation proof.

► **Theorem 3** ([15]). *Let \mathbb{F} be any field. Let f be a set-multilinear polynomial of degree d , computed by a homogeneous product-depth Δ circuit of size s . Then f is computed by a $\text{poly}(d^d, s)$ -size set-multilinear circuit of product-depth Δ .*

Combining the two steps of homogenization, then set-multilinearization, LST obtained the following.

► **Theorem 4** ([15]). *Let \mathbb{F} be a field of characteristic $\text{char}(\mathbb{F}) > d$ (or zero). Let f be a set-multilinear polynomial computed by a product-depth Δ circuit of size s . Then f can be computed by a $(d^d \cdot s)^{O(1)}$ -size set-multilinear circuit of product-depth 2Δ .*

The most natural method to obtaining the result of LST in arbitrary fields would be to give an efficient homogenization for low-depth circuits over all fields, and indeed this was posed as an open question in [8, 9, 16]. However, a barrier to this approach is that it is still open to develop an analogue of Theorem 2 in small characteristic fields.

Additionally, a recent work of Fournier, Limaye, Srinivasan, Tavenas [8] formalized this barrier, by showing that in small characteristic that a certain form of Newton identities cannot hold. This shows that while in large characteristic the Newton identities imply efficient low-depth homogenization for low-degree polynomials, there are provable barriers for obtaining an analogous result in small characteristic.

2.1 Lower bounds over any field, without explicit set-multilinearization

Our first result is to show that while we do not overcome this barrier, we never the less obtain the lower bound of LST over any field, which we state here incorporating the improved parameters from Bhargav, Dutta, and Saxena [3].

► **Theorem** (Main Theorem, Theorem 10). *Let \mathbb{F} be any field, and $d = o(\log n)$. Then the iterated matrix multiplication polynomial $\text{IMM}_{n,d} = (X_1 \cdots X_d)_{1,1}$ where X_i is an $n \times n$ symbolic matrix, requires*

$$n^{\Theta\left(d^{\frac{1}{F_{2\Delta+2} - 1}} / \Delta\right)}$$

size algebraic circuits of product depth Δ , where F_k is the k -th Fibonacci number (so $F_0 = 0$, $F_1 = F_2 = 1$, $F_3 = 2$, etc.).²

We observe that one can bypass the above barrier, and indeed the entire need for efficient low-depth homogenization, by arguing that the original techniques of LST already suffice for obtaining their result in low-characteristic, due to considerations from mathematical logic. That is, we study the *proof* of LST, and argue that the proof is sufficiently algebraic so that generic algebraic arguments imply the result holds over arbitrary fields.

² Bhargav, Dutta, and Saxena [3] use a slightly different indexing of Fibonacci numbers.

We abstract the methods of LST as follows, which is loosely inspired by Geometric Complexity Theory, as well as the theory of algebraic natural proofs ([7, 10]). Given a polynomial f of degree d in n variables, we can view the polynomial as a list of $N = \binom{n+d}{d}$ coefficients which we call the *coefficient vector* $\overline{\text{coeff}}(f)$. We then seek to construct a polynomial Q (or in fact a collection of polynomials Q_1, \dots, Q_M) in N variables such that $Q(\overline{\text{coeff}}(f)) = 0$ whenever f has a small low-depth algebraic circuit. At the same time, we want $Q(\overline{\text{coeff}}(P)) \neq 0$ (or in fact, $Q_i(\overline{\text{coeff}}(P)) \neq 0$ for some i) for some polynomial P . One can then conclude that P does not have a small low-depth algebraic circuit.

One can instantiate LST in this framework as follows. One can carefully rewrite the coefficients of polynomial f into a matrix C_f , and argue that the matrix C_f has low-rank. As matrix rank is characterized by the vanishing of determinant of submatrices, we can take the polynomials Q_i to be the relevant determinants. Any polynomial P whose associated matrix C_P has high-rank will have a non-vanishing determinant, and hence $Q_i(\overline{\text{coeff}}(P)) \neq 0$ for some i .

Many lower bounds techniques in algebraic complexity theory fall in the above “rank based” framework, and often in such proofs one proves that the matrix is high rank by arguing that there exists a large triangular submatrix whose diagonal entries are all 1. In such cases, the determinant of this submatrix is in fact 1, so the matrix is high-rank over *every* field. LST follows this approach, and as such this part of the framework does not depend on the characteristic.

Instead, the dependence on the characteristic comes into the argument that the rank of the matrices C_f is small. To show the rank is small over any field, we first note that the relevant determinants are in fact polynomials with *integer* coefficients, regardless of the actual field of consideration. Second, we note that small low-depth algebraic circuits have a *universal circuit* $U(\bar{x}, \bar{y})$ ([18]),³ such that $f = U(\bar{x}, \bar{\beta})$ for some constants $\bar{\beta}$ and $U(\bar{x}, \bar{y})$ has a small low-depth circuit. Further, U has *integer* coefficients. We then argue that viewing the field of computation as the characteristic zero field $\mathbb{Q}(\bar{y})$ (rational functions in the indeterminates \bar{y}), the matrix C_U must have low-rank, as the argument of LST applies. However, C_U having low-rank over $\mathbb{Q}(\bar{y})$ implies certain determinants of *integer* polynomials vanish, and thus also modulo p for any prime p . Hence C_U has low-rank over *any* field, and this will remain true even when we substitute $\bar{y} \leftarrow \bar{\beta}$. In particular, C_f has low rank as desired.

The overall idea is the standard fact from mathematical logic that if you want to prove a polynomial identity $A(\bar{x}) = 0$ where A has integer coefficients, then proving this identity in characteristic zero implies the result over every field because zero over the integers is zero modulo every prime p . A well-known example of this is the Cayley-Hamilton theorem, which states that every $n \times n$ matrix A is a root of its characteristic polynomial $p_A(x) = \det(xI - A)$. Viewing the entries of A as symbolic this can be viewed as a polynomial identity with integer coefficients, so suffices to prove it in characteristic zero, even if you use techniques specific to characteristic zero. The Cayley-Hamilton theorem can be proven in characteristic zero in two steps. First, one argues the theorem is true for all diagonalizable matrices, which is simple. Second, one argues that diagonalizable matrices are topologically dense in \mathbb{C} (say, in the Euclidean topology, a notion highly tied to characteristic zero) amongst all matrices, and hence by continuity the identity must also vanish on all matrices as desired. As such, one proves the Cayley-Hamilton theorem over all fields using techniques highly specific to characteristic zero.⁴

³ In the actual proof we do not need the universal circuit machinery and appeal to simpler arguments.

⁴ One can avoid the use of characteristic zero here using better algebraic techniques, such as Jordan normal form.

2.2 Lower bounds over any field, via explicit set-multilinearization

The results from the previous section showed that we were able to obtain the result of LST over any field *without* actually efficiently converting algebraic circuits to be set-multilinear. This is very suggestive that such a transformation should be achievable. While the barrier from above regarding efficient small-characteristic homogenization still is present, we observe that we can bypass this barrier (again) by *combining* the homogenization and set-multilinearization steps used by LST into a single transformation.

► **Theorem (Corollary 27).** *Let \mathbb{F} be any field. Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. Let f be a set-multilinear polynomial computed by a product-depth Δ circuit of size s . Then f can be computed by a $(d^d \cdot s)^{O(1)}$ -size set-multilinear circuit of product-depth 2Δ .*

This result is proven by a standard gate simulation argument. The new component is to replace the use of the Newton identities by LST (which only work in large characteristic), by the Binet-Minc identity [17], which is a non-trivial depth-4 set-multilinear identity for computing (rectangular) permanents over any field. This is a natural step, as the Binet-Minc identity can, in large characteristic, be used (see Corollary 19) to recover the efficient depth-4 homogeneous formula for the elementary symmetric polynomial (Theorem 2) whose lack of small characteristic analogue is a barrier for LST holding in small characteristic.

Replacing the set-multilinearization of LST with the above result allows the rest of the proof of LST (and the improvement of Bhargav, Dutta, and Saxena [3]) to work over any field, giving another proof of our main result from above.

We note that that Binet-Minc identity has also recently been used in algebraic complexity by Curticapean, Limaye and Srinivasan [4] for unrelated reasons.

The above Corollary 27 is a more *constructive* method for proving the LST result in small characteristic, while the logical method is much more indirect. However we present the logical approach because it was the first proof we discovered, which motivated the constructive proof, and also the logical approach may have applications in other situations.

2.3 Related Work

An intriguing aspect of the work of LST is that it does not use the somewhat recent notion of shifted partial derivatives, which has powered numerous advances in algebraic circuit lower bounds in the past decade. Motivated by this, Amireddy, Garg, Kayal, Saha, Thankey [1] were able to essentially re-establish LST using shifted partial derivatives. They gave a novel analysis of shifted partials using several “imbalance” ideas related to LST, but crucially their analysis avoided the need to discuss set-multilinear polynomials and as such is perhaps more flexible than the set-multilinear methods of LST.

One of their main lower bounds was that any homogeneous product-depth Δ formula computing $\text{IMM}_{n,d}$ requires size $\geq n^{\Omega(d^{2^{1-\Delta}}/\Delta)}$, when $d \leq O(\lg n)$, over any field. Recall that LST showed, over fields of large characteristic, that general circuits computing degree d polynomials can be homogenized with a doubling in product-depth, and a $2^{\Omega(\sqrt{d})}$ blow-up in circuit size. Invoking this transformation, it follows that the results of Amireddy, Garg, Kayal, Saha, Thankey [1] establish super-polynomial lower bounds for general low-depth circuits, over fields of large characteristic. Quantitatively, the resulting bounds are in between those of LST and those of Bhargav, Dutta, and Saxena [3].

While this paper does not provide a low-depth homogenization transformation akin to that of LST over arbitrary fields, the logical methods of section 3 straightforwardly extend to the setting of Amireddy, Garg, Kayal, Saha, Thankey [1]. That is, the rank-based lower bound they establish for $\text{IMM}_{n,d}$ holds over all fields. The rank-based upper bound shows that all

low-depth general circuits are simple in characteristic zero, using LST's homogenization and their shifted partial analysis. One then can, as in section 3, generically transfer these two rank bounds to arbitrary fields, hence obtaining the lower bound for computing $\text{IMM}_{n,d}$ via low-depth circuits over an arbitrary field. We omit the straightforward details, in particular because the resulting parameters are worse than what are obtained in this work because we invoke the improved parameters from the set-multilinear lower bounds of Bhargava, Dutta, and Saxena [3].

3 Lower bounds over any field, via mathematical logic

In this section, we prove that the LST lower bound holds over any field, using techniques from mathematical logic that transfer the result from large characteristic to all characteristic. The following is our key lemma that transfers algebraic statements between different fields, in particular showing that an integer matrix having low rank in characteristic zero implies it has low-rank over every field. For our actual needs, we will need to consider matrices with entries that are polynomials with integer coefficients.

► **Lemma 5.** *Let $M \in \mathbb{Z}[\bar{x}]^{m \times n}$ be a matrix with integer polynomial entries. Let \mathbb{F} be any field, and interpret $M \in \mathbb{F}[\bar{x}]^{m \times n}$ via the unique non-trivial ring homomorphism $\varphi : \mathbb{Z} \rightarrow \mathbb{F}$. Then for any $\bar{\alpha} \in \mathbb{F}^{\bar{x}}$,*

$$\text{rank}_{\mathbb{F}} M(\bar{\alpha}) \leq \text{rank}_{\mathbb{F}(\bar{x})} M(\bar{x}) \leq \text{rank}_{\mathbb{Q}(\bar{x})} M(\bar{x}).$$

Proof.

1) $\text{rank}_{\mathbb{F}(\bar{x})} M(\bar{x}) \leq \text{rank}_{\mathbb{Q}(\bar{x})} M(\bar{x})$: In both cases the matrix M is the same, we simply change the field of interpretation. Recall that a matrix is $\text{rank} \leq s$ iff all $(s+1) \times (s+1)$ submatrices have zero determinant. Let $r = \text{rank}_{\mathbb{Q}(\bar{x})} M(\bar{x})$ so that all $(r+1) \times (r+1)$ submatrices of $M[\bar{x}]$ have zero determinant in $\mathbb{Q}(\bar{x})$. As M has polynomial entries, and the determinant is a polynomial, it follows that under the homomorphism φ that these submatrices still have zero determinant. As such, all $(s+1) \times (s+1)$ submatrices of M have zero determinant when viewed as a matrix in $\mathbb{F}[\bar{x}]^{m \times n}$, so that $\text{rank}_{\mathbb{F}(\bar{x})} M(\bar{x}) \leq r$.

2) $\text{rank}_{\mathbb{F}} M(\bar{\alpha}) \leq \text{rank}_{\mathbb{F}(\bar{x})} M(\bar{x})$: Let $t = \text{rank}_{\mathbb{F}(\bar{x})} M(\bar{x})$, so that all $(t+1) \times (t+1)$ submatrices of M have determinant zero in $\mathbb{F}[\bar{x}]$. Define the ring homomorphism $\psi : \mathbb{F}[\bar{x}] \rightarrow \mathbb{F}$ by $\bar{x} \rightarrow \bar{\alpha}$. As above, it then follows that all $(t+1) \times (t+1)$ submatrices of M have zero determinant under this homomorphism. But as $\psi(M(\bar{x})) = M(\bar{\alpha})$, it then follows that all such submatrices of $M(\bar{\alpha})$ have zero determinant in \mathbb{F} , so $\text{rank}_{\mathbb{F}} M(\bar{\alpha}) \leq t$. ◀

► **Definition 6.** *Let \mathbb{F} a field. Let \bar{x} be a set of variables with a partition $\bar{x} = \bar{y}_1 \sqcup \cdots \sqcup \bar{y}_{d_y} \sqcup \bar{z}_1 \sqcup \cdots \sqcup \bar{z}_{d_z}$, where $d = d_y + d_z$. Let Y denote the set of all set-multilinear monomials with respect to the partition $\bar{y} = \bar{y}_1 \sqcup \cdots \sqcup \bar{y}_{d_y}$, and let Z denote the set of all set-multilinear monomials with respect to the partition $\bar{z} = \bar{z}_1 \sqcup \cdots \sqcup \bar{z}_{d_z}$.*

*Given a polynomial $f(\bar{x})$ which is set-multilinear with respect to the above partition, define the **coefficient matrix** $C_f \in \mathbb{F}^{Y \times Z}$ by*

$$(C_f)_{\bar{y}^{\bar{b}}, \bar{z}^{\bar{c}}} = \text{Coeff}_{\bar{y}^{\bar{b}} \bar{z}^{\bar{c}}}(f),$$

where $\bar{y}^{\bar{b}}, \bar{z}^{\bar{c}}$ are set-multilinear monomials, and Coeff takes the coefficient of $\bar{y}^{\bar{b}} \bar{z}^{\bar{c}}$ in f .

The **relative rank** of f is then defined as

$$\text{relrank}_{\mathbb{F}}(f) := \frac{\text{rank}_{\mathbb{F}}(C_f)}{\sqrt{|Y| \cdot |Z|}}.$$

where $\text{rank}_{\mathbb{F}}(C_f)$ is the matrix rank of C_f over the field \mathbb{F} .

The following set-multilinearization result is a slight extension of what is proven in LST, and follows from their methods as noted by [9].

► **Definition 7.** Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. A monomial is *set-multilinear (with respect to the partition $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$)* if it can be written as $\prod_{i=1}^d (\bar{x}_i)_{j_i}$ for some j_1, \dots, j_d .

Define the *set-multilinear projection (with respect to the partition $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$)* to be the linear map $\pi_{\text{sm}} : \mathbb{F}[\bar{x}] \rightarrow \mathbb{F}[\bar{x}]$ which is identity on set-multilinear monomials, and zero on all other monomials.

► **Theorem 8 ([9, 15]).** Let \mathbb{F} be a field of characteristic $\text{char}(\mathbb{F}) > d$ (or zero), where d is a parameter. Let f be a product-depth Δ circuit of size s . Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. Then the set-multilinear projection $\pi_{\text{sm}}(f)$ has a $(d^d \cdot s)^{O(1)}$ -size set-multilinear circuit of product-depth 2Δ .

We quote here the summary of the LST lower bound results that we need, incorporating the improved parameters from Bhargav, Dutta, and Saxena [3].

► **Theorem 9 ([3, 15]).** Let \mathbb{F} a field. Let \bar{x} be a set of n variables, and $d \leq o(\log n)$ a parameter. Then there exists a partition $\bar{x} = \bar{y}_1 \sqcup \cdots \sqcup \bar{y}_{d_y} \sqcup \bar{z}_1 \sqcup \cdots \sqcup \bar{z}_{d_z}$, only depending on n and d , where $d = d_y + d_z$, such that any $f(\bar{x})$ computed by a set-multilinear circuit of size s and product-depth Δ has relative rank bounded by

$$\text{relrank}(f) \leq s \cdot n^{-\Theta\left(d^{\frac{1}{\Delta+2}-1} / \Delta\right)}.$$

Further, there exists a set-multilinear polynomial P with $\{0, 1\}$ -coefficients such that

$$\text{relrank}(P) \geq \frac{1}{n^{\Theta(1)}},$$

and P can be computed via evaluating the iterated matrix multiplication polynomial $\text{IMM}_{n,d}$ to carefully chosen linear forms.

We now give our characteristic-free version of the above.

► **Theorem 10.** Let \mathbb{F} be any field, and $d = o(\log n)$. Then $\text{IMM}_{n,d}$ requires

$$n^{\Theta\left(d^{\frac{1}{2\Delta+2}-1} / \Delta\right)}$$

size algebraic circuits of product depth Δ .

Proof. Let \bar{x} be a set of n variables. From Theorem 9, there exists a partition $\bar{x} = \bar{y}_1 \sqcup \cdots \sqcup \bar{y}_{d_y} \sqcup \bar{z}_1 \sqcup \cdots \sqcup \bar{z}_{d_z}$, only depending on n and d (and *not* on \mathbb{F}), where $d = d_y + d_z$, such that any $f(\bar{x})$ computed by a set-multilinear circuit of size s and product-depth Δ has relative rank bounded by

$$\text{relrank}(f) \leq s \cdot n^{-\Theta\left(d^{\frac{1}{\Delta+2}-1} / \Delta\right)}.$$

Further, there exists a set-multilinear polynomial P with $\{0, 1\}$ -coefficients such that

$$\text{relrank}(P) \geq \frac{1}{n^{\Theta(1)}},$$

and P can be computed via evaluating the iterated matrix multiplication polynomial $\text{IMM}_{n,d}$ to carefully chosen linear forms. In particular, any algebraic circuit lower bound for P extends, up to $\text{poly}(n)$ factors, to $\text{IMM}_{n,d}$, so it suffices to prove the lower bound for P .

Suppose P (interpreted as a polynomial in $\mathbb{F}[\bar{x}]$) is computed by an algebraic circuit Φ over \mathbb{F} of size s and product-depth Δ . Create a new algebraic circuit Ψ by replacing each field constant used in Φ with a distinct variable, so Ψ is size- s product-depth Δ algebraic circuit over the original variables \bar{x} along with new variables \bar{w} . Denote $f(\bar{x}, \bar{w})$ to be the polynomial computed by Ψ . As Ψ is defined free from field constants, f can be viewed as an integer polynomial $f \in \mathbb{Z}[\bar{x}, \bar{w}]$. Finally, we can relate P and f by undoing the above, so that there are values $\bar{\gamma}$ from \mathbb{F} such that $P(\bar{x}) = f(\bar{x}, \bar{\gamma})$.

Note that $f(\bar{x}, \bar{w})$ may not be set-multilinear, or even of particularly low-degree. However, it does follow that $\pi_{\text{sm}}(P(\bar{x})) = \pi_{\text{sm}}(f(\bar{x}, \bar{\gamma}))$ as the equality $P(\bar{x}) = f(\bar{x}, \bar{\gamma})$ is coefficient-wise, so applying π_{sm} to each side of equation either keeps the coefficient of P and f the same (if the monomial is set-multilinear) or makes both coefficients zero (if the monomial is not set-multilinear). As P is set-multilinear, we have $P = \pi_{\text{sm}}(P)$, so hence $P = \pi_{\text{sm}}(f(\bar{x}, \bar{\gamma}))$.

Now view Ψ as a circuit with constants over the field $\mathbb{Q}(\bar{w})$, so that $f \in \mathbb{Q}(\bar{w})[\bar{x}]$. It follows from Theorem 8 that $\pi_{\text{sm}}(f)$ has a $(d^d \cdot s)^{O(1)}$ -size set-multilinear circuit of product-depth 2Δ , and as such,

$$\text{relrank}_{\mathbb{Q}(\bar{w})}(\pi_{\text{sm}}(f(\bar{x}, \bar{w}))) \leq (d^d \cdot s)^{O(1)} \cdot n^{-\Theta\left(d^{\frac{1}{F_{2\Delta+2}^{-1}}}/\Delta\right)}.$$

Using that relative rank is just the (scaled) rank of a matrix, we can invoke Lemma 5, to see that

$$\text{relrank}_{\mathbb{F}}(\pi_{\text{sm}}(f(\bar{x}, \bar{\gamma}))) \leq \text{relrank}_{\mathbb{Q}(\bar{w})}(\pi_{\text{sm}}(f(\bar{x}, \bar{w}))).$$

As $P = \pi_{\text{sm}}(f(\bar{x}, \bar{\gamma}))$, we thus obtain that

$$\frac{1}{n^{\Theta(1)}} \leq \text{relrank}_{\mathbb{F}}(P) \leq (d^d \cdot s)^{O(1)} \cdot n^{-\Theta\left(d^{\frac{1}{F_{2\Delta+2}^{-1}}}/\Delta\right)}$$

which yields the desired lower bound for s (using that $d = o(\log n)$). ◀

4 Lower bounds over any field, constructively

In this section we give a constructive proof that any small low-depth algebraic circuit can be non-trivially set-multilinearized, over any field. As mentioned, by replacing the field-dependent set-multilinearization of LST with our set-multilinearization, this gives another proof of our main theorem (Theorem 10). The starting point for our construction is the *rectangular permanent*.

► **Definition 11.** Let X be an $n \times m$ symbolic matrix of variables, with $n \leq m$, where $X_{i,j} = x_{i,j}$ are distinct variables. Define the (**rectangular permanent**) $\text{perm}(X) \in \mathbb{Z}[(x_{i,j})_{i \in [n], j \in [m]}]$ by

$$\text{perm}_{n \times m}(X) = \sum_{\sigma: [n] \rightarrow [m]} x_{1, \sigma(1)} \cdots x_{n, \sigma(n)},$$

that is, the sum runs over all injective maps σ from $[n]$ to $[m]$.

Note in particular that $\text{perm}_{n \times m}$ is a set-multilinear polynomial when we partition the matrix into its rows.

It is sometimes helpful to view the rectangular permanent as a sum of square permanents.

31:10 Low-Depth Algebraic Circuit Lower Bounds over Any Field

► **Lemma 12.** *Let X be an $n \times m$ symbolic matrix of variables, with $n \leq m$, where $X_{i,j} = x_{i,j}$ are distinct variables. Then,*

$$\text{perm}_{n \times m}(X) = \sum_{S \in \binom{[m]}{n}} \text{perm}_{n \times n}(X|_{[n] \times S}).$$

The above lemma shows the rectangular permanent is computable by $\text{poly}(m^n)$ size algebraic circuits. However, the following identity gives a non-trivially better algorithm.

► **Theorem 13** (Binet-Minc Identity [17]). *Let X be an $n \times m$ symbolic matrix of variables, with $n \leq m$, then the permanent can be computed by*

$$\text{perm}_{n \times m}(X) = \sum_{\mathcal{F} \in \mathcal{P}_n} (-1)^{n-|\mathcal{F}|} \prod_{S \in \mathcal{F}} (|S| - 1)! \sum_{j=1}^m \prod_{i \in S} x_{i,j}$$

where \mathcal{P}_n is the set of all partitions of $[n]$ into (non-empty) sets, and $|\mathcal{F}|$ is the number of parts in the partition \mathcal{F} of $[n]$.

To understand the complexity of this expression it is helpful to have the following definition.

► **Definition 14.** *Define the n -th Bell number B_n to be the number of ways to partition $[n]$ into (non-empty) sets.*

We will use the following asymptotic estimate of Bell number size.

► **Fact 15** ([6]). $B_n = \Theta\left(\frac{n}{\ln n}\right)^n$.

The following lemma is easy to prove from the definition of Bell numbers.

► **Lemma 16.** $B_n \cdot B_m \leq B_{n+m}$.

The Binet-Minc identity immediately implies the following algebraic circuit for the permanent.

► **Corollary 17.** *The rectangular permanent $\text{perm}_{n \times m}$ has a $\sum^{B_n} \prod^n \sum^m \prod^n$ formula of size $\text{poly}(m, B_n)$, where the super-scripts are upper bounds on the respective fan-ins of the formula. Further, this formula is set-multilinear (and hence homogeneous) with respect to the partition of the $n \times m$ variables into rows.*

Note that for $m = n$ that this formula has complexity $\Theta\left(\frac{n}{\ln n}\right)^n$, whereas Ryser's formula is also set-multilinear but has size $\text{poly}(2^n)$ (and is depth-3).

We now note that when the rows of the matrix are identical, the Binet-Minc identity yields a small homogeneous depth-4 formula for the elementary symmetric polynomials, in large characteristic. The resulting formula has the same parameters as the argument of Shpilka and Wigderson [20], who proved it using the Newton identities. In particular, this relation is analogous to how Ryser's formula for the permanent, when applied to a matrix with identical rows, yields Fischer's depth-3 powering formula for the monomial.

► **Lemma 18.** *Let \bar{x} be n variables, and let Y be an $d \times n$ symbolic matrix of variables, with $d \leq n$, where $Y_{i,j} = x_j$. Then,*

$$\text{perm}_{d \times n}(Y) = d! \text{esym}_{n,d}(\bar{x})$$

Proof. $d = n$: This is immediate, as each monomial in the permanent now becomes $x_1 \cdots x_n$, and there are $n!$ many copies of this monomial.

$d < n$: Via Lemma 12,

$$\begin{aligned} \text{perm}_{d \times n}(Y) &= \sum_{S \in \binom{[n]}{d}} \frac{\text{perm}_{d \times d}(Y|_{[d] \times S})}{=d! \prod_{i \in S} x_i} \\ &= d! \text{esym}_{n,d}(\bar{x}). \end{aligned}$$

We now re-analyze the complexity of the above.

► **Corollary 19.** *Let \mathbb{F} be a field of characteristic $> d$ (or zero). The degree- d elementary symmetric polynomial in n variables $\text{esym}_{n,d}$ has a homogeneous $\sum^{2^{O(\sqrt{d})}} \prod^d \sum^n \wedge^d$ formula of size $\text{poly}(n, 2^{\sqrt{d}})$.*

Proof.

Construction, correctness: Apply the Binet-Minc identity (Theorem 13) to the $d \times n$ matrix Y where $Y_{i,j} = x_j$. By Lemma 18 this computes $d! \text{esym}_{n,d}$, and we can divide by $d!$ in \mathbb{F} .

Complexity: We now analyze the complexity of the above.

$$\begin{aligned} \text{perm}_{d \times n}(Y) &= \sum_{\mathcal{F} \in \mathcal{P}_d} (-1)^{d-|\mathcal{F}|} \prod_{S \in \mathcal{F}} (|S| - 1)! \sum_{j=1}^n \prod_{i \in S} \underbrace{Y_{i,j}}_{=x_j^{|S|}} \\ &= \sum_{\mathcal{F} \in \mathcal{P}_d} \underbrace{(-1)^{d-|\mathcal{F}|} \prod_{S \in \mathcal{F}} (|S| - 1)! \sum_{j=1}^n x_j^{|S|}}_{f_{\mathcal{F}}} \end{aligned}$$

noting that $f_{\mathcal{F}}$ only depends on the sizes of the how the partition \mathcal{F} refines the set $[n]$, we can group together the identical summands $f_{\mathcal{F}}$ based on how they partition the integer n ,

$$= \sum_{\bar{\lambda} \vdash d} (-1)^{d-|\bar{\lambda}|} N_{\bar{\lambda}} \prod_{\lambda \in \bar{\lambda}} (\lambda - 1)! \sum_{j=1}^n x_j^{\lambda}$$

where the summation runs over all integer partitions $\bar{\lambda}$ of d , $|\bar{\lambda}|$ is the number of parts in the partition $\bar{\lambda}$, and $N_{\bar{\lambda}} \in \mathbb{Z}$ is the number of set partitions of $[n]$ whose set sizes equal the integer partition $\bar{\lambda}$.

Now note that the above formula is homogeneous (as the Binet-Minc identity is), and the fan-ins of the formula are as desired because in particular the number of integer partitions of d is $2^{O(\sqrt{d})}$ ([6]). Finally, observe that the bottom-most product gate is a powering operation.

The above implies that Binet-Minc can recover that depth-3 circuits can be efficient homogenized into depth-5 circuits. As Binet-Minc holds over *all* fields and is additionally set-multilinear, this suggests that we can perhaps go directly to set-multilinearization, bypassing homogenization as an intermediate step. To do so, we need a slightly more general variant of the permanent.

► **Definition 20.** *Let X be an $n \times m$ symbolic matrix of variables, with $n \leq m$, where $X_{i,j} = x_{i,j}$ are distinct variables. Let $k \leq n$. Define the **k -surjective (rectangular) permanent** $\text{perm}_{n \times m; k}(X) \in \mathbb{Z}[(x_{i,j})_{i \in [n], j \in [m]}]$ by*

$$\text{perm}_{n \times m; k}(X) = \sum_{\sigma: [n] \hookrightarrow [m], \text{im}(\sigma) \supseteq [k]} x_{1, \sigma(1)} \cdots x_{n, \sigma(n)},$$

that is, the sum runs over all injective maps σ from $[n]$ to $[m]$ that contain $[k]$ in their image.

31:12 Low-Depth Algebraic Circuit Lower Bounds over Any Field

Note in particular that $\text{perm}_{n \times m; k}$ is a set-multilinear polynomial when we partition the matrix into its rows.

We can compute the surjective permanent by standard permanents.

► **Lemma 21.** *Let X be an $n \times m$ symbolic matrix of variables, with $n \leq m$, where $X_{i,j} = x_{i,j}$ are distinct variables. Let $k \leq n$. Then, the k -surjective permanent can be written as*

$$\text{perm}_{n \times m; k}(X) = \sum_{S \in \binom{[n]}{k}} \text{perm}_{k \times k}(X|_{S \times [k]}) \cdot \text{perm}_{(n-k) \times (m-k)}(X|_{([n] \setminus S) \times ([m] \setminus [k])}).$$

Further, this expression is set-multilinear with respect to partitioning X by its rows.

Proof. This follows from the observation that each map $\sigma : [n] \hookrightarrow [m]$ with $\text{im}(\sigma) \supseteq [k]$ uniquely decomposes into a bijection $\tau : S \leftrightarrow [k]$ for some $S \in \binom{[n]}{k}$, and an injection $\nu : ([n] \setminus S) \hookrightarrow ([m] \setminus [k])$.

The claim about set-multilinearity then follows from noting that the multiplication $\text{perm}_{k \times k}(X|_{S \times [k]}) \cdot \text{perm}_{(n-k) \times (m-k)}(X|_{([n] \setminus S) \times ([m] \setminus [k])})$ is a product of two polynomials who only use disjoint rows of X . ◀

We now analyze the complexity of computing the surjective permanent, by reduction to standard permanents, and then applying the Binet-Minc identity.

► **Corollary 22.** *Let $k \leq n \leq m$. The $n \times m$ k -surjective permanent has a $\text{poly}(m, \Theta(\frac{n}{\ln n})^n)$ -size depth-4 formula that is set-multilinear with respect to rows.*

Proof. Via the above lemma, and the formula complexity of the Binet-Minc identity,

$$\text{perm}_{n \times m; k}(X) = \sum_{S \in \binom{[n]}{k}} \frac{\text{perm}_{k \times k}(X|_{S \times [k]})}{\sum^{B_k} \prod^k \sum^k \prod^k} \cdot \frac{\text{perm}_{(n-k) \times (m-k)}(X|_{([n] \setminus S) \times ([m] \setminus [k])})}{\sum^{B_{n-k}} \prod^{n-k} \sum^{m-k} \prod^{n-k}}$$

distributing the multiplication past the addition,

$$= \frac{\sum_{S \in \binom{[n]}{k}} \sum^{B_k \cdot B_{n-k}} \left(\prod^k \sum^k \prod^k \right) \cdot \left(\prod^{n-k} \sum^{m-k} \prod^{n-k} \right)}{\prod^n \sum^m \prod^n}$$

from which the size bound follows by noting that $\binom{n}{k} B_k B_{n-k} \leq 2^n B_n \leq \Theta(\frac{n}{\ln n})^n$ via Lemma 16 and Fact 15.

The set-multilinearity of this formula follows from the fact that the decomposition used here from the above lemma is set-multilinear, that Binet-Minc is set-multilinear, and that our use of the distributive law preserves set-multilinearity. ◀

We now proceed to give a non-trivial set-multilinearization for low-depth algebraic circuits. To do so, it will be helpful to have more notation for extracting various set-multilinear components of polynomials.

► **Definition 23.** *Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \dots \sqcup \bar{x}_d$.*

*Let $S \subseteq [d]$. A monomial is **S -set-multilinear** if it can be written as $\prod_{i \in S} (\bar{x}_i)_{j_i}$ for some $(j_i)_{i \in S}$. Define the **S -set-multilinear projection** to be the linear map $\pi_{\text{sm}, S}$ which is identity on S -set-multilinear monomials, and zero on all other monomials.*

*The set of S -set-multilinear monomials for some $S \subseteq [d]$ are called **at most set-multilinear monomials**. Define the **non-set-multilinear projection** to be the linear map $\pi_{\neg \text{sm}}$ which is zero on at-most set-multilinear monomials, and identity on all other monomials.*

In particular, for $S = [d]$, $\pi_{\text{sm}}(f)$ and $\pi_{\text{sm},S}(f)$ are the same. For $S = \emptyset$, $\pi_{\text{sm},S}(f)$ is the constant part of f , $\pi_{\text{sm},\emptyset}(f) = f(\bar{0})$. More generally, we can decompose f into its set-multilinear parts as follows.

► **Lemma 24.** *Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. Then for any polynomial f ,*

$$f = \pi_{\neg\text{sm}}(f) + f(\bar{0}) + \sum_{\emptyset \neq S \subseteq [d]} \pi_{\text{sm},S}(f).$$

It immediately follows that we can simulate an addition (or even a linear combination) of polynomials by instead adding the constituent set-multilinear parts.

► **Lemma 25.** *Let f be any field. Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. Let f_1, \dots, f_m be polynomials, with $f = \alpha_1 f_1 + \cdots + \alpha_m f_m$ for $\alpha_1, \dots, \alpha_m \in \mathbb{F}$. Let $S \subseteq [d]$. Then $\pi_{\text{sm},S}(f)$ can be computed by a depth-1 $\text{poly}(m, 2^d)$ -size set-multilinear \sum -circuit given $\{\pi_{\text{sm},S}(f_j)\}_{S \subseteq [d], j \in [m]}$ as inputs.*

Less trivially is the simulation of a multiplication, for which we use the formula for the surjective permanent (Corollary 22).

► **Lemma 26.** *Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \cdots \sqcup \bar{x}_d$. Let f_1, \dots, f_m be polynomials, with $f = f_1 \cdots f_m$. Let $S \subseteq [d]$ be of size ℓ . Then $\pi_{\text{sm},S}(f)$ can be computed by a depth-4 $\text{poly}(m, \Theta(\frac{\ell}{\ln \ell})^\ell, 2^d)$ -size set-multilinear $\sum \Pi \sum \Pi$ -circuit given $\{\pi_{\text{sm},S}(f_j)\}_{S \subseteq [d], j \in [m]}$ as inputs.*

Proof. The number of inputs to the circuit is $m2^d$. It remains to bound the number of gates by $\text{poly}(m, \Theta(\frac{\ell}{\ln \ell})^\ell)$.

Rearrange the f_i as needed so that $f_1(\bar{0}) = \cdots = f_k(\bar{0}) = 0$ and $f_{k+1}(\bar{0}), \dots, f_m(\bar{0}) \neq 0$, for some $k \leq m$. Thus, we can normalize the computation via

$$f = \prod_{i>k} f_i(\bar{0}) \cdot \prod_{i \in [k]} f_i \cdot \prod_{i>k} \frac{f_i}{f_i(\bar{0})},$$

and thus define

$$g_i = \begin{cases} f_i & i \leq k \\ f_i/f_i(\bar{0}) & i > k \end{cases},$$

and $g = \prod_i g_i$. It then follows that $f = \prod_{i>k} f_i(\bar{0}) \cdot g$, $g_{k+1}(\bar{0}) = \cdots = g_m(\bar{0}) = 1$, and $\pi_{\text{sm},S}(f) = \prod_{i>k} f_i(\bar{0}) \cdot \pi_{\text{sm},S}(g)$. As $\prod_{i>k} f_i(\bar{0})$ is a non-zero constant, it suffices to prove the claim for $\pi_{\text{sm},S}(g)$.

Now write g as

$$g = \prod_{i=1}^k \left(\pi_{\neg\text{sm}}(g_i) + \sum_{\emptyset \neq S_i \subseteq [d]} \pi_{\text{sm},S_i}(g_i) \right) \cdot \prod_{i>k} \left(\pi_{\neg\text{sm}}(g_i) + 1 + \sum_{\emptyset \neq S_i \subseteq [d]} \pi_{\text{sm},S_i}(g_i) \right).$$

In expanding the above product, the only *at-most* set-multilinear terms are of the form

$$\prod_j \pi_{\text{sm},S_{i_j}}(g_{i_j}),$$

where the $S_{i_j} \subseteq [d]$ are disjoint (as otherwise we create non-multilinear terms), and the indices $\{i_j\}_j$ are distinct (as we take exactly one term from each g_i [possibly the term 1]).

31:14 Low-Depth Algebraic Circuit Lower Bounds over Any Field

Further, as $g_i(\bar{0}) = 0$ for $i \leq k$, we must have $\{i_j\}_j \supseteq [k]$. Hence, by collecting like terms, we see that

$$\pi_{\text{sm},S}(g) = \sum_{\mathcal{F} \in \mathcal{P}_S} \sum_{\sigma: \mathcal{F} \hookrightarrow [m]; \text{im}(\sigma) \supseteq [k]} \prod_{T \in \mathcal{F}} \pi_{\text{sm},T}(g_{\sigma(T)})$$

where \mathcal{P}_S is the collection of set partitions of S , which we can rewrite in terms of k -surjective permanents as

$$= \sum_{\mathcal{F} \in \mathcal{P}_S} \text{perm}_{|\mathcal{F}| \times m; k}(A_{\mathcal{F}}),$$

where for a partition \mathcal{F} of S the matrix $A_{\mathcal{F}}$ is $|\mathcal{F}| \times m$ size matrix defined by

$$(A_{\mathcal{F}})_{T,j} = \pi_{\text{sm},T}(g_j),$$

for $T \in \mathcal{F}$ and $j \in m$. Note that the matrix $A_{\mathcal{F}}$ has rows that access disjoint parts of the set-multilinear partition of \bar{x} , and so as the k -surjective permanent is set-multilinear with respect to rows, the computation $\text{perm}_{|\mathcal{F}| \times m; k}(A_{\mathcal{F}})$ is set-multilinear with respect to the partition of \bar{x} . Further, a set-multilinear computation of this permanent will be set-multilinear with respect to \bar{x} . Hence, we can use the set-multilinear computation from Corollary 22 for the surjective permanent, which in this case yields a depth-4 set-multilinear formula of size $\text{poly}(m, \Theta(\frac{|\mathcal{F}|}{\ln |\mathcal{F}|})^{|\mathcal{F}|})$.

Computing $\pi_{\text{sm},S}(g)$ is then a sum of $B_{|S|}$ many such $\text{perm}_{|\mathcal{F}| \times m; k}(A_{\mathcal{F}})$ terms, which does not increase depth as we collapse two sequential layers of addition gates, and this preserves set-multilinearity. The resulting size of the expression is $\text{poly}(m, \Theta(\frac{|\mathcal{F}|}{\ln |\mathcal{F}|})^{|\mathcal{F}|}, B_{|S|}, 2^d) \leq \text{poly}(m, \Theta(\frac{\ell}{\ln \ell})^{\ell}, 2^d)$, as $|S| = \ell$, $|\mathcal{F}| \leq |S|$. ◀

We now conclude with our set-multilinearization result over any field by gate-simulation.

► **Corollary 27.** *Let \mathbb{F} be any field. Let the variables \bar{x} be partitioned into $\bar{x} = \bar{x}_1 \sqcup \dots \sqcup \bar{x}_d$. Suppose $f \in \mathbb{F}[\bar{x}]$ can be computed by a size s product-depth Δ algebraic circuit. Then the set-multilinear projection $\pi_{\text{sm}}(f) \in \mathbb{F}[\bar{x}]$ can be computed by a size $\text{poly}(s, \Theta(\frac{d}{\ln d})^d)$ -size product-depth 2Δ circuit.*

Proof. By gate simulation. Let Φ be the hypothesized circuit computing f . For each node v in Φ , split v into its at-most set-multilinear parts $\pi_{\text{sm},S}(v)$ for each $S \subseteq [d]$.

If $v = \alpha_1 v_1 + \dots + \alpha_m v_m$, then we can express the at-most set-multilinear parts of v in terms of the v_i using a $\text{poly}(m, 2^d)$ -size product-depth 0 set-multilinear circuit by Lemma 25. If $v = v_1 \times \dots \times v_m$, then we can express the at-most set-multilinear parts of v in terms of the v_i using a $\text{poly}(m, \Theta(\frac{d}{\ln d})^d)$ -size product-depth 2 set-multilinear circuit by Lemma 26.

Correctness of the computation follows by induction on the circuit.

The overall size of the circuit has increased from s to $\text{poly}(s, \Theta(\frac{d}{\ln d})^d)$ by counting the size of the additional local gadgets of the gate simulation. Simulation of addition gates adds no product depth. Each product gate in the original circuit is turned into a product-depth 2 circuit in the gate simulation, hence the overall product-depth has at most doubled. ◀

References

- 1 Prashanth Amireddy, Ankit Garg, Neeraj Kayal, Chandan Saha, and Bhargav Thankey. Low-depth arithmetic circuit lower bounds: Bypassing set-multilinearization. In *Proceedings of the 50th International Colloquium on Automata, Languages and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:20, 2023. Full version in the Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR22-151. doi:10.4230/LIPICS.ICALP.2023.12.

- 2 Robert Andrews. Algebraic hardness versus randomness in low characteristic. In *Proceedings of the 35th Annual Computational Complexity Conference (CCC 2020)*, volume 169 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:32, 2020. Full version at [arXiv:2005.10885](https://arxiv.org/abs/2005.10885). doi:10.4230/LIPICS.CCC.2020.37.
- 3 C. S. Bhargav, Sagnik Dutta, and Nitin Saxena. Improved lower bound, and proof barrier, for constant depth algebraic circuits. In *Proceedings of the 47th International Symposium on the Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:16, 2022. doi:10.4230/LIPICS.MFCS.2022.18.
- 4 Radu Curticapean, Nutan Limaye, and Srikanth Srinivasan. On the VNP-hardness of some monomial symmetric polynomials. In *42nd International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2022)*, volume 250 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:14, 2022. Full version in the Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR22-139. doi:10.4230/LIPICS.FSTTCS.2022.16.
- 5 Ismor Fischer. Sums of like powers of multivariate linear forms. *Mathematics Magazine*, 67(1):59–61, 1994. URL: <http://www.jstor.org/stable/2690560>.
- 6 Philippe Flajolet and Robert Sedgewick. *Analytic combinatorics*. Cambridge University Press, 2009. doi:10.1017/CB09780511801655.
- 7 Michael A. Forbes, Amir Shpilka, and Ben Lee Volk. Succinct hitting sets and barriers to proving algebraic circuits lower bounds. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC 2017)*, pages 653–664, 2017. Full version at [arXiv:1701.05328](https://arxiv.org/abs/1701.05328). doi:10.1145/3055399.3055496.
- 8 Hervé Fournier, Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. On the power of homogeneous algebraic formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, TR23-191, 2023. URL: <https://eccc.weizmann.ac.il/report/2023/191>.
- 9 Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Zameret. Simple hard instances for low-depth algebraic proofs. In *Preliminary version in the 63rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2022)*, pages 188–199, 2022. Full version at [arXiv:2205.07175](https://arxiv.org/abs/2205.07175). doi:10.1109/FOCS54457.2022.00025.
- 10 Joshua A. Grochow, Mrinal Kumar, Michael E. Saks, and Shubhangi Saraf. Towards an algebraic natural proofs barrier via polynomial identity testing. *arXiv*, 1701.01717, 2017. URL: <http://arxiv.org/abs/1701.01717>.
- 11 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, pages 110–119, 2014. Full version at [arXiv:1404.3820](https://arxiv.org/abs/1404.3820). doi:10.1109/FOCS.2014.20.
- 12 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proceedings of Innovations in Theoretical Computer Science (ITCS 2013)*, pages 529–540, 2013. Full version at [arXiv:1208.5413](https://arxiv.org/abs/1208.5413). doi:10.1145/2422436.2422494.
- 13 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Satharishi. Arithmetic circuits: A chasm at depth three. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 578–587, 2013. Full version in the Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR13-026. doi:10.1109/FOCS.2013.68.
- 14 Erich L. Kaltofen. Factorization of polynomials given by straight-line programs. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 375–412. JAI Press, Inc., Greenwich, CT, USA, 1989. URL: http://www.math.ncsu.edu/~kaltopen/bibliography/89/Ka89_slpfac.pdf.
- 15 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *Preliminary version in the 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2021)*, pages 804–814, 2022. Full version in the Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR21-081. doi:10.1109/FOCS52979.2021.00083.

31:16 Low-Depth Algebraic Circuit Lower Bounds over Any Field

- 16 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Guest column: Lower bounds against constant-depth algebraic circuits. *SIGACT News*, 53(2):40–62, 2022. doi:10.1145/3544979.3544989.
- 17 Henryk Minc. Evaluation of permanents. *Proc. Edinburgh Math. Soc. (2)*, 22(1):27–32, 1979. doi:10.1017/S0013091500027760.
- 18 Ran Raz. Elusive functions and lower bounds for arithmetic circuits. *Theory of Computing*, 6(1):135–177, 2010. 40th Annual ACM Symposium on Theory of Computing (STOC 2008). doi:10.4086/TOC.2010.V006A007.
- 19 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Matematicheskie Zametki*, 41(4):598–607, April 1987.
- 20 Amir Shpilka and Avi Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001. Preliminary version in the 14th Annual IEEE Conference on Computational Complexity (CCC 1999). doi:10.1007/PL00001609.
- 21 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC 1987)*, pages 77–82, 1987. doi:10.1145/28395.28404.
- 22 Ryan Williams. Finding paths of length k in $O^*(2^k)$ time. *Inf. Process. Lett.*, 109(6):315–318, 2009. doi:10.1016/J.IPL.2008.11.004.

BPL \subseteq L-AC¹

Kuan Cheng   

Center on Frontiers of Computing Studies, School of Computer Science, Peking University, Beijing, China

Yichuan Wang   

Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Abstract

Whether $\text{BPL} = \text{L}$ (which is conjectured to be equal) or even whether $\text{BPL} \subseteq \text{NL}$, is a big open problem in theoretical computer science. It is well known that $\text{L} \subseteq \text{NL} \subseteq \text{L-AC}^1$. In this work we show that $\text{BPL} \subseteq \text{L-AC}^1$ also holds. Our proof is based on a new iteration method for boosting precision in approximating matrix powering, which is inspired by the Richardson Iteration method developed in a recent line of work [1, 28, 10, 17, 12, 25, 8]. We also improve the algorithm for approximate counting in low-depth L-AC circuits from an *additive error* setting to a *multiplicative error* setting.

2012 ACM Subject Classification Theory of computation \rightarrow Pseudorandomness and derandomization; Theory of computation \rightarrow Computational complexity and cryptography

Keywords and phrases Randomized Space Complexity, Circuit Complexity, Derandomization

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.32

Related Version *Full Version*: <https://ecc.weizmann.ac.il/report/2024/048/>

Acknowledgements We thank anonymous reviewers for helpful comments.

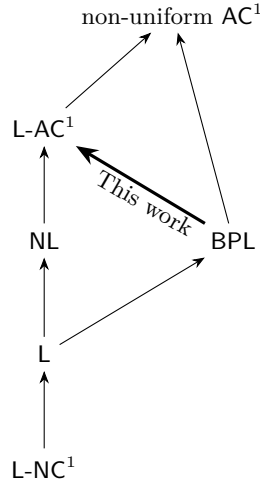
1 Introduction

BPL is the class of languages that can be computed by randomized logspace Turing Machines (TM) with error probability $\leq 1/3$. Here by *randomized* we mean that the TM has read-once access to a random tape. We also require that the TM halts on any input randomness. Whether $\text{BPL} \stackrel{?}{=} \text{L}$ is a big open problem of *space-bounded derandomization* in theoretical computer science. Most believe that $\text{L} = \text{BPL}$ is true. Different from *time-bounded derandomization*, we even do not know whether $\text{L} = \text{NL}$ implies $\text{L} = \text{BPL}$. But on the other hand, there is no known barrier for proving $\text{L} = \text{BPL}$. The seminal work by Saks and Zhou [29] shows that $\text{BPL} \subseteq \text{L}^{3/2}$ and this was improved to be $\text{BPL} \subseteq \text{SPACE} [O((\log n)^{3/2}/\sqrt{\log \log n})]$ by Hoza [17].

The relation between (Randomized) small space-bounded computation and uniform low-depth circuits is also an interesting topic. It is well known that $\text{L-NC}^1 \subseteq \text{L} \subseteq \text{NL} \subseteq \text{L-AC}^1$, where L-NC^1 and L-AC^1 are complexity classes of logspace-uniform $O(\log n)$ -depth NC and AC circuits. But for BPL, there is still an interesting question: is BPL also a subset of L-AC^1 ? If the conjecture $\text{L} = \text{BPL}$ is true, or even if $\text{BPL} \subseteq \text{NL}$, then immediately $\text{BPL} \subseteq \text{L-AC}^1$. But without these assumptions, it becomes a challenge. In this work, we will unconditionally prove that $\text{BPL} \subseteq \text{L-AC}^1$. On the other hand, we mention that the inclusion $\text{BPL} \subseteq \text{AC}^1$ for non-uniform AC^1 , is obvious via non-uniform derandomization techniques.¹ See Figure 1 for a visualization of the known relations between the complexity classes.

¹ There are two ways to prove $\text{BPL} \subseteq \text{non-uniform AC}^1$: (1) By $\text{L} \subseteq \text{AC}^1$ we know BPL can be computed by randomized AC^1 circuits, then apply the non-uniform derandomization for AC in [3] we know $\text{BPL} \subseteq \text{AC}^1$; (2) First by a standard non-uniform derandomization argument we have $\text{BPL} \subseteq \text{L}_{/\text{poly}}$, then by $\text{L} \subseteq \text{AC}^1$ we know $\text{BPL} \subseteq \text{L}_{/\text{poly}} \subseteq \text{AC}^1$.





■ **Figure 1** Relation of Complexity Classes. $A \rightarrow B$ means $A \subseteq B$.

One may view derandomizing BPL as the problem of approximating powers of substochastic matrices. For a TM with s bits of memory, one can label all its states by elements in $[2^s]$. We can define $\mathbf{A} \in \mathbb{R}^{2^s \times 2^s}$ to be its transition matrix in the sense that $\mathbf{A}_{i,j}$ is the probability that the machine moves from state i to state j in one step. Note that it must arrive at an *accept* or *reject* state in 2^s steps, so we only need to approximate \mathbf{A}^{2^s} . Saks and Zhou [29] showed that approximating \mathbf{A}^n for $\mathbf{A} \in \mathbb{R}^{w \times w}$ can be done in space $O((\log n)^{3/2} + \sqrt{\log n} \cdot \log w)$. Hoza [17] gave a logarithmic improvement in the $n = \text{poly}(w)$ regime, attaining $O(\frac{1}{\sqrt{\log \log n}} (\log n)^{3/2})$ space. Cohen, Doron, Sberlo and Ta-shma [12], and also Putterman and Pyne [25] independently improved [29]’s result to $\tilde{O}(\log n + \sqrt{\log n} \cdot \log w)$. We mention that a closely related problem setting is to approximate the multiplication of many distinct matrices, also called the iterated matrix multiplication (IMM) setting. This corresponds to the *read-once branching program* (ROBP) model. IMM asks one to approximate $\mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_n$ for a given sequence $\mathbf{A}_1, \dots, \mathbf{A}_n \in \mathbb{R}^{w \times w}$. [29, 17, 12, 25] can also work for IMM, attaining the same parameters respectively as their results for matrix powering. In [29] and [17], this is done via a simple block-box reduction from the powering setting. While in [12] and [25], a more careful analysis is applied. In the rest of our paper, we mainly consider matrix powering since this is enough for our main result and IMM is also in BPL. A key idea in [17, 12, 25] is to use Richardson Iteration to boost precision, which is developed in a line of work [1, 28, 10, 12, 25, 8]. We briefly recall this method here. Consider the problem of approximating \mathbf{X}^{-1} for an invertible matrix \mathbf{X} . Assume we already have a matrix \mathbf{Y} , which is an approximation of \mathbf{X}^{-1} such that $\|\mathbf{I} - \mathbf{YX}\| < \varepsilon$. Then we can rewrite $\mathbf{X}\mathbf{X}^{-1} = \mathbf{I}$ as

$$\mathbf{X}^{-1} = (\mathbf{I} - \mathbf{YX})\mathbf{X}^{-1} + \mathbf{Y}.$$

Start from $\mathbf{Y}^{(0)} = \mathbf{Y}$, by taking the iteration

$$\mathbf{Y}^{(i+1)} := (\mathbf{I} - \mathbf{YX})\mathbf{Y}^{(i)} + \mathbf{Y},$$

we can reduce $\|\mathbf{Y}^{(i)} - \mathbf{X}^{-1}\|$ very quickly. Then in the application of approximating $\mathbf{A}^1, \dots, \mathbf{A}^n$, we can take

$$\mathbf{X} := \begin{pmatrix} \mathbf{I} & & & & & \\ -\mathbf{A} & \mathbf{I} & & & & \\ & -\mathbf{A} & \mathbf{I} & & & \\ & & & \ddots & & \\ & & & & -\mathbf{A} & \mathbf{I} \end{pmatrix}, \mathbf{X}^{-1} = \begin{pmatrix} \mathbf{I} & & & & & \\ \mathbf{A} & \mathbf{I} & & & & \\ \mathbf{A}^2 & \mathbf{A} & \mathbf{I} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \mathbf{A}^{n-1} & \mathbf{A}^{n-2} & \mathbf{A}^{n-3} & \dots & \mathbf{I} & \\ \mathbf{A}^n & \mathbf{A}^{n-1} & \mathbf{A}^{n-2} & \dots & \mathbf{A} & \mathbf{I} \end{pmatrix}.$$

In this way, approximating $\mathbf{A}, \mathbf{A}^2, \dots, \mathbf{A}^n$ do not necessarily need to conduct approximating for their inverse matrices. Although Richardson Iteration is a powerful method, an interesting question is: are there any other iteration methods that have different or more powerful effects? In fact we develop a more efficient iteration algorithm for boosting precision for our setting in Section 4, which is the main ingredient of our proof of $\text{BPL} \subseteq \text{L-AC}^1$. The new iteration keeps using the idea of boosting precision via numerical analysis techniques, and it does not rely on approximating matrix inversions even in its analysis.

Another side of proving $\text{BPL} \subseteq \text{L-AC}^1$ is on the power of L-AC circuits. A key tool here is the approximate counting computable by L-AC circuits. Specifically, the task is to decide whether an n -bit string contains $\leq a$ or $\geq b$ 1's by $\text{poly}(n)$ -size low depth L-AC circuits. The L-AC^0 algorithm for distinguishing $\geq 2n/3$ 1's and $\leq n/3$ 1's was developed by Ajtai [2]. A line of work [3, 30, 31, 13] further studies this question, achieving depth $O\left(\frac{\log \frac{n}{b-a}}{\log \log n} + 1\right)$ (also see our Lemma 17). We will show that this can actually be done by $O\left(\frac{\log \frac{b}{b-a}}{\log \log n} + 1\right)$ -depth poly -size L-AC circuits. This can be viewed as improving the previous results from an *additive error* setting to a *multiplicative error* setting. But even so, one can see still that L-AC circuits are good at aggregating on *many* inputs, but not good at *high precision*. This triggers one to think of some steps of boosting precision potentially by iteration methods.

1.1 Our Result

► **Theorem 1** (Main Theorem, see also Corollary 21). $\text{BPL} \subseteq \text{L-AC}^1$.

► **Theorem 2** (Multiplicative Approximate Counting in AC, see also Theorem 14). *Let $n, a, b \in \mathbb{N}$ such that $0 \leq a < b \leq n$. Then there exists a $\text{poly}(n)$ -size $O\left(\frac{\log \frac{b}{b-a}}{\log \log n} + 1\right)$ -depth L-uniform AC circuit family $\{\mathcal{C}_{n,a,b}\}$ that computes $\text{GapMaj}[a, b]$ on n bits.²*

1.2 Related Work

We investigate some more about related work on derandomizing BPL. For other results not covered, we refer to these surveys [18][16].

A remarkable line of work [5, 22, 20, 24, 4, 6, 19, 7, 11, 27, 17] develops PRGs, weighted PRGs, and Hitting-set generators for ROBPs, which directly provide black-box derandomizations for BPL and its related classes. Among them, the celebrated work [22] by Nisan presents a logspace computable pseudorandom generator with seed length $O(\log n \log \frac{nw}{\varepsilon})$, error ε , for length n width w ROBPs. Based on some special properties of this generator, Saks and Zhou [29] showed $\text{BPL} \subseteq \text{L}^{3/2}$, and Nisan [23] showed that $\text{BPL} \subseteq \text{TISP}[\text{poly}(n), O((\log n)^2)]$. Nisan and Zuckerman [24] showed a PRG for ROBPs with large widths but very short lengths

² $\text{GapMaj}[a, b]$ is the promise problem that asks us to distinguish whether the number of 1's in an n -bit string is $\geq b$ or $\leq a$. See Definition 11 for a formal definition.

i.e. $n = \text{poly}(\log w)$, attaining seed length $O(\log w)$ with error $2^{-\log^{0.99} w}$. Armoni [4] gave an improved construction by interpolating [22] and [24]. The recent improvement for PRGs [6, 19, 7, 11, 27, 17] focus on achieving seed length optimal in the error parameter. Several iteration-type methods are applied by these work, including the use of Richardson Iteration developed by Ahmadijad, Kelner, Murtagh, Peebles, Sidford, and Vadhan [1], and then further developed in [28, 10, 17, 12, 25, 8].

There is also a sequence of work studying derandomizing BPL under assumptions. Klivans and van Melkebeek [21] showed that under the assumption that $\text{SPACE}[O(n)]$ requires $2^{\Omega(n)}$ circuit size, one can have $\text{L} = \text{BPL}$. Cheng and Hoza [9] showed that under the assumption that there exists a black-box hitting-set generator computable in logspace, one can have $\text{L} = \text{BPL}$. Some recent works [15, 26, 14] further study upon this line with various and enhanced requirements for derandomization.

1.3 Proof Overview

We sketch the proof of $\text{BPL} \subseteq \text{L-AC}^1$ and discuss the organization of our paper.

In Section 2, we describe some basic concepts and tools for our main proof.

In Section 3, we prove that deciding whether n bits contains $\leq a$ or $\geq b$ 1's can be done in $\text{poly}(n)$ -size $O\left(\frac{\log \frac{b-a}{a}}{\log \log n} + 1\right)$ -depth, see Theorem 14. This will be a building block for approximating matrix operations. The main idea to prove Theorem 14 is to reduce the general $\text{GapMaj}[a, b]$ to the special case $\text{GapMaj}[n/3, 2n/3]$ via pairwise independent hash functions, and then apply [2]'s algorithm for $\text{GapMaj}[n/3, 2n/3]$.

One can see that low-depth L-AC circuits are good at aggregating on *many* inputs, but without a *high precision*. This motivates us to consider a step of boosting precision for matrix powerings. However, we must be very careful since we cannot pay too much in depth.

In Section 4, we give the core iteration step. This is a depth-efficient iteration algorithm for boosting precision in matrix powerings, which is the main ingredient of our proof.

► **Theorem 3** (see also Theorem 18). *Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a substochastic matrix and $k, t \in \mathbb{Z}^+$ such that $\log n \geq k \geq t$. Suppose substochastic matrices $\mathbf{B}_0, \dots, \mathbf{B}_{k-1}$ are approximations of $\mathbf{A}^{2^0}, \dots, \mathbf{A}^{2^{k-1}}$ such that $\|\mathbf{B}_i - \mathbf{A}^{2^i}\|_1 \leq \varepsilon_i$ for $i = 1, 2, \dots, k-1$. Define*³

$$\begin{aligned} \mathbf{C} := & - \sum_{i=1}^{t-1} \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-i+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \mathbf{B}_{k-i}^2 \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q} \\ & + \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-t+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \mathbf{B}_{k-t}^2 \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q}. \end{aligned}$$

Then

$$\|\mathbf{C} - \mathbf{A}^{2^k}\|_1 \leq \sum_{i=1}^{t-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^t \varepsilon_{k-t}.$$

³ Here $\sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-i+1\}}}$ means taking the sum over all possible two-partitions of the set $\{k-1, k-2, \dots, k-i+1\}$. Each two-partition partitions $\{k-1, k-2, \dots, k-i+1\}$ into two disjoint subsets $\{j_1, \dots, j_p\}, \{j'_1, \dots, j'_q\}$. Here set elements are sorted in increasing order, i.e., $j_1 < \dots < j_p$ and $j'_1 < \dots < j'_q$. Therefore this \sum is sum of 2^{i-1} terms. When $i = 1$, $\{k-1, k-2, \dots, k-i+1\}$ represents the empty set.

Intuitively speaking, we can obtain a good approximation of \mathbf{A}^{2^k} given these $\mathbf{B}_{k-1}, \dots, \mathbf{B}_0$, which either has lower accuracy or is an approximation of $\mathbf{A}^{2^{k'}}$ for much smaller k' . We will prove that the iteration step can be easily computed by low-depth L-AC circuits in Theorem 19 which crucially uses our depth-efficient approximate counting in Section 3.

In Section 5 we present the complete algorithm. We compute intermediate matrices $\mathbf{M}(k, t)$ for $k, t \leq O(\log n)$, where $\mathbf{M}(k, t)$ is a $1/2^t$ -approximation of \mathbf{A}^{2^k} (i.e., $\|\mathbf{M}(k, t) - \mathbf{A}^{2^k}\|_1 \leq 1/2^t$). We will use the iteration step developed in Section 4 to show that, for any $k, t \leq O(\log n)$, given all $\mathbf{M}(k-i, [t/2] + 2i)$'s (for $i = 1, 2, \dots$), we can compute a valid $\mathbf{M}(k, t)$ in $O(t)$ -depth. Then we can compute a valid $\mathbf{M}(\log n, \log n)$ in $O(\log n)$ -depth.

Finally in Section 6 we will discuss some open problems.

2 Preliminaries

2.1 Matrix Approximation

► **Definition 4** (L_1 -norm). Define the L_1 -norm of a vector $(x_1, \dots, x_n)^\top \in \mathbb{R}^n$ to be

$$\|(x_1, \dots, x_n)^\top\|_1 := |x_1| + \dots + |x_n|.$$

Define the L_1 -norm of a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ to be

$$\|\mathbf{A}\|_1 := \sup_{\mathbf{x} \in \mathbb{R}^n} \frac{\|\mathbf{A}\mathbf{x}\|_1}{\|\mathbf{x}\|_1} = \max_{1 \leq j \leq n} \{|\mathbf{A}_{1,j}| + |\mathbf{A}_{2,j}| + \dots + |\mathbf{A}_{n,j}|\}.$$

► **Theorem 5.** For any $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$, we have:

1. $\|\mathbf{A} + \mathbf{B}\|_1 \leq \|\mathbf{A}\|_1 + \|\mathbf{B}\|_1$;
2. $\|\mathbf{A}\mathbf{B}\|_1 \leq \|\mathbf{A}\|_1 \|\mathbf{B}\|_1$;
3. If $\|\mathbf{A}\|_1, \|\mathbf{B}\|_1 \leq 1$, then for any $p \in \mathbb{Z}^+$, $\|\mathbf{A}^p - \mathbf{B}^p\|_1 \leq p \|\mathbf{A} - \mathbf{B}\|_1$.

► **Definition 6** (Non-negative Matrix). We say a matrix is non-negative if each of its entry is non-negative.

► **Definition 7** (Substochastic Matrix). We say a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a substochastic matrix if \mathbf{A} is non-negative and $\|\mathbf{A}\|_1 \leq 1$.

For simplicity, we always assume that the size of a substochastic matrix is a power of 2. To represent a substochastic matrix, we independently represent each entry in binary, accurate to $100 \log n$ decimal places.

2.2 L-uniform AC Circuit Family and Approximate Counting

► **Definition 8** (AC circuit). AC circuit is a circuit with input gates, NOT gates, unbounded fan-in AND/OR gates, and (possibly more than one) output gates. The size of a circuit is defined by the number of AND/OR gates. The depth of a circuit is defined by the largest number of AND/OR gates on any path from an input gate to an output gate.

► **Definition 9** (L-uniform AC circuit family). For functions $S, d: \mathbb{Z}^+ \rightarrow \mathbb{R}^+$, we say a collection of circuits $\{\mathcal{C}_n\}_{n \in \mathbb{Z}^+}$ is an S -size d -depth L-uniform AC circuit family, if each \mathcal{C}_n has size $\leq S(n)$ and depth $\leq d(n)$, and given the binary representation of n , the description of \mathcal{C}_n can be computed in uniform $O(\log n)$ -space.

We need to mention that the number of input gates in \mathcal{C}_n is not necessarily n . Also note that since we can encode a tuple of $O(1)$ many integers to a single integer, we can also consider circuit collections with a tuple of integers as an index.

► **Definition 10** (Complexity Class L-AC¹). *We say a language L is in the class L-AC¹ if there exists a poly(n)-size $O(\log n)$ -depth L-uniform AC circuit family $\{\mathcal{C}_n\}$ such that \mathcal{C}_n computes L on n -bit inputs.*

► **Definition 11** (GapMaj). *For $n \in \mathbb{Z}^+$ and $a, b \in \mathbb{R}$ such that $0 \leq a < b \leq n$, define the promise problem $\text{GapMaj}[a, b]$ on n bits as follow:*

$$\text{GapMaj}[a, b](x_1, \dots, x_n) := \begin{cases} \text{YES} & \text{if } x_1, \dots, x_n \text{ contains } \geq b \text{ 1's} \\ \text{NO} & \text{if } x_1, \dots, x_n \text{ contains } \leq a \text{ 1's} \\ \perp & \text{otherwise} \end{cases}$$

2.3 Tool: Pairwise Independent Hash Function

We will use pairwise independent hash function as a tool for approximate counting in AC. We shall use the following construction based on convolution, which was also used in [22].

► **Definition 12** (Convolution-Based Pairwise Independent Hash Function). *Suppose m is a power of 2. Define $H_m: [m^3] \times [m] \rightarrow [m]$ by: for $(k, x) \in [m^3] \times [m]$, let $x_1 \cdots x_{\log m}$ be the binary representation of $x - 1$, let $a_1 \cdots a_{2 \log m} b_1 \cdots b_{\log m}$ be the binary representation of $k - 1$, let $y_j := \left(\sum_{i=1}^{\log m} a_{i+j} x_i + b_j \right) \bmod 2$ for $j \in [\log m]$, then define $H_m(k, x)$ by letting $y_1 \cdots y_{\log m}$ be the binary representation of $H_m(k, x) - 1$.*

► **Theorem 13.** *H_m is Pairwise Independent Hash Function in the following sense: for any $1 \leq i < j \leq m$, when k is sampled from the uniform distribution over $[m^3]$, the joint distribution of $(H_m(k, i), H_m(k, j))$ is identical to the uniform distribution over $[m] \times [m]$.*

3 Approximate Counting in AC

The goal of this Section is to prove Theorem 14, which will be a building block for the proof of $\text{BPL} \subseteq \text{L-AC}^1$.

► **Theorem 14.** *Let $n, a, b \in \mathbb{N}$ such that $0 \leq a < b \leq n$. Then there exists a poly(n)-size $O\left(\frac{\log \frac{b}{b-a}}{\log \log n} + 1\right)$ -depth L-uniform AC circuit family $\{\mathcal{C}_{n,a,b}\}$ that computes $\text{GapMaj}[a, b]$ on n bits.*

The proof depends on the next few Lemmas.

► **Lemma 15** ([2]). *Let $n \in \mathbb{Z}^+$. Then there exists poly(n)-size $O(1)$ -depth L-uniform AC circuit family $\{\mathcal{C}_n^{(0)}\}$ that computes $\text{GapMaj}[n/3, 2n/3]$ on n bits.*

► **Lemma 16** (Exact Counting). *Let $n, \ell \in \mathbb{Z}^+$ such that $n \geq \ell$. Then there exists a poly(n)-size $O\left(\frac{\log \ell}{\log \log n} + 1\right)$ -depth L-uniform AC circuit family $\{\mathcal{E}_{n,\ell}\}$ such that on ℓ bits of input, $\mathcal{E}_{n,\ell}$ outputs the exact number of 1's over the input bits, in binary form.*

We remark that in Lemma 16, n is only used to bound the size of the circuit. Also, n, ℓ are not necessarily polynomially related.

Proof. We only need to show how to compute sum of $O(\sqrt{\log n})$ many $O(\log n)$ -bit⁴ non-negative integers in $O(1)$ -depth, then by divide-and-conquer we can compute sum of ℓ bits in $O\left(\frac{\log \ell}{\log \log n} + 1\right)$ -depth.

View the $O(\log n)$ -bit integers as $2^{\lceil \sqrt{\log n} \rceil}$ -base $O(\sqrt{\log n})$ -digit integers. We use the grade-school algorithm to sum $O(\sqrt{\log n})$ integers as follow. We first guess the result and all carry-bits, which involve at most $O(\sqrt{\log n}) \cdot O\left(\log\left(\sqrt{\log n} \cdot 2^{\lceil \sqrt{\log n} \rceil}\right)\right) = O(\log n)$ bits, and thus has at most $\text{poly}(n)$ choices. Then we can apply a local check on each digit, each local check involves at most $O(\log n)$ bits, and thus deciding whether all local checks are passed can be computed in $O(1)$ -depth. Then we can take the result of the only guess that passes all local checks. The total cost is $O(1)$ -depth. ◀

▶ **Lemma 17.** *Let $n, a, b \in \mathbb{N}$ such that $0 \leq a < b \leq n$. Then there exists a $\text{poly}(n)$ -size $O\left(\frac{\log \frac{n}{b-a}}{\log \log n} + 1\right)$ -depth L-uniform AC circuit family $\{\mathcal{C}_{n,a,b}^{(1)}\}$ that computes $\text{GapMaj}[a, b]$ on n bits.*

Proof. Only consider the case that n is a power of 2, otherwise we can use a simple padding argument. By Lemma 15, it suffices to show how to reduce $\text{GapMaj}[a, b]$ on n bits to $\text{GapMaj}[n^3/3, 2n^3/3]$ on n^3 bits, via a $\text{poly}(n)$ -size $O\left(\frac{\log \frac{n}{b-a}}{\log \log n} + 1\right)$ -depth L-uniform AC circuit.

If $b - a \leq 4\sqrt{n}$ then we can directly compute the number of 1's exactly via Lemma 16. Below we only consider $b - a > 4\sqrt{n}$.

Let $\ell := \left\lceil \frac{12n^2}{(b-a)^2} \right\rceil$. Suppose the $\text{GapMaj}[a, b]$ instance is x_1, x_2, \dots, x_n . Let H_n be the hash function defined in Definition 12. Define y_1, \dots, y_{n^3} as follow: for $i \in [n^3]$, let y_i be 1 if at least $\frac{a+b}{2n}$ fraction of $x_{H_n(i,1)}, \dots, x_{H_n(i,\ell)}$ is 1, otherwise let y_i be 0. Note that y_1, \dots, y_{n^3} can be computed via a $\text{poly}(n)$ -size $O\left(\frac{\log \ell}{\log \log n} + 1\right)$ -depth L-uniform AC circuit, by Lemma 16. Here $O\left(\frac{\log \ell}{\log \log n} + 1\right) = O\left(\frac{\log \frac{n}{b-a}}{\log \log n} + 1\right)$.

Let's do some simple calculations. Assume p fraction of x_1, \dots, x_n is 1. Let S_i be number of 1's in $x_{H_n(i,1)}, \dots, x_{H_n(i,\ell)}$. Then we have $\mathbb{E}_{i \sim [n^3]}[S_i] = p\ell$ and $\text{Var}_{i \sim [n^3]}[S_i] \leq \ell$. So if $p \leq \frac{a}{n}$, then $\Pr_{i \sim [n^3]} \left[S_i \geq \ell \cdot \frac{(a+b)}{2n} \right] \leq \frac{\ell}{\left(\ell \cdot \frac{(b-a)}{2n}\right)^2} = \frac{4n^2}{\ell(b-a)^2} \leq \frac{1}{3}$. Similarly if $p \geq \frac{b}{n}$ then $\Pr_{i \sim [n^3]} \left[S_i \leq \ell \cdot \frac{(a+b)}{2n} \right] \leq \frac{1}{3}$. This means if x_1, \dots, x_n is YES/NO instance of $\text{GapMaj}[a, b]$, then y_1, \dots, y_{n^3} is YES/NO instance of $\text{GapMaj}[n^3/3, 2n^3/3]$. The reduction is completed. ◀

Proof of Theorem 14. We will try to reduce to Lemma 17. Suppose the $\text{GapMaj}[a, b]$ instance is x_1, x_2, \dots, x_n . We only consider the case n is a power of 2, otherwise use a simple padding argument. We only consider the case $10\left(\frac{b}{b-a}\right)^2 < \frac{n}{b-a}$ (or equivalently, $n(b-a) > 10b^2$), otherwise we can directly apply Lemma 17.

Let $\ell := \left\lceil \frac{n(b-a)}{2b^2} \right\rceil$. For $i \in [n^3]$, let $y_i := x_{H_n(i,1)} \vee \dots \vee x_{H_n(i,\ell)}$, here H_n is the hash function defined in Definition 12. Then y_1, \dots, y_{n^3} can be computed via $\text{poly}(n)$ -size $O(1)$ -depth L-uniform AC circuit.

Assume p fraction of x_1, \dots, x_n is 1. Let S_i be number of 1's in $x_{H_n(i,1)}, \dots, x_{H_n(i,\ell)}$. Then we have $\mathbb{E}_{i \sim [n^3]}[S_i] = p\ell$ and $\mathbb{E}_{i \sim [n^3]}[S_i^2] = \ell(\ell-1)p^2 + \ell p \leq \ell p + \ell^2 p^2$. Thus by

$$\frac{\mathbb{E}_{i \sim [n^3]}[S_i^2]}{\mathbb{E}_{i \sim [n^3]}[S_i]^2} \leq \frac{\Pr_{i \sim [n^3]}[S_i \geq 1]}{\mathbb{E}_{i \sim [n^3]}[S_i]}$$

⁴ Here “ $O(\log n)$ -bit integers” refers to integers which has $O(\log n)$ -bits in its binary representation.

we know: if $p \leq \frac{a}{n}$, then $\Pr_{i \sim [n^3]}[S_i \geq 1] \leq \frac{\ell a}{n}$; if $p \geq \frac{b}{n}$, then $\Pr_{i \sim [n^3]}[S_i \geq 1] \geq \frac{(\frac{\ell b}{n})^2}{\frac{\ell b}{n} + (\frac{\ell b}{n})^2} \geq \frac{\ell b}{n} - (\frac{\ell b}{n})^2$. To summarize, if x_1, \dots, x_n is YES/NO instance of $\text{GapMaj}[a, b]$, then y_1, \dots, y_{n^3} is YES/NO instance of $\text{GapMaj}\left[\left[n^3 \cdot \frac{\ell a}{n}\right], \left[n^3 \cdot \left(\frac{\ell b}{n} - (\frac{\ell b}{n})^2\right)\right]\right]$.

Finally we observe that $\left(\frac{\ell b}{n} - (\frac{\ell b}{n})^2\right) - \frac{\ell a}{n} = \ell \cdot \left(\frac{b-a}{n} - \frac{\ell b^2}{n^2}\right) \geq \frac{n(b-a)}{3b^2} \cdot \frac{b-a}{2n} = \frac{(b-a)^2}{6b^2}$. Thus by Lemma 17, $\text{GapMaj}\left[\left[n^3 \cdot \frac{\ell a}{n}\right], \left[n^3 \cdot \left(\frac{\ell b}{n} - (\frac{\ell b}{n})^2\right)\right]\right]$ over n^3 bits can be computed via a $\text{poly}(n)$ -size $O\left(\frac{\log \frac{b}{b-a}}{\log \log n} + 1\right)$ -depth L-uniform AC circuit. \blacktriangleleft

4 The Iteration Method

In this section, we will introduce the iteration step, which is the core of our proof of $\text{BPL} \subseteq \text{L-AC}^1$.

► **Theorem 18** (The Iteration). *Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a substochastic matrix and $k, t \in \mathbb{Z}^+$ such that $\log n \geq k \geq t$. Suppose substochastic matrices $\mathbf{B}_0, \dots, \mathbf{B}_{k-1}$ are approximations of $\mathbf{A}^{2^0}, \dots, \mathbf{A}^{2^{k-1}}$ such that $\|\mathbf{B}_i - \mathbf{A}^{2^i}\|_1 \leq \varepsilon_i$ for $i = 1, 2, \dots, k-1$. Define*

$$\begin{aligned} \mathbf{C} := & - \sum_{i=1}^{t-1} \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-i+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \mathbf{B}_{k-i}^2 \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q} \\ & + \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-t+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \mathbf{B}_{k-t}^2 \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q}. \end{aligned}$$

Then

$$\|\mathbf{C} - \mathbf{A}^{2^k}\|_1 \leq \sum_{i=1}^{t-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^t \varepsilon_{k-t}.$$

Proof. Note that

$$\begin{aligned} \mathbf{C} - \mathbf{A}^{2^k} = & - \sum_{i=1}^{t-1} \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-i+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \left(\mathbf{A}^{2^{k-i}} - \mathbf{B}_{k-i}\right)^2 \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q} \\ & - \sum_{\substack{\{j_1 < \dots < j_p\} \uplus \{j'_1 < \dots < j'_q\} \\ = \{k-1, k-2, \dots, k-t+1\}}} \mathbf{B}_{j_p} \cdots \mathbf{B}_{j_1} \left(\mathbf{A}^{2^{k-t+1}} - \mathbf{B}_{k-t}^2\right) \mathbf{B}_{j'_1} \cdots \mathbf{B}_{j'_q}. \end{aligned}$$

So by Theorem 5,

$$\begin{aligned} \|\mathbf{C} - \mathbf{A}^{2^k}\|_1 & \leq \sum_{i=1}^{t-1} 2^{i-1} \|\mathbf{A}^{2^{k-i}} - \mathbf{B}_{k-i}\|_1^2 + 2^t \|\mathbf{A}^{2^{k-t+1}} - \mathbf{B}_{k-t}^2\|_1 \\ & \leq \sum_{i=1}^{t-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^t \varepsilon_{k-t}. \end{aligned} \quad \blacktriangleleft$$

► **Theorem 19** (Computing the Iteration). *Let $n, k, t, \mathbf{A}, \mathbf{B}_0, \dots, \mathbf{B}_{k-1}, \varepsilon_0, \dots, \varepsilon_{k-1}, \mathbf{C}$ be as defined in Theorem 18. Let d be an integer such that $4 \log n \geq d \geq t/10$. Then there exists a poly(n)-size $O(d)$ -depth L -uniform AC circuit family $\{\mathcal{I}_{n,k,t,d}\}$ such that on inputs $\mathbf{B}_{k-t}, \dots, \mathbf{B}_{k-1}$, if*

$$\sum_{i=1}^{t-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^t \varepsilon_{k-t} \leq \frac{1}{2^{d+2}}$$

is satisfied, then $\mathcal{I}_{n,k,t,d}$ outputs a substochastic matrix \mathbf{C}' such that $\|\mathbf{C}' - \mathbf{A}^{2^k}\|_1 \leq 1/2^d$.

The intuition behind Theorem 19 is that to approximately compute \mathbf{C} , all arithmetic operations only need a multiplicative accuracy of $1/2^{\Theta(d)}$. This can be done efficiently by L -uniform AC circuit by Theorem 14.

Proof of Theorem 19. We observe that \mathbf{C} is the sum of 2^{t-1} “+” terms and $2^{t-1} - 1$ “-” terms, and each term is a multiplication of not more than $t + 1$ substochastic matrices. We will first show how to approximate the multiplication of substochastic matrices and then show how to approximate their sum.

To approximate $\mathbf{Z} := \mathbf{X}\mathbf{Y}$ for two substochastic matrices \mathbf{X}, \mathbf{Y} , we only need to approximate $\sum_{r=1}^n \mathbf{X}_{i,r} \mathbf{Y}_{r,j}$ for each pair $(i, j) \in [n]^2$. We first represent each entry $\mathbf{X}_{i,r}, \mathbf{Y}_{r,j}$ using n^{100} bits such that fraction of 1’s in these n^{100} bits is equal to the entry, then use a layer of AND gate to represent each $\mathbf{X}_{i,r} \mathbf{Y}_{r,j}$ using fraction of 1’s in n^{200} bits, and then represent each $\frac{1}{n} \sum_{r=1}^n \mathbf{X}_{i,r} \mathbf{Y}_{r,j}$ using fraction of 1’s in n^{201} bits. Then we invoke $\mathcal{C}_{n^{201}, \ell, \lceil \ell(1+1/2^{20d+10}) \rceil}$ (as defined in Theorem 14, which has depth $\leq O\left(\frac{d}{\log \log n} + 1\right) \leq O\left(\frac{d}{\log(t+1)}\right)^5$ for $\ell = 1, 2, \dots, n^{200}$ over these n^{201} bits. Suppose ℓ_0 is the smallest index such that $\mathcal{C}_{n^{201}, \ell_0, \lceil \ell_0(1+1/2^{20d+10}) \rceil}$ outputs 0, then we have

$$\frac{\ell_0 - 1}{n^{200}} < \mathbf{Z}_{i,j} < \frac{\ell_0 \left(1 + \frac{1}{2^{20d+10}}\right)}{n^{200}}$$

and thus⁶

$$\frac{\mathbf{Z}_{i,j}}{1 + \frac{1}{2^{20d+10}}} - \frac{1}{n^{100}} \leq \frac{1}{n^{100}} \left[\frac{\ell_0}{n^{100}} \right] \leq \mathbf{Z}_{i,j}.$$

Use $[\ell_0/n^{100}]/n^{100}$ as an approximation of $\mathbf{Z}_{i,j}$, then we obtain an approximation $\tilde{\mathbf{Z}}$ of \mathbf{Z} such that $\mathbf{Z} - \tilde{\mathbf{Z}}$ is non-negative and $\tilde{\mathbf{Z}}$ is substochastic and $\|\mathbf{Z} - \tilde{\mathbf{Z}}\|_1 \leq 1/2^{20d+10} + 1/n^{99}$. We need to be careful that here we need a *multiplicative* small error on each entry and thus we need to strengthen Lemma 17 to Theorem 14.

Then multiplication of not more than $t + 1$ substochastic matrices can be computed via $O(\log(t + 1))$ layers of multiplication of two matrices. Recall that multiplying two matrices uses $O\left(\frac{d}{\log(t+1)}\right)$ -depth and has additive error $1/2^{20d+10} + 1/n^{99}$. So the total depth for computing multiplication of not more than $t + 1$ substochastic matrices is $O(d)$ and the total error is $\leq t(1/2^{20d+10} + 1/n^{99}) \leq 1/2^{19d+5}$.

⁵ In Theorem 14 we take $(a, b) = (\ell, \lceil \ell(1 + 1/2^{20d+10}) \rceil)$, and then $\log \frac{b}{b-a} \leq O(d)$.

⁶ Since $n^{200} \mathbf{Z}_{i,j}$ is an integer, we have $\frac{\ell_0 - 1}{n^{200}} < \mathbf{Z}_{i,j} \implies \frac{\ell_0}{n^{200}} \leq \mathbf{Z}_{i,j}$.

To summarize, suppose $\mathbf{C} = -\sum_{i=1}^{2^{t-1}-1} \mathbf{D}_i + \sum_{i=1}^{2^{t-1}} \mathbf{D}'_i$, here each $\mathbf{D}_i, \mathbf{D}'_i$ is multiplication of some substochastic matrices. Then we can compute their approximations $\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{D}}'_i$ in $O(d)$ depth such that $\left\| \mathbf{D}_i - \widetilde{\mathbf{D}}_i \right\|_1 \leq 1/2^{19d+5}$ and $\left\| \mathbf{D}'_i - \widetilde{\mathbf{D}}'_i \right\|_1 \leq 1/2^{19d+5}$.

We approximate $\frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}-1} \widetilde{\mathbf{D}}_i$ and $\frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}} \widetilde{\mathbf{D}}'_i$. Use the similar idea as summing $\frac{1}{n} \sum_{r=1}^n \mathbf{X}_{i,r} \mathbf{Y}_{r,j}$, we can compute substochastic matrices $\mathbf{C}^-, \mathbf{C}^+$ using $O(d)$ -depth, such that

$$\left\| \mathbf{C}^- - \frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}-1} \widetilde{\mathbf{D}}_i \right\|_1 \leq \frac{1}{2^{19d+5}},$$

$$\left\| \mathbf{C}^+ - \frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}} \widetilde{\mathbf{D}}'_i \right\|_1 \leq \frac{1}{2^{19d+5}}.$$

Then $2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-)$ is a good approximation of \mathbf{A}^{2^k} since

$$\begin{aligned} \left\| 2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-) - \mathbf{A}^{2^k} \right\|_1 &\leq 2^{t-1} \left\| \mathbf{C}^- - \frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}-1} \widetilde{\mathbf{D}}_i \right\|_1 + 2^{t-1} \left\| \mathbf{C}^+ - \frac{1}{2^{t-1}} \sum_{i=1}^{2^{t-1}} \widetilde{\mathbf{D}}'_i \right\|_1 \\ &\quad + \sum_{i=1}^{2^{t-1}-1} \left\| \mathbf{D}_i - \widetilde{\mathbf{D}}_i \right\|_1 + \sum_{i=1}^{2^{t-1}} \left\| \mathbf{D}'_i - \widetilde{\mathbf{D}}'_i \right\|_1 \\ &\quad + \left\| -\sum_{i=1}^{2^{t-1}-1} \mathbf{D}_i + \sum_{i=1}^{2^{t-1}} \mathbf{D}'_i - \mathbf{A}^{2^k} \right\|_1 \\ &\leq \frac{2^{t-1}}{2^{19d+5}} + \frac{2^{t-1}}{2^{19d+5}} + \frac{2^{t-1}}{2^{19d+5}} + \frac{2^{t-1}}{2^{19d+5}} + \left\| \mathbf{C} - \mathbf{A}^{2^k} \right\|_1 \\ &\leq \frac{1}{2^{9d+4}} + \left(\sum_{i=1}^{t-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^t \varepsilon_{k-t} \right) \\ &\leq \frac{1}{2^{9d+4}} + \frac{1}{2^{d+2}}. \end{aligned}$$

Here the last step is from the statement of Theorem 19.

Finally we compute a substochastic matrix \mathbf{C}' which is a good approximation of \mathbf{A}^{2^k} and $2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-)$. Here we need to be careful that \mathbf{C} and $2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-)$ are not necessarily non-negative or substochastic (but \mathbf{A}^{2^k} is guaranteed substochastic). Let

$$\mathbf{C}''_{i,j} := \max\{2^{t-1}(\mathbf{C}^+_{i,j} - \mathbf{C}^-_{i,j}), 0\},$$

$$\mathbf{C}'_{i,j} := \frac{1}{n^{100}} \left[\mathbf{C}''_{i,j} \left(1 - \frac{1}{2^{d+1}} \right) \cdot n^{100} \right].$$

We can compute \mathbf{C}' given $\mathbf{C}^+, \mathbf{C}^-$ by hardwiring the map $(\mathbf{C}^+_{i,j}, \mathbf{C}^-_{i,j}) \mapsto \mathbf{C}'_{i,j}$, which is L-uniform. Obviously \mathbf{C}' is non-negative. Note that \mathbf{C}'' is entrywise closer to \mathbf{A}^{2^k} than $2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-)$ and hence

$$\left\| \mathbf{C}'' - \mathbf{A}^{2^k} \right\|_1 \leq \left\| 2^{t-1}(\mathbf{C}^+ - \mathbf{C}^-) - \mathbf{A}^{2^k} \right\|_1 \leq \frac{1}{2^{9d+4}} + \frac{1}{2^{d+2}}$$

Therefore \mathbf{C}' is substochastic since

$$\left\| \mathbf{C}' \right\|_1 \leq \left(1 - \frac{1}{2^{d+1}} \right) \left\| \mathbf{C}'' \right\|_1 \leq \left(1 - \frac{1}{2^{d+1}} \right) \left(1 + \frac{1}{2^{9d+4}} + \frac{1}{2^{d+2}} \right) \leq 1.$$

Also note that

$$\begin{aligned}
\left\| \mathbf{C}' - \mathbf{A}^{2^k} \right\|_1 &\leq \left\| \mathbf{C}' - \mathbf{C}'' \right\|_1 + \left\| \mathbf{C}'' - \mathbf{A}^{2^k} \right\|_1 \\
&\leq \frac{1}{n^{99}} + \frac{1}{2^{d+1}} \left\| \mathbf{C}'' \right\|_1 + \left\| \mathbf{C}'' - \mathbf{A}^{2^k} \right\|_1 \\
&\leq \frac{1}{n^{99}} + \frac{1}{2^{d+1}} \left(1 + \frac{1}{2^{9d+4}} + \frac{1}{2^{d+2}} \right) + \frac{1}{2^{9d+4}} + \frac{1}{2^{d+2}} \\
&\leq \frac{1}{2^d}.
\end{aligned}$$

To summarize, we can output a valid \mathbf{C}' in $O(d)$ -depth. And the circuit is $\text{poly}(n)$ -size and L -uniform. \blacktriangleleft

5 The Complete Algorithm

► **Theorem 20.** *Let n be a power of 2. Then there exists a $\text{poly}(n)$ -size $O(\log n)$ -depth L -uniform AC circuit family $\{\mathcal{M}_n\}^7$ such that on input a substochastic matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, \mathcal{M}_n outputs a substochastic matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ such that $\|\mathbf{M} - \mathbf{A}^n\|_1 \leq 1/n$.*

Proof. Only consider $\log n \geq 10$. For $k, t \in \mathbb{N}$ such that $k \leq \log n$ and $1 \leq t \leq 3 \log n - 2k$, we wish to compute a substochastic matrix $\mathbf{M}(k, t)$, which is an approximation of \mathbf{A}^{2^k} , such that $\left\| \mathbf{M}(k, t) - \mathbf{A}^{2^k} \right\|_1 \leq 1/2^t$. Then $\mathbf{M} := \mathbf{M}(\log n, \log n)$ is the desired matrix.

For $k = 0$, we can trivially let $\mathbf{M}(0, t) := \mathbf{A}$. Now we show how to recursively compute $\mathbf{M}(k_0, t_0)$ for $k_0 = 1, 2, \dots, \log n$.

In Theorem 18, take the same n, \mathbf{A} and take $k := k_0$, take $\mathbf{B}_{k-i} := \mathbf{M}(k-i, \lceil t_0/2 \rceil + 2i)$ for $1 \leq i \leq k$. Then we can take $\varepsilon_{k-i} := 1/2^{\lceil t_0/2 \rceil + 2i}$ for $1 \leq i \leq k-1$ and $\varepsilon_0 = 0$. Now we will invoke Theorem 18, 19 by choosing parameter t properly according to the following two cases.

Case 1. $k \leq 2t_0 + 2$.

Take the parameter t in Theorem 18 to be $t := k$. Then

$$\sum_{i=1}^{k-1} 2^{i-1} \varepsilon_{k-i}^2 + 2^k \varepsilon_0 = \sum_{i=1}^{k-1} \frac{1}{2^{2\lceil t_0/2 \rceil + 3i + 1}} \leq \frac{1}{2^{t_0+2}}.$$

In Theorem 19 take $d := t_0$. It is easy to verify that $\log n \geq k \geq t$ and $4 \log n \geq d \geq t/10$ hold when we invoke Theorem 18, 19. Given $\mathbf{B}_{k-1}, \dots, \mathbf{B}_0$, use $\mathcal{I}_{n, k_0, k_0, t_0}$ (defined in Theorem 19) we can compute a substochastic matrix \mathbf{C}' such that $\left\| \mathbf{C}' - \mathbf{A}^{2^k} \right\|_1 \leq 1/2^{t_0}$.

Case 2. $k \geq 2t_0 + 3$.

Take $t := 2t_0 + 3$ in Theorem 18. Then

$$\sum_{i=1}^{2t_0+2} 2^{i-1} \varepsilon_{k-i}^2 + 2^{2t_0+3} \varepsilon_{k-2t_0-3} \leq \sum_{i=1}^{2t_0+2} \frac{1}{2^{2\lceil t_0/2 \rceil + 3i + 1}} + \frac{1}{2^{\lceil t_0/2 \rceil + 2t_0 + 3}} \leq \frac{1}{2^{t_0+2}}.$$

In Theorem 19 take $d := t_0$. Given $\mathbf{B}_{k-1}, \dots, \mathbf{B}_0$, use $\mathcal{I}_{n, k_0, 2t_0+3, t_0}$ we can compute a substochastic matrix \mathbf{C}' such that $\left\| \mathbf{C}' - \mathbf{A}^{2^k} \right\|_1 \leq 1/2^{t_0}$.

⁷ We require that given n , description of \mathcal{M}_n can be computed in space $O(\log n)$.

To summarize, take $\mathbf{M}(k_0, t_0) := \mathbf{C}'$, we can compute $\mathbf{M}(k_0, t_0)$ given $\mathbf{M}(k_0 - i, \lceil t_0/2 \rceil + 2i)$ for $1 \leq i \leq k_0$, via a $\text{poly}(n)$ -size $O(t_0)$ -depth L-uniform AC circuit.

Let $\gamma > 0$ be a concrete constant such that we can compute $\mathbf{M}(k_0, t_0)$ given $\mathbf{M}(k_0 - i, \lceil t_0/2 \rceil + 2i)$'s via a $\text{poly}(n)$ -size γt_0 -depth L-uniform AC circuit. Note that if $\mathbf{M}(k_0 - i, \lceil t_0/2 \rceil + 2i)$ can be computed in $2\gamma(2(k_0 - i) + (\lceil t_0/2 \rceil + 2i))$ -depth for $1 \leq i \leq k_0$, then $\mathbf{M}(k_0, t_0)$ can be computed in

$$\gamma t_0 + \max_{1 \leq i \leq k_0} \{2\gamma(2(k_0 - i) + (\lceil t_0/2 \rceil + 2i))\} \leq 2\gamma(2k_0 + t_0)$$

-depth. Also note that $\mathbf{M}(0, t_0)$'s are just the inputs, so by induction we know $\mathbf{M}(k_0, t_0)$ can be computed in $2\gamma(2k_0 + t_0)$ -depth. Specially, $\mathbf{M}(\log n, \log n)$ (which is the desired output) can be computed in $6\gamma \log n \leq O(\log n)$ -depth. Also note that we use “compute $\mathbf{M}(k_0, t_0)$ given $\mathbf{M}(k_0 - i, \lceil t_0/2 \rceil + 2i)$ ” $O((\log n)^2)$ many times, so the total circuit size for computing $\mathbf{M}(\log n, \log n)$ is still $\text{poly}(n)$. \blacktriangleleft

► **Corollary 21.** BPL \subseteq L-AC¹.

6 Open Problems

1. Our algorithm based on the improved iteration can be thought of as low-depth of *precision requirement*. Can this method be applied to obtain other interesting results in derandomizing BPL? It seems that the space-bounded model or nondeterministic space-bounded model cannot deal with low accuracy aggregating on many bits at low cost, as in the AC circuit model.
2. Our algorithm involves a “ $\times O(\log \log n)$ ” step when multiplying $O(\log n)$ matrices and a “ $/O(\log \log n)$ ” step in approximate counting in AC, which seems *coincidentally* achieves $O(\log n)$ -depth. Can we improve the algorithm to obtain an $O\left(\frac{\log n}{\log \log n}\right)$ -depth AC circuit for approximating powers of substochastic matrices? We need to mention that this does not imply BPL can be computed by $O\left(\frac{\log n}{\log \log n}\right)$ -depth AC circuits since we do not know whether L can be computed by $O\left(\frac{\log n}{\log \log n}\right)$ -depth AC circuits.

References

- 1 AmirMahdi Ahmadinejad, Jonathan A. Kelner, Jack Murtagh, John Peebles, Aaron Sidford, and Salil P. Vadhan. High-precision estimation of random walks in small space. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1295–1306. IEEE, 2020. doi:10.1109/FOCS46700.2020.00123.
- 2 Miklós Ajtai. Approximate counting with uniform constant-depth circuits. In Jin-Yi Cai, editor, *Advances In Computational Complexity Theory, Proceedings of a DIMACS Workshop, New Jersey, USA, December 3-7, 1990*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 1–20. DIMACS/AMS, 1990. doi:10.1090/DIMACS/013/01.
- 3 Miklós Ajtai and Michael Ben-Or. A theorem on probabilistic constant depth computations. In Richard A. DeMillo, editor, *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 471–474. ACM, 1984. doi:10.1145/800057.808715.
- 4 Roy Armoni. On the derandomization of space-bounded computations. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 47–59. Springer, 1998.

- 5 László Babai, Noam Nisan, and Mária Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992.
- 6 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, 2019.
- 7 Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. In *35th Computational Complexity Conference*, 2020.
- 8 Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1224–1239. IEEE, 2023. doi:10.1109/FOCS57990.2023.00072.
- 9 Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPICs*, pages 10:1–10:25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.CCC.2020.10.
- 10 Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error reduction for weighted prgs against read once branching programs. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPICs*, pages 22:1–22:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.CCC.2021.22.
- 11 Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error reduction for weighted prgs against read once branching programs. In *Proceedings of the 36th Computational Complexity Conference*, page 1, 2021.
- 12 Gil Cohen, Dean Doron, Ori Sberlo, and Amnon Ta-Shma. Approximating iterated multiplication of stochastic matrices in small space. 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 35–45. ACM, 2023. doi:10.1145/3564246.3585181.
- 13 Joshua Cook. Size bounds on low depth circuits for promise majority. In Nitin Saxena and Sunil Simon, editors, *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2020, December 14-18, 2020, BITS Pilani, K K Birla Goa Campus, Goa, India (Virtual Conference)*, volume 182 of *LIPICs*, pages 19:1–19:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.FSTTCS.2020.19.
- 14 Dean Doron, Edward Pyne, and Roei Tell. Opening up the distinguisher: A hardness to randomness approach for $BPL = L$ that uses properties of BPL. *Electron. Colloquium Comput. Complex.*, TR23-208, 2023. arXiv:TR23-208.
- 15 Dean Doron and Roei Tell. Derandomization with minimal memory footprint. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*, volume 264 of *LIPICs*, pages 11:1–11:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.CCC.2023.11.
- 16 Pooya Hatami and William Hoza. Theory of unconditional pseudorandom generators. *Electron. Colloquium Comput. Complex.*, TR23-019, 2023. arXiv:TR23-019.
- 17 William M. Hoza. Better pseudodistributions and derandomization for space-bounded computation. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPICs*, pages 28:1–28:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.APPROX/RANDOM.2021.28.
- 18 William M. Hoza. Recent progress on derandomizing space-bounded computation. *Bull. EATCS*, 138, 2022. URL: <http://eatcs.org/beatcs/index.php/beatcs/article/view/728>.
- 19 William M Hoza and David Zuckerman. Simple optimal hitting sets for small-success rl. *SIAM Journal on Computing*, 49(4):811–820, 2020.

- 20 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.
- 21 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.
- 22 Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992. doi:10.1007/BF01305237.
- 23 Noam Nisan. RL \subseteq SC. *Comput. Complex.*, 4:1–11, 1994. doi:10.1007/BF01205052.
- 24 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.
- 25 Aaron (Louie) Putterman and Edward Pyne. Near-optimal derandomization of medium-width branching programs. 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 23–34. ACM, 2023. doi:10.1145/3564246.3585108.
- 26 Edward Pyne, Ran Raz, and Wei Zhan. Certified hardness vs. randomness for log-space. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 989–1007. IEEE, 2023. doi:10.1109/FOCS57990.2023.00061.
- 27 Edward Pyne and Salil Vadhan. Pseudodistributions that beat all pseudorandom generators. In *36th Computational Complexity Conference (CCC 2021)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.
- 28 Edward Pyne and Salil P. Vadhan. Pseudodistributions that beat all pseudorandom generators (extended abstract). In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPICs*, pages 33:1–33:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.CCC.2021.33.
- 29 Michael E. Saks and Shiyu Zhou. $BP_{\text{H}}\text{SPACE}(S) \subseteq \text{DSPACE}(S^{3/2})$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999. doi:10.1006/JCSS.1998.1616.
- 30 Emanuele Viola. On approximate majority and probabilistic time. In *22nd Annual IEEE Conference on Computational Complexity (CCC 2007), 13-16 June 2007, San Diego, California, USA*, pages 155–168. IEEE Computer Society, 2007. doi:10.1109/CCC.2007.16.
- 31 Emanuele Viola. Randomness buys depth for approximate counting. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 230–239. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.19.

Failure of Feasible Disjunction Property for k -DNF Resolution and NP-Hardness of Automating It

Michal Garlík  

Department of Computing, Imperial College London, UK

Abstract

We show that for every integer $k \geq 2$, the $\text{Res}(k)$ propositional proof system does not have the weak feasible disjunction property. Next, we generalize a result of Atserias and Müller [3] to $\text{Res}(k)$. We show that if NP is not included in P (resp. QP, SUBEXP) then for every integer $k \geq 1$, $\text{Res}(k)$ is not automatable in polynomial (resp. quasi-polynomial, subexponential) time.

2012 ACM Subject Classification Theory of computation \rightarrow Proof complexity

Keywords and phrases reflection principle, feasible disjunction property, k -DNF Resolution

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.33

Related Version *Full Version:* <https://ecc.weizmann.ac.il/report/2020/037/>

Funding *Michal Garlík:* Supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program (grant agreement No. 101002742). Part of this work was carried out at the Euler International Mathematical Institute and supported by a grant from the Ministry of Science and Higher Education of the Russian Federation (agreement No. 075-15-2019-1620 dated 08/11/2019 and 075-15-2022-289 dated 06/04/2022). Part of this work was supported by European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme, grant agreement ERC-2014-CoG 648276 (AUTAR).

Acknowledgements I am grateful to Albert Atserias and Jan Krajíček for their valuable comments on an earlier version of the paper. I would like to thank Ilario Bonacina and Moritz Müller for several related conversations.

1 Introduction

Following Pudlák [20], a proof system P has *weak feasible disjunction property* if there exists a polynomial p such that if a formula $A \vee B$, in which A and B do not share variables, has a P proof of length t , then either A or B has a P proof of length $p(t)$. In this paper we deal with refutation systems, which for the previous definition amounts to replacing in it “ \vee ” by “ \wedge ” and “proof” by “refutation”. It is known and easy to see that resolution has the weak feasible disjunction property. Resolution also has feasible interpolation, a prominent concept in proof complexity introduced by Krajíček [11, 12]. A refutation system P has *feasible interpolation* if there is a polynomial p and an algorithm that when given as input a refutation Π of size r of a CNF $A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z})$, where $\bar{y}, \bar{x}, \bar{z}$ are disjoint sets of propositional variables, and a truth assignment σ to the variables \bar{x} outputs in time $p(r)$ a value $i \in \{0, 1\}$ such that if $i = 0$ then $A \upharpoonright \sigma$ is unsatisfiable and if $i = 1$ then $B \upharpoonright \sigma$ is unsatisfiable. Here $F \upharpoonright \sigma$ denotes the formula obtained from F by an application of a partial truth assignment σ to the variables of F that are in the domain of σ .

Pudlák [20] comments that so far the weak feasible disjunction property has been observed in all proof systems that were shown to have feasible interpolation. This is because known feasible interpolation algorithms, like those in Chapter 17.7 in [15], actually construct a refutation of one of the conjuncts.

A proof system P is *polynomially bounded* if there is a polynomial p such that any tautology of size r has a P proof of size $p(r)$. A fundamental problem in proof complexity is to show that no polynomially bounded proof system exists. This is equivalent to establishing



© Michal Garlík;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 33; pp. 33:1–33:23

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



$\text{NP} \neq \text{coNP}$, as observed by Cook and Reckhow [6]. There is a potentially useful observation by Krajíček [14] that for the purpose of proving that some proof system P is not polynomially bounded we may assume without a loss of generality that P admits the weak feasible disjunction property. This readily follows from the fact that if a disjunction of two formulas that do not share variables is a tautology, then one of the disjuncts is.

A propositional version of the negation of the *reflection principle* for a proof system P is a conjunction of a propositional formula expressing that ‘ \bar{z} is a satisfying assignment of formula \bar{x} of length r ’ and a propositional formula expressing that ‘ \bar{y} is a P refutation of length t of formula \bar{x} of length r ’. Here P, t, r are fixed parameters and $\bar{x}, \bar{y}, \bar{z}$ are disjoint sets of variables. When we plug in for the common variables \bar{x} some formula F of length r , we denote the conjunction by $\text{SAT}^F \wedge \text{REF}_{P,t}^F$, and we call the second conjunct a *P refutation statement for F* . We need to define one very mild requirement on a proof system in order to state a result from [20] about the weak feasible disjunction property that is the main source of motivation for this paper. We say that P is *closed under restrictions* if there is a polynomial p such that whenever F has a P proof of length t and σ is a partial truth assignment to the variables of F , then there is a P proof of $F \upharpoonright \sigma$ of length at most $p(t)$.

There is a proposition proved in [20] saying that if a proof system P has the weak feasible disjunction property, has polynomial-size proofs of the reflection principle for P , is closed under restrictions, and has the property that given a P proof of $\neg \text{SAT}^{\neg F}$ there is at most polynomially longer P proof of F , then for every formula F and every integer t which is at least the size of F , either there is a P proof of F of length $t^{O(1)}$, or there is a $t^{O(1)}$ long P proof of $\neg \text{REF}_{P,t}^F$. Pudlák comments that the conclusion of this proposition seems unlikely (and therefore it seems unlikely that a proof system satisfying the remaining three reasonable properties has the weak feasible disjunction property). He concludes that the weak feasible disjunction property is very unlikely to occur unless the system is very weak. Motivated to find and emphasize the contrast between resolution and $\text{Res}(2)$ (see Section 2) in this respect, we show the following theorem.

► **Theorem 1.** *For every integer $k \geq 2$, $\text{Res}(k)$ does not have the weak feasible disjunction property. Moreover, there are families $\{A_n\}_{n \geq 1}$ and $\{B_{n,k}\}_{n \geq 1, k \geq 1}$ of CNFs, where A_n has size $n^{O(1)}$, $B_{n,k}$ has size $n^{O(k)}$, and A_n and $B_{n,k}$ do not share any variables, such that all the following hold:*

- (i) *There exists $\alpha > 0$ and an integer n_1 such that for every $k \geq 1$ and $n \geq n_1$, any $\text{Res}(k)$ refutation of A_n has size greater than 2^{n^α} .*
- (ii) *For every $k \geq 1$ there is $\beta > 0$ and an integer n_2 such that for every $n \geq n_2$, any $\text{Res}(k)$ refutation of $B_{n,k}$ has size greater than $2^{\beta n}$.*
- (iii) *For all integers $n \geq 1$ and $k \geq 1$, $A_n \wedge B_{n,k}$ has a $\text{Res}(2)$ refutation of size $O(k^2 n^{7k+7})$.*

The idea is to employ a reflection, but instead of the reflection principle for $\text{Res}(k)$, which would correspond to the hypothesis of Pudlák’s proposition above, we work with the reflection principle for resolution and make it harder by the relativization technique of Dantchev and Riis [7]. More precisely, we replace in the reflection principle the resolution refutation statement by its k -fold relativization. Most of this paper is then concerned with proving length lower bounds on $\text{Res}(k)$ refutations of a version of the k -fold relativization of $\text{REF}_{\text{Res},t}^F$ for every unsatisfiable CNF F (Theorem 18). This lower bound will be used to prove Item ii above, but since it works for every unsatisfiable F , Item i will be easy to get choosing F to be hard enough for $\text{Res}(k)$. The upper bound, Item iii, generalizes upper bounds for similar formulas [2, 3, 8], which all build on an idea from [20].

To prove Theorem 18, the mentioned main lower bound, we develop a switching lemma in the spirit of [21] but respecting the functional properties of the formula $\text{REF}_{\text{Res},t}^F$. This will come at a cost of worse parameters in the switching lemma and its narrowed applicability in terms of random restrictions it works for.

Our second result is a generalization of conditional non-automatability results for resolution [3] to the systems $\text{Res}(k)$. Following [5, 2] and [3], we say that a refutation system P is *automatable in time* $T : \mathbb{N} \rightarrow \mathbb{N}$ if there is an algorithm that when given as input an unsatisfiable CNF F of size r outputs a P refutation of F in time $T(r + s_P(F))$, where $s_P(F)$ is the length of a shortest P refutation of F . If the function T is a polynomial, then P is simply called *automatable*. A refutation system P is *weakly automatable* if there is a refutation system Q , a polynomial p , and an algorithm that when given as input an unsatisfiable CNF F of size r outputs a Q refutation of F in time $p(r + s_P(F))$. It is known that feasible interpolation is implied by weak automatability in refutation systems that are closed under restrictions (see Theorem 3 in [2]).

First negative automatability results were obtained by Krajíček and Pudlák [16] who showed that Extended Frege systems do not have feasible interpolation assuming that RSA is secure against P/poly. Bonet et al. [5, 4] showed that Frege systems and constant-depth Frege systems do not have feasible interpolation assuming the Diffie-Hellman key exchange procedure is secure against polynomial and subexponential size circuits, respectively. All these proof systems are closed under restrictions, hence these results conditionally rule out weak automatability and automatability. As for resolution, before a recent breakthrough by Atserias and Müller [3] who showed that resolution is not automatable unless $P = NP$, it was known by a result of Alekhovich and Razborov [1] that resolution is not automatable unless $W[P] = FPT$. Here $W[P]$ is the class of parametrized problems that are fixed-parameter reducible to the problem of deciding if a monotone circuit C has a satisfying assignment of Hamming weight k . Regarding weak automatability, Atserias and Bonet [2] proved that for every $k > 1$, the following are equivalent: (i) $\text{Res}(k)$ has feasible interpolation, (ii) $\text{Res}(k)$ is weakly automatable, (iii) resolution is weakly automatable. We refer the interested reader to the introduction section of [3] for more on the history of the automatability problem.

Let QP denote the class of problems decidable in quasi-polynomial time $2^{(\log n)^{O(1)}}$, and let SUBEXP denote the class of problems decidable in subexponential time $2^{n^{o(1)}}$. We show the following theorem, which was proved for $k = 1$ in [3].

► **Theorem 2.**

1. If $NP \not\subseteq P$ then for every integer $k \geq 1$, $\text{Res}(k)$ is not automatable in polynomial time.
2. If $NP \not\subseteq QP$ then for every integer $k \geq 1$, $\text{Res}(k)$ is not automatable in quasi-polynomial time.
3. If $NP \not\subseteq \text{SUBEXP}$ then for every integer $k \geq 1$, $\text{Res}(k)$ is not automatable in subexponential time.

The basic idea of the proof is the same as in [3]: to map every formula F to a resolution refutation statement for F and to show that if F is satisfiable then the refutation statement has a polynomial-length $\text{Res}(k)$ refutation, and if F is unsatisfiable then the refutation statement requires long $\text{Res}(k)$ refutations. An automating algorithm that finds short refutations quickly enough can then be used to distinguish between the two situations, and hence to solve SAT. We thus need to show strong lower bounds on the length of $\text{Res}(k)$ refutations of a version of resolution refutation statements. To do this, we use the aforementioned Theorem 18 again.

Among the subsequent developments on non-automatability that appeared after [3] are NP-hardness of non-automatability of Cutting Planes [9], of Nullstellensatz and Polynomial Calculus [10], and of constant-depth Frege [18].

2 Preliminaries

For a function f , we write $\text{dom}(f)$ to denote its domain and $\text{im}(f)$ to denote its image. Functions f and g are *compatible* if $f \cup g$ is a function. If k is an integer, $[k]$ denotes the set $\{1, \dots, k\}$. For a propositional variable x , we denote by x^1 the *positive literal* of x , that is, x , and we denote by x^0 the *negative literal* of x , that is, $\neg x$. A *clause* (resp. *term*) is a set of literals, and is written as a disjunction (resp. conjunction) of its elements. A *CNF* is a set of clauses, and is written as a conjunction of the clauses. A CNF whose each clause contains at most k literals is called a *k -CNF*. A *DNF* is a set of terms, and is written as a disjunction of the terms. A *k -DNF* is a DNF in which each term has at most k literals. We will identify 1-DNFs with clauses. A clause that does not contain both the positive and negative literal of the same variable is called *non-tautological*. If C and D are clauses and $D \subseteq C$, we say that C is a *weakening* of D . We say that a clause D is the *resolvent* of clauses C_1 and C_2 on a variable x if $x \in C_1$, $\neg x \in C_2$ and $D = (C_1 \setminus \{x\}) \cup (C_2 \setminus \{\neg x\})$. A clause E is obtained by the *resolution rule* from clauses C_1 and C_2 if E is a weakening of the resolvent of C_1 and C_2 on a variable x . The clauses C_1 and C_2 are called the *premises* of the rule.

Let C be a clause and F a CNF. A sequence of clauses $\Pi = (C_1, \dots, C_t)$ is a *resolution derivation of C from F* if $C_t = C$ and for all $u \in [t]$, either C_u is a weakening of a clause in F , or there are $v, w \in [u - 1]$ such that C_u is obtained by the resolution rule from C_v and C_w . A *resolution refutation of F* is a resolution derivation of the empty clause from F . The *length* of a resolution derivation $\Pi = (C_1, \dots, C_t)$ is t . For $v \in [t]$, the *height of v in Π* is defined as the maximum integer h such that there is a subsequence $(C_{v_1}, \dots, C_{v_h})$ of Π in which $v_h = v$ and for each $i \in [h - 1]$, C_{v_i} is a premise of a resolution rule by which $C_{v_{i+1}}$ is obtained in Π . The *height of Π* is the maximum height of v in Π over $v \in [t]$.

Let x_1, \dots, x_n be propositional variables. A *partial assignment* to x_1, \dots, x_n is a partial map from $\{x_1, \dots, x_n\}$ to $\{0, 1\}$. For a partial assignment γ and a CNF F , we denote by $F \upharpoonright \gamma$ the CNF formed from F by removing every clause containing a literal satisfied by γ , and removing every literal falsified by γ from the remaining clauses. If $\Pi = (C_1, \dots, C_t)$ is a sequence of clauses, $\Pi \upharpoonright \gamma$ is formed from Π by the same operations. It is easy to check that if Π is a resolution refutation of F , then $\Pi \upharpoonright \gamma$ is a resolution refutation of $F \upharpoonright \gamma$.

The $\text{Res}(k)$ refutation system is a generalization of resolution introduced by Krajíček [13]¹. The lines in $\text{Res}(k)$ consist of k -DNFs. The inference rules of $\text{Res}(k)$ are the following (A, B are k -DNFs, $j \in [k]$, and l, l_1, \dots, l_j are literals):

$$\begin{array}{c} \frac{}{x \vee \neg x} \text{ Axiom} \\ \frac{A}{A \vee B} \text{ Weakening} \end{array} \quad \frac{A \vee l_1 \quad B \vee (l_2 \wedge \dots \wedge l_j)}{A \vee B \vee (l_1 \wedge \dots \wedge l_j)} \wedge\text{-introduction} \quad \frac{A \vee (l_1 \wedge \dots \wedge l_j) \quad B \vee \neg l_1 \vee \dots \vee \neg l_j}{A \vee B} \text{ Cut}$$

A *$\text{Res}(k)$ derivation from a CNF F* is a sequence of k -DNFs (D_1, \dots, D_t) so that each D_i either is a member of F or is obtained from the preceding lines by an application of one of the inference rules. A $\text{Res}(k)$ derivation (D_1, \dots, D_t) from F whose final line D_t is the empty clause is called a *$\text{Res}(k)$ refutation of F* . The *length* of a $\text{Res}(k)$ derivation $\Pi = (D_1, \dots, D_t)$, denoted by $|\Pi|$, is t . The *size* of Π , denoted by $\text{size}(\Pi)$, is the number of symbols in Π .

¹ In [13] (see also Chapter 5.7 in [15]) more general fragments $R(f)$ of DNF-resolution are introduced, where $f : \mathbb{N} \rightarrow \mathbb{N}$ is non-decreasing and a refutation Π is said to have $R(f)$ -size s if its lines are $f(s)$ -DNFs and $|\Pi| \leq s$. In the present paper we work with constant functions f .

3 Resolution Refutations of s Levels of t Clauses

It will be convenient to work with a variant of resolution in which the clauses forming a refutation are arranged in layers. All definitions in this section are taken from [8].

► **Definition 3.** *Let F be a CNF consisting of r clauses in n variables x_1, \dots, x_n . A resolution refutation of F of s levels of t clauses is a sequence of clauses $C_{i,j}$ indexed by all pairs $(i, j) \in [s] \times [t]$, such that each clause $C_{1,j}$ on the first level is a weakening of a clause in F , each clause $C_{i,j}$ on level $i \in [s] \setminus \{1\}$ is a weakening of the resolvent of two clauses from level $i - 1$ on a variable, and the clause $C_{s,t}$ is empty.*

The following proposition says that the requirement that clauses be arranged in layers is not very restrictive, since this system quadratically simulates resolution and preserves the height of the refutation.

► **Proposition 4** ([8]). *Assume that a $(n-1)$ -CNF F in n variables has a resolution refutation of height h and length s . Then F has a resolution refutation of h levels of $3s$ clauses.*

We now formalize refutation statements for this system in the same way as in [8]. Let n, r, s, t be integers. Suppose that F is a CNF consisting of r clauses C_1, \dots, C_r in n variables x_1, \dots, x_n . We define a propositional formula $\text{REF}_{s,t}^F$, which expresses that F has a resolution refutation of s levels of t clauses.

First we list the variables of $\text{REF}_{s,t}^F$. *D-variables* $D(i, j, \ell, b)$, $i \in [s]$, $j \in [t]$, $\ell \in [n]$, $b \in \{0, 1\}$, encode clauses $C_{i,j}$ as follows: $D(i, j, \ell, 0)$ (resp. $D(i, j, \ell, 1)$) means that the literal $\neg x_\ell$ (resp. x_ℓ) is in $C_{i,j}$. *R-variables* $R(i, j, j')$ (resp. *L-variables* $L(i, j, j')$), $i \in [s] \setminus \{1\}$, $j, j' \in [t]$, say that $C_{i-1, j'}$ is a premise of the resolution rule by which $C_{i,j}$ is obtained, and it is the premise that contains the negative (resp. positive) literal of the resolved variable. *V-variables* $V(i, j, \ell)$, $i \in [s] \setminus \{1\}$, $j \in [t]$, $\ell \in [n]$, say that $C_{i,j}$ is inferred by resolving on x_ℓ . *I-variables* $I(j, m)$, $j \in [t]$, $m \in [r]$, indicate that $C_{1,j}$ is a weakening of C_m .

The formula $\text{REF}_{s,t}^F$ (see Appendix A for a precise formalization) consists of clauses of several kinds: they express that clause $C_{1,j}$ (described by D -variables) on the first level contains all literals of the clause from F of which it is a weakening; that clauses $C_{i,j}$ are non-tautological; that the premises of the resolution rule contain the appropriate literal of the resolved variable and that all the other literals of the premises are passed to the clause inferred from them; that the last clause $C_{s,t}$ is empty; and that the V, I, L, R -variables define functions with appropriate domains and ranges.

► **Definition 5.** *For $i \in [s]$, $j, j' \in [t]$, $\ell \in [n]$, $b \in \{0, 1\}$, $m \in [r]$, we say that (i, j) is the home pair of the variable $D(i, j, \ell, b)$, of the variables $R(i, j, j')$, $L(i, j, j')$, $V(i, j, \ell)$ if $i \neq 1$, and of the variable $I(j, m)$ if $i = 1$.*

We write $V(i, j, \cdot)$ to stand for the set $\{V(i, j, \ell) : \ell \in [n]\}$. Similarly, we write $I(j, \cdot)$, $L(i, j, \cdot)$, and $R(i, j, \cdot)$ to stand for the set $\{I(j, m) : m \in [r]\}$, $\{L(i, j, j') : j' \in [t]\}$, and $\{R(i, j, j') : j' \in [t]\}$, respectively. We denote by $D(i, j, \cdot, \cdot)$ the set $\{D(i, j, \ell, b) : \ell \in [n], b \in \{0, 1\}\}$.

Let σ be a partial assignment. We say that $V(i, j, \cdot)$ is set to ℓ by σ if $\sigma(V(i, j, \ell)) = 1$ and for all $\ell' \in [n] \setminus \{\ell\}$, $\sigma(V(i, j, \ell')) = 0$. Similarly, we say that $I(j, \cdot)$ is set to m by σ if $\sigma(I(j, m)) = 1$ and for all $m' \in [r] \setminus \{m\}$ we have $\sigma(I(j, m')) = 0$. We say that $L(i, j, \cdot)$ (resp. $R(i, j, \cdot)$) is set to j' by σ if $\sigma(L(i, j, j')) = 1$ (resp. $\sigma(R(i, j, j')) = 1$) and for all $j'' \in [t] \setminus \{j'\}$, we have $\sigma(L(i, j, j'')) = 0$ (resp. $\sigma(R(i, j, j'')) = 0$). We say that $D(i, j, \cdot, \cdot)$ is set to a clause $C_{i,j}$ by σ if for all $\ell \in [n]$, $b \in \{0, 1\}$ we have $\sigma(D(i, j, \ell, b)) = 1$ if $x_\ell^b \in C_{i,j}$ and $\sigma(D(i, j, \ell, b)) = 0$ if $x_\ell^b \notin C_{i,j}$.

For $Y \in \{D(i, j, \cdot, \cdot), V(i, j, \cdot), I(j, \cdot), R(i, j, \cdot), L(i, j, \cdot)\}$, we say that Y is set by σ if Y is set to v by σ for some value v . We will sometimes omit saying “by σ ” if σ is clear from the context.

4 Reflection Principle for Resolution

We repeat the formulation of a version of the reflection principle from [8]. The CNF expressing the negation of the reflection principle for resolution can be written as $\text{SAT}^{n,r} \wedge \text{REF}_{s,t}^{n,r}$, the conjunction of two CNFs. The only common variables of the formulas $\text{SAT}^{n,r}$ and $\text{REF}_{s,t}^{n,r}$ encode a CNF with r clauses in n variables. The meaning of $\text{SAT}^{n,r}$ is that the encoded CNF is satisfiable, while the meaning of $\text{REF}_{s,t}^{n,r}$ is that the same CNF has a resolution refutation of s levels of t clauses. A formal definition is given below.

We list the variables of the formula $\text{SAT}^{n,r}$. C -variables $C(m, \ell, b)$, $m \in [r], \ell \in [n], b \in \{0, 1\}$, encode clauses C_m in the usual way: $C(m, \ell, 1)$ (resp. $C(m, \ell, 0)$) means that the literal x_ℓ (resp. $\neg x_\ell$) is in C_m . T -variables $T(\ell)$, $\ell \in [n]$, and $T(m, \ell, b)$, $m \in [r], \ell \in [n], b \in \{0, 1\}$, encode the truth value of the literals and clauses of the CNF $\{C_1, \dots, C_r\}$ (under an assignment to the variables x_1, \dots, x_n). The meaning of $T(\ell)$ is that the literal x_ℓ is satisfied. The meaning of $T(m, \ell, 1)$ (resp. $T(m, \ell, 0)$) is that clause C_m is satisfied through the literal x_ℓ (resp. $\neg x_\ell$).

The clauses of $\text{SAT}^{n,r}$ are the following:

$$T(m, 1, 1) \vee T(m, 1, 0) \vee \dots \vee T(m, n, 1) \vee T(m, n, 0) \quad m \in [r], \quad (1)$$

$$\neg T(m, \ell, 1) \vee T(\ell) \quad m \in [r], \ell \in [n], \quad (2)$$

$$\neg T(m, \ell, 0) \vee \neg T(\ell) \quad m \in [r], \ell \in [n], \quad (3)$$

$$\neg T(m, \ell, b) \vee C(m, \ell, b) \quad m \in [r], \ell \in [n], b \in \{0, 1\}. \quad (4)$$

For $m \in [r]$, clause (1) says that clause C_m is satisfied through some literal. Clauses (2) and (3) say that if C_m is satisfied through a literal, then the literal is satisfied. The meaning of (4) is that if clause C_m is satisfied through a literal, then it contains the literal.

Now let us take a look at $\text{REF}_{s,t}^{n,r}$. The variables of $\text{REF}_{s,t}^{n,r}$ are the variables $C(m, \ell, b)$ of $\text{SAT}^{n,r}$ together with all the variables of $\text{REF}_{s,t}^F$ for some (and each) F of r clauses in n variables. That is, $\text{REF}_{s,t}^{n,r}$ has the following variables: $C(m, \ell, b)$ for $m \in [r], \ell \in [n], b \in \{0, 1\}$; $D(i, j, \ell, b)$ for $i \in [s], j \in [t], \ell \in [n], b \in \{0, 1\}$; $L(i, j, j')$ and $R(i, j, j')$ for $i \in [s] \setminus \{1\}, j, j' \in [t]$; $V(i, j, \ell)$ for $i \in [s] \setminus \{1\}, j \in [t], \ell \in [n]$; $I(j, m)$ for $j \in [t], m \in [r]$.

The clauses of $\text{REF}_{s,t}^{n,r}$ are all clauses (12) - (25) of $\text{REF}_{s,t}^F$ together with the following set of clauses (to replace clauses (11)):

$$\neg I(j, m) \vee \neg C(m, \ell, b) \vee D(1, j, \ell, b) \quad j \in [t], m \in [r], \ell \in [n], b \in \{0, 1\}, \quad (5)$$

saying that if $C_{1,j}$ is a weakening of C_m , then the former contains each literal of the latter. So the difference with (11) is that C_m is no longer a clause of some fixed formula F , but is described by C -variables.

In Appendix B we show a simple proposition saying that evaluating $\text{SAT}^{n,r}$ by a partial assignment describing a CNF F with r clauses in n variables results in a formula equivalent to F .

5 The Upper Bounds

In this section we work with a stronger formulation of the reflection principle for resolution, expressed by a CNF formula $\text{SAT}^{n,r} \wedge \text{R}^k \text{REF}_{s,t}^{n,r}$. The difference with the previous formulation $\text{SAT}^{n,r} \wedge \text{REF}_{s,t}^{n,r}$ is that we replace $\text{REF}_{s,t}^{n,r}$ by its k -fold relativization

$R^k \text{REF}_{s,t}^{n,r}$. The first-order logic notion of relativization of a first-order formula to a relation was put to use in propositional proof complexity by Dantchev and Riis [7]. Informally speaking, relativization turns a statement that some property holds for the whole universe into a statement that the property holds for any non-empty subset S of the universe. In the propositional setting, this is realized by introducing new variables to encode the characteristic function of the subset S of the universe. We will actually use a conjunction of k fresh variables for each element of the universe to indicate the presence of the element in S .

We first treat the case of a fixed F , that is, we show how to relativize $\text{REF}_{s,t}^F$. The relativization of $\text{REF}_{s,t}^{n,r}$ will then be immediate.

The k -fold relativization of $\text{REF}_{s,t}^F$ is denoted by $R^k \text{REF}_{s,t}^F$. The variables of this CNF are those of $\text{REF}_{s,t}^F$ together with new variables $S_u(i, j)$, $(i, j) \in [s] \times [t]$, $u \in [k]$. The meaning of $R^k \text{REF}_{s,t}^F$ is that those clauses $C_{i,j}$ (described by D -variables) for which $\bigwedge_{u \in [k]} S_u(i, j)$ is satisfied form a resolution refutation of F of s levels of at most t clauses. That is, only the selected clauses $C_{i,j}$ have to form a refutation, and nothing is asked of the clauses that are not selected. Formally, $R^k \text{REF}_{s,t}^F$ is defined in Appendix C.

It is immediate that the partial assignment that maps $S_u(i, j)$ to 1 for all $(i, j) \in [s] \times [t]$ and all $u \in [k]$ maps $R^k \text{REF}_{s,t}^F$ to $\text{REF}_{s,t}^F$.

We now define the formula $R^k \text{REF}_{s,t}^{n,r}$ by a change to $R^k \text{REF}_{s,t}^F$ completely analogous to the change by which we obtained $\text{REF}_{s,t}^{n,r}$ from $\text{REF}_{s,t}^F$. That is, the clauses of $R^k \text{REF}_{s,t}^{n,r}$ are (27) - (43) of $R^k \text{REF}_{s,t}^F$ together with the following clauses (to replace (26)):

$$\bigvee_{u \in [k]} \neg S_u(1, j) \vee \neg I(j, m) \vee \neg C(m, \ell, b) \vee D(1, j, \ell, b) \quad j \in [t], m \in [r], \ell \in [n], b \in \{0, 1\}, \quad (6)$$

saying that if clause $C_{1,j}$ is selected and is a weakening of clause C_m (described by C -variables), then it contains each literal of C_m .

► **Theorem 6.** *The negation of the reflection principle for resolution expressed by the formula $\text{SAT}^{n,r} \wedge R^k \text{REF}_{s,t}^{n,r}$ has $\text{Res}(2)$ refutations of size $O(trn^2 + tr^2 + trnk + st^2n^3 + st^2n^2k + st^2nk^2 + st^3n)$.*

The proof of this theorem is in Appendix D

6 The Lower Bounds

We need a modification of two results of Segerlind, Buss and Impagliazzo [21]. Namely, their switching lemma works with the usual notion of the width of a clause, and we would like it to work with the notion of “the number of pairs mentioned” in the sense of Definition 9 below. This is because our random restrictions have to respect the functional properties of the formula $\text{REF}_{s,t}^F$ (expressed by clauses (18) - (25)), and it is therefore convenient to require that they evaluate variables in groups determined by the home pair. Consequently, we do not want to represent a k -DNF simplified by a random restriction by a standard decision tree like in [21], as such a tree would branch exponentially in t , which would prevent taking union bounds over the branches of shallow trees occurring in the proof of our switching lemma. To circumvent this problem, the decision trees we construct (called decision trees over $\text{REF}_{s,t}^F$) ask queries like “What is the left premise of clause $C_{i,j}$?” rather than queries like “Is $L(i, j, j')$ true?”. This makes their branching more manageable (though still exponential in the number of variables of F), but there is a price to pay in terms of the parameters

of the switching lemma (Theorem 16) and its more complicated proof, which uses certain independence properties of our random restrictions. Also, such trees no longer represent formulas over all partial assignments, but only over assignments that do not violate the functionality axioms and evaluate variables in groups determined by home pairs. Accordingly, we need to adapt to our different notions of width and representation a result in [21] which says that if the lines of a $\text{Res}(k)$ refutation can be strongly represented by shallow decision trees, then the refutation can be converted to a resolution refutation of a small width.

Our random restrictions (Definition 15) will be applied to k -DNFs in the variables of $\text{R}^k \text{REF}_{s,t}^F$ and they are defined in two stages, the first of which evaluates all S -variables, thereby declaring some pairs (i, j) selected (when $\bigwedge_{u \in [k]} S_u(i, j)$ evaluates to 1), and in the second stage all variables with a home pair that was not selected are evaluated randomly and independently. The restricted k -DNF formula is therefore in the variables of $\text{REF}_{s,t}^F$, and the purpose of the switching lemma is to show that it can be represented by a shallow decision tree over $\text{REF}_{s,t}^F$ with a high probability. We begin with a definition of these trees and the notion of representation. Before reading the next definition, it is useful to recall Definition 5.

► **Definition 7.** A decision tree over $\text{REF}_{s,t}^F$ is a rooted tree T in which every internal node is labelled with a pair $(i, j) \in [s] \times [t]$. There are $2^{2^n} \cdot r$ edges leaving each node labelled with $(1, j) \in \{1\} \times [t]$, and they are labelled with pairs $(C_{1,j}, m)$, where $C_{1,j}$ is a clause in variables x_1, \dots, x_n , and $m \in [r]$. There are $2^{2^n} \cdot nt^2$ edges leaving each node labelled with $(i, j) \in \{2, \dots, s\} \times [t]$, and these edges are labelled with tuples $(C_{i,j}, \ell, j', j'')$, where $C_{i,j}$ is a clause in variables x_1, \dots, x_n , $\ell \in [n]$, and $j', j'' \in [t]$. The leaves of T are labelled with either 0 or 1. No pair (i, j) is allowed to label two nodes on any path from the root to a leaf of T . For each node v of T , the path from the root to v is viewed as a partial assignment π_v that for each edge that is on the path, leaving a node with a label (i, j) , evaluates the variables of $\text{REF}_{s,t}^F$ with home pair (i, j) in the following way: If $i = 1$ and the label of the edge is $(C_{1,j}, m)$, then π_v sets $D(1, j, \cdot, \cdot)$ to $C_{1,j}$ and $I(j, \cdot)$ to m ; otherwise $i \in [s] \setminus \{1\}$ and the label of the edge is some tuple $(C_{i,j}, \ell, j', j'')$, in which case π_v sets $D(i, j, \cdot, \cdot)$ to $C_{i,j}$, $V(i, j, \cdot)$ to ℓ , $L(i, j, \cdot)$ to j' , and $R(i, j, \cdot)$ to j'' . For $b \in \{0, 1\}$, we let $\text{Br}_b(T)$ stand for the set of paths (viewed as partial assignments) that lead from the root to a leaf labelled with b .

► **Definition 8.** Let G be a DNF in the variables of $\text{REF}_{s,t}^F$. We say that a decision tree T over $\text{REF}_{s,t}^F$ strongly represents G if for every $\pi \in \text{Br}_0(T)$, for every $q \in G$, $q \upharpoonright \pi = 0$ and for every $\pi \in \text{Br}_1(T)$, there exists $q \in G$, $q \upharpoonright \pi = 1$. The representation index-height of G , $h_i(G)$, is the minimum height of a decision tree over $\text{REF}_{s,t}^F$ strongly representing G .

► **Definition 9.** Let π be a partial assignment to the variables of $\text{REF}_{s,t}^F$, and let E be a clause in the variables of $\text{REF}_{s,t}^F$. We say that a pair $(i, j) \in [s] \times [t]$ is mentioned in π (resp. E) if it is the home pair of a variable in $\text{dom}(\pi)$ (resp. of a variable a literal of which is in E).

► **Definition 10.** Let C be a clause in the variables of $\text{REF}_{s,t}^F$. The index-width of C is the number of pairs $(i, j) \in [s] \times [t]$ that are mentioned in C . The index-width of a resolution derivation is the maximum index-width of a clause in the derivation.

The following theorem is an adaptation of [21, Theorem 5.1]. Its proof is in Appendix E

► **Theorem 11.** Let H be a CNF in the variables of $\text{REF}_{s,t}^F$ whose every clause has index-width at most $h \geq 1$. If for some $k \geq 1$ there is a $\text{Res}(k)$ refutation of H such that for each line G of the refutation, $h_i(G) \leq h$, then there is a resolution refutation of H together with the functionality clauses (18) - (25) of $\text{REF}_{s,t}^F$ such that the index-width of the refutation is at most $3h$.

We now turn our attention to the formula $R^k \text{REF}_{s,t}^F$. Recall from its definition in Section 5 that its variables are those of $\text{REF}_{s,t}^F$ together with variables $S_u(i, j)$, $(i, j) \in [s] \times [t]$, $u \in [k]$. In the following definition we extend the notion of a home pair from Definition 5 to the S -variables, and we extend the notion of a pair being mentioned accordingly.

► **Definition 12.** For $(i, j) \in [s] \times [t]$ and $u \in [k]$, the home pair of the variable $S_u(i, j)$ is (i, j) .

We say that a pair (i, j) is mentioned in a clause E (resp. a partial assignment π ; a term q) if it is a home pair of a variable a literal of which is in E (resp. which is in $\text{dom}(\pi)$; a literal of which is in q).

► **Definition 13.** Let $U \subseteq [s] \times [t]$ and let G be a DNF in the variables of $R^k \text{REF}_{s,t}^F$. If for each term $q \in G$ there is $(i, j) \in U$ such that (i, j) is mentioned in q , then we say that U is an index-cover of G . The index-covering number of G , $c_i(G)$, is the minimum cardinality of an index-cover of G .

► **Definition 14.** For a set $U \subseteq [s] \times [t]$, denote by $\text{Var}(U)$ the set of all variables of $\text{REF}_{s,t}^F$ with home pair in U , that is,

$$\text{Var}(U) := \bigcup_{(i,j) \in U} D(i, j, \cdot, \cdot) \cup \bigcup_{(i,j) \in U \setminus ([1] \times [t])} (R(i, j, \cdot) \cup L(i, j, \cdot) \cup V(i, j, \cdot)) \cup \bigcup_{(1,j) \in U} I(j, \cdot).$$

Also, denote by $\text{Var}_S(U)$ the set of all S -variables with home pair in U ; in symbols, $\text{Var}_S(U) := \{S_u(i, j) : u \in [k], (i, j) \in U\}$.

We generalize random restrictions from [3] to our case of $R^k \text{REF}_{s,t}^F$.

► **Definition 15.** A random restriction ρ_k is a partial assignment to the variables of $R^k \text{REF}_{s,t}^F$ given by the following experiment:

1. Independently for each $(i, j) \in [s] \times [t]$ and $u \in [k]$, map $S_u(i, j)$ to 0 or 1, each with probability $1/2$.
2. Let A be the set of those $(i, j) \in [s] \times [t]$ such that for every $u \in [k]$, $S_u(i, j)$ is mapped to 1.
3. Map independently each variable from $\text{Var}([s] \times [t] \setminus A)$ to 0 or 1, each with probability $1/2$.

The main theorem of this section is the following switching lemma. Its proof is in Appendix F

► **Theorem 16.** Suppose that $k \geq 1, a \geq 1$ are integers such that $k \geq a$. There is $\delta > 0$ and an integer $n_0 > 0$ such that if n, r, s, t are integers satisfying

$$r \leq t \leq 2^{\delta n} \text{ and } n_0 \leq n, \tag{7}$$

and F is a CNF with r clauses in n variables, then for every a -DNF G in the variables of $R^k \text{REF}_{s,t}^F$ and every $w > 0$,

$$\Pr[h_i(G \upharpoonright \rho_k) > w] \leq 2^{-\frac{w}{n^{a-1}} \gamma(a)}, \tag{8}$$

where $\gamma(a) = \frac{(\log e)^a}{2^{a^2+3a-2a^1}}$.

The following theorem states a width lower bound. Its proof is in Appendix G

► **Theorem 17.** *Let $w > 0$. If n, r, s, t are integers satisfying*

$$2 \leq n + 1 \leq s, \quad 2w < t, \quad (9)$$

and F is an unsatisfiable CNF consisting of r clauses C_1, \dots, C_r in n variables x_1, \dots, x_n , then any resolution refutation of $\text{REF}_{s,t}^F$ has index-width greater than w .

We now put together all the results so far in this section to show a length lower bound on $\text{Res}(k)$ refutations of $\text{R}^k\text{REF}_{s,t}^F$ with an unsatisfiable F . The proof of the next theorem is in Appendix H.

► **Theorem 18.** *Suppose $k \geq 1$ is an integer. There is $\delta > 0$ and an integer $n_0 > 0$ such that if n, r, s, t are integers satisfying*

$$n_0 \leq n, \quad n + 1 \leq s \leq t, \quad r \leq t \leq 2^{\delta n}, \quad n^k \leq t, \quad (10)$$

and F is an unsatisfiable CNF consisting of r clauses C_1, \dots, C_r in n variables x_1, \dots, x_n , then any $\text{Res}(k)$ refutation of $\text{R}^k\text{REF}_{s,t}^F$ has length greater than $2^{\beta(k) \frac{t}{n^{k-1}}}$, where $\beta(k) := \frac{(\log e)^k}{2^{k^2+4k+4}k!}$.

From this theorem it is not difficult to derive the main results of this paper, Theorem 1 and Theorem 2. We include their proofs in Appendix I.

7 Conclusion

We have shown that for every integer $k \geq 2$, the system $\text{Res}(k)$ does not have the weak feasible disjunction property and, unless $\text{P} = \text{NP}$, it is not automatable. Because of the factor t/n^{k-1} that appears in the exponent of the lower bound in Theorem 18 and originates in the switching lemma (Theorem 16), we have not been able to extend the results to superconstant k .

A more important open question is to rule out weak automatability of these systems assuming some standard hardness assumption. Weak automatability behaves better; e.g. in contrast to automatability, it trivially follows that if a proof system P simulates Q and P is weakly automatable, then so is Q .

References

- 1 Michael Alekhovich and Alexander A. Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM Journal on Computing*, 38(4):1347–1363, 2008. doi:10.1137/06066850X.
- 2 Albert Atserias and Maria Luisa Bonet. On the automatizability of resolution and related propositional proof systems. *Information and Computation*, 189(2):182–201, 2004.
- 3 Albert Atserias and Moritz Müller. Automating resolution is NP-hard. In *60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 498–509. IEEE, 2019.
- 4 Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, Toniann Pitassi, and Ran Raz. Non-automatizability of bounded-depth Frege proofs. *Computational Complexity*, 13(1-2):47–68, 2004. doi:10.1007/s00037-004-0183-5.
- 5 Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for Frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000. doi:10.1137/S0097539798353230.
- 6 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.
- 7 Stefan Dantchev and Søren Riis. On relativisation and complexity gap for resolution-based proof systems. In M. Baaz and J. A. Makowsky, editors, *Computer Science Logic (CSL 2003)*, volume 2803 of *Lecture Notes in Computer Science*, pages 142–154. Springer, 2003.

- 8 Michal Garlík. Resolution Lower Bounds for Refutation Statements. In Peter Rossmanith, Pinar Heggenes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019)*, volume 138 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.MFCS.2019.37.
- 9 M Göös, S Koroth, I Mertz, and T Pitassi. Automating cutting planes is np-hard. *STOC 2020: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 2020. doi:10.1145/3357713.3384248.
- 10 M Göös, J Nordström, T Pitassi, R Robere, D Sokolov, and S deRezende. Automating algebraic proof systems is np-hard. *Preprint available at the Electronic Colloquium on Computational Complexity (ECCC), TR20-064*, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/064/>.
- 11 Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *The Journal of Symbolic Logic*, 59(1):73–86, 1994. doi:10.2307/2275250.
- 12 Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(2):457–486, 1997. doi:10.2307/2275541.
- 13 Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170(1-2):123–140, 2001. doi:10.4064/fm170-1-8.
- 14 Jan Krajíček. On the proof complexity of the Nisan–Wigderson generator based on a hard $NP \cap coNP$ function. *Journal of Mathematical Logic*, 11(01):11–27, 2011. doi:10.1142/S0219061311000979.
- 15 Jan Krajíček. *Proof Complexity*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2019. doi:10.1017/9781108242066.
- 16 Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for S_2^1 and EF. *Information and Computation*, 140(1):82–94, 1998. doi:10.1006/inco.1997.2674.
- 17 Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth frege proofs of the pigeonhole principle. *Random Structures & Algorithms*, 7(1):15–39, 1995. doi:10.1002/rsa.3240070103.
- 18 Theodoros Papamakarios. Depth d frege systems are not automatable unless $p = np$. *Preprint available at the Electronic Colloquium on Computational Complexity (ECCC), TR23-121*, 2023. URL: <https://eccc.weizmann.ac.il/report/2023/121/>.
- 19 Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational Complexity*, 3(2):97–140, 1993. doi:10.1007/BF01200117.
- 20 Pavel Pudlák. On reducibility and symmetry of disjoint NP-pairs. *Theoretical Computer Science*, 295:323–339, 2003.
- 21 Nathan Segerlind, Samuel R. Buss, and Russel Impagliazzo. A switching lemma for small restrictions and lower bounds for k-DNF resolution. *SIAM Journal on Computing*, 33(5):1171–1200, 2004.

A Formula $REF_{s,t}^F$

The formula $REF_{s,t}^F$ is the union of the following fifteen sets of clauses:

$$\neg I(j, m) \vee D(1, j, \ell, b) \quad j \in [t], m \in [r], b \in \{0, 1\}, x_\ell^b \in C_m, \quad (11)$$

clause $C_{1,j}$ contains all literals of C_m if $C_{1,j}$ is a weakening of C_m ,

$$\neg D(i, j, \ell, 0) \vee \neg D(i, j, \ell, 1) \quad i \in [s], j \in [t], \ell \in [n], \quad (12)$$

clause $C_{i,j}$ cannot contain both $\neg x_\ell$ and x_ℓ ,

$$\neg L(i, j, j') \vee \neg V(i, j, \ell) \vee D(i-1, j', \ell, 1) \quad i \in [s] \setminus \{1\}, j, j' \in [t], \ell \in [n], \quad (13)$$

$$\neg R(i, j, j') \vee \neg V(i, j, \ell) \vee D(i-1, j', \ell, 0) \quad i \in [s] \setminus \{1\}, j, j' \in [t], \ell \in [n], \quad (14)$$

33:12 Failure of Feasible Disjunction Property and Automatability in k -DNF Resolution

clause $C_{i-1,j'}$ used as the premise given by $L(i, j, j')$ (resp. $R(i, j, j')$) when resolving on x_ℓ must contain x_ℓ (resp. $\neg x_\ell$),

$$\begin{aligned} \neg L(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell', b) \vee D(i, j, \ell', b) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell, \ell' \in [n], b \in \{0, 1\}, (\ell', b) \neq (\ell, 1), \end{aligned} \quad (15)$$

$$\begin{aligned} \neg R(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell', b) \vee D(i, j, \ell', b) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell, \ell' \in [n], b \in \{0, 1\}, (\ell', b) \neq (\ell, 0), \end{aligned} \quad (16)$$

clause $C_{i,j}$ inferred by resolving on x_ℓ must contain each literal different from x_ℓ (resp. $\neg x_\ell$) from the premise given by $L(i, j, j')$ (resp. $R(i, j, j')$),

$$\neg D(s, t, \ell, b) \quad \ell \in [n], b \in \{0, 1\}, \quad (17)$$

clause $C_{s,t}$ is the empty clause,

$$V(i, j, 1) \vee V(i, j, 2) \vee \dots \vee V(i, j, n) \quad i \in [s] \setminus \{1\}, j \in [t], \quad (18)$$

$$I(j, 1) \vee I(j, 2) \vee \dots \vee I(j, r) \quad j \in [t], \quad (19)$$

$$L(i, j, 1) \vee L(i, j, 2) \vee \dots \vee L(i, j, t) \quad i \in [s] \setminus \{1\}, j \in [t], \quad (20)$$

$$R(i, j, 1) \vee R(i, j, 2) \vee \dots \vee R(i, j, t) \quad i \in [s] \setminus \{1\}, j \in [t], \quad (21)$$

$$\neg V(i, j, \ell) \vee \neg V(i, j, \ell') \quad i \in [s] \setminus \{1\}, j \in [t], \ell, \ell' \in [n], \ell \neq \ell', \quad (22)$$

$$\neg I(j, m) \vee \neg I(j, m') \quad j \in [t], m, m' \in [r], m \neq m', \quad (23)$$

$$\neg L(i, j, j') \vee \neg L(i, j, j'') \quad i \in [s] \setminus \{1\}, j, j', j'' \in [t], j' \neq j'', \quad (24)$$

$$\neg R(i, j, j') \vee \neg R(i, j, j'') \quad i \in [s] \setminus \{1\}, j, j', j'' \in [t], j' \neq j'', \quad (25)$$

the V, I, L, R -variables define functions.

B Obtaining F from $\text{SAT}^{n,r}$

► **Proposition 19.** *Let F be a CNF of r clauses C_1, \dots, C_r in n variables x_1, \dots, x_n , and let γ_F be the assignment describing F , that is, the domain of γ_F consists of all C -variables, and $\gamma_F(C(m, \ell, b)) = 1$ if $x_\ell^b \in C_m$ and $\gamma_F(C(m, \ell, b)) = 0$ if $x_\ell^b \notin C_m$. Then there is a substitution τ that maps the T -variables of $\text{SAT}^{n,r} \upharpoonright \gamma_F$ to $\{0, 1\} \cup \{x_\ell^b : \ell \in [n], b \in \{0, 1\}\}$ such that $(\text{SAT}^{n,r} \upharpoonright \gamma_F) \upharpoonright \tau$ is F together with some tautological clauses in the variables x_1, \dots, x_n .*

Proof. Define τ as follows. If $\gamma_F(C(m, \ell, b)) = 0$, then set $\tau(T(m, \ell, b)) = 0$. This deletes $T(m, \ell, b)$ from (1) and satisfies the corresponding clauses of (4) together with either (2) (if $b = 1$) or (3) (if $b = 0$). If $\gamma_F(C(m, \ell, b)) = 1$, then the corresponding clause in (4) has been satisfied and we define $\tau(T(m, \ell, b)) = x_\ell^b$ and $\tau(T(\ell)) = x_\ell$. This choice turns (2) (if $b = 1$) or (3) (if $b = 0$) into a tautological clause and correctly substitutes the remaining literals of (1) to yield the clause C_m of F . ◀

C Formula $\text{R}^k\text{REF}_{s,t}^F$

The formula $\text{R}^k\text{REF}_{s,t}^F$ is the union of the following sets of clauses:

$$\bigvee_{u \in [k]} \neg S_u(1, j) \vee \neg I(j, m) \vee D(1, j, \ell, b) \quad j \in [t], m \in [r], b \in \{0, 1\}, x_\ell^b \in C_m, \quad (26)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg D(i, j, \ell, 1) \vee \neg D(i, j, \ell, 0) \quad i \in [s], j \in [t], \ell \in [n], \quad (27)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg L(i, j, j') \vee \neg V(i, j, \ell) \vee D(i-1, j', \ell, 1) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell \in [n], \quad (28)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg R(i, j, j') \vee \neg V(i, j, \ell) \vee D(i-1, j', \ell, 0) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell \in [n], \quad (29)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg L(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell', b) \vee D(i, j, \ell', b) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell, \ell' \in [n], b \in \{0, 1\}, (\ell', b) \neq (\ell, 1), \quad (30)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg R(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell', b) \vee D(i, j, \ell', b) \\ i \in [s] \setminus \{1\}, j, j' \in [t], \ell, \ell' \in [n], b \in \{0, 1\}, (\ell', b) \neq (\ell, 0), \quad (31)$$

$$\bigvee_{u \in [k]} \neg S_u(s, t) \vee \neg D(s, t, \ell, b) \quad \ell \in [n], b \in \{0, 1\}, \quad (32)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \bigvee_{\ell \in [n]} V(i, j, \ell) \quad i \in [s] \setminus \{1\}, j \in [t], \quad (33)$$

$$\bigvee_{u \in [k]} \neg S_u(1, j) \vee \bigvee_{m \in [r]} I(j, m) \quad j \in [t], \quad (34)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \bigvee_{j' \in [t]} L(i, j, j') \quad i \in [s] \setminus \{1\}, j \in [t], \quad (35)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \bigvee_{j' \in [t]} R(i, j, j') \quad i \in [s] \setminus \{1\}, j \in [t], \quad (36)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg V(i, j, \ell) \vee \neg V(i, j, \ell') \quad i \in [s] \setminus \{1\}, j \in [t], \ell, \ell' \in [n], \ell \neq \ell', \quad (37)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg I(j, m) \vee \neg I(j, m') \quad j \in [t], m, m' \in [r], m \neq m', \quad (38)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg L(i, j, j') \vee \neg L(i, j, j'') \quad i \in [s] \setminus \{1\}, j, j', j'' \in [t], j' \neq j'', \quad (39)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg R(i, j, j') \vee \neg R(i, j, j'') \quad i \in [s] \setminus \{1\}, j, j', j'' \in [t], j' \neq j'', \quad (40)$$

$$S_u(s, t) \quad u \in [k], \quad (41)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg L(i, j, j') \vee S_{u'}(i-1, j') \quad i \in [s] \setminus \{1\}, j, j' \in [t], u' \in [k], \quad (42)$$

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg R(i, j, j') \vee S_{u'}(i-1, j') \quad i \in [s] \setminus \{1\}, j, j' \in [t], u' \in [k]. \quad (43)$$

Clauses in (26) - (40) are just the clauses in (11) - (25) with the additional disjuncts $\bigvee_{u \in [k]} \neg S_u(i, j)$ with the corresponding (i, j) . Clauses (41) together with (32) make sure that $C_{s,t}$ is selected and empty. Clauses in (42) and (43) ensure that if $C_{i-1,j'}$ is not selected then it cannot be used as a premise.

D Proof of Theorem 6

By induction on $i \in [s]$, we derive for each $j \in [t]$ the clause

$$D_{i,j} := \bigvee_{u \in [k]} \neg S(i, j) \vee \bigvee_{\ell \in [n], b \in \{0,1\}} (D(i, j, \ell, b) \wedge T(\ell)^b). \quad (44)$$

The meaning of this clause is that if $C_{i,j}$ (the clause described by $D(i, j, \cdot, \cdot)$) is selected then it contains a satisfied literal. Then, cutting $D_{s,t}$ with (32) for each $\ell \in [n]$ and $b \in \{0, 1\}$, followed by k cuts with clauses (41), yields the empty clause.

Base case: $i = 1$. For each $j \in [t], m \in [r], \ell \in [n], b \in \{0, 1\}$, cut (4) with (6) to obtain $\bigvee_{u \in [k]} \neg S_u(1, j) \vee \neg I(j, m) \vee \neg T(m, \ell, b) \vee D(1, j, \ell, b)$. Applying \wedge -introduction to this and $\neg T(m, \ell, b) \vee T(\ell)^b$ (which is either (2) or (3)) yields

$$\bigvee_{u \in [k]} \neg S_u(1, j) \vee \neg I(j, m) \vee \neg T(m, \ell, b) \vee (D(1, j, \ell, b) \wedge T(\ell)^b). \quad (45)$$

Cutting (45) for each $\ell \in [n]$ and $b \in \{0, 1\}$ with (1) gives $\neg I(j, m) \vee D_{1,j}$. Cutting this for each $m \in [r]$ with (34) yields $D_{1,j}$.

Induction step: Assume we have derived $D_{i-1,j'}$ for all $j' \in [t]$. We derive $D_{i,j}$ for each $j \in [t]$. In the following we write P_0 in place of R and P_1 in place of L .

For each $\ell \in [n], b \in \{0, 1\}, j' \in [t]$, cut $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg D(i-1, j', \ell, 1) \vee \neg D(i-1, j', \ell, 0)$ (from (27)) with $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee D(i-1, j', \ell, 1-b)$ (which is from (28) or (29)) to obtain $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell, b)$. Cutting this with $D_{i-1,j'}$ yields

$$\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee (D_{i-1,j'} \setminus \{D(i-1, j', \ell, b) \wedge T(\ell)^b\}). \quad (46)$$

Cut (46) with $T(\ell) \vee \neg T(\ell)$ to get

$$\begin{aligned} & \bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee T(\ell)^{1-b} \\ & \vee (D_{i-1,j'} \setminus \{D(i-1, j', \ell, 0) \wedge \neg T(\ell), D(i-1, j', \ell, 1) \wedge T(\ell)\}). \end{aligned} \quad (47)$$

Next, for each $\ell' \in [n] \setminus \{\ell\}$ and $b' \in \{0, 1\}$, apply \wedge -introduction to $T(\ell') \vee \neg T(\ell')$ and $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee \neg D(i-1, j', \ell', b') \vee D(i, j, \ell', b')$ (from (30) or (31)) to get

$$\begin{aligned} & \bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee (D(i, j, \ell', b') \wedge T(\ell')^{b'}) \\ & \vee \neg D(i-1, j', \ell', b') \vee T(\ell')^{1-b'}. \end{aligned} \quad (48)$$

Cutting (48), for each $\ell' \in [n] \setminus \{\ell\}$ and $b' \in \{0, 1\}$, with (47) results, after an additional weakening by $(D(i, j, \ell, 0) \wedge T(\ell)^0) \vee (D(i, j, \ell, 1) \wedge T(\ell)^1)$, in

$$\bigvee_{u \in [k]} \neg S_u(i-1, j') \vee \neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee T(\ell)^{1-b} \vee D_{i,j}. \quad (49)$$

Cut (49), for each $u' \in [k]$, with $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \neg P_{1-b}(i, j, j') \vee S_{u'}(i-1, j')$ (from (42) or (43)) to get

$$\neg P_{1-b}(i, j, j') \vee \neg V(i, j, \ell) \vee T(\ell)^{1-b} \vee D_{i,j}. \quad (50)$$

Recall that we have obtained (50) for each $\ell \in [n], b \in \{0, 1\}, j' \in [t]$. Cutting (50), for each $j' \in [t]$, with $\bigvee_{u \in [k]} \neg S_u(i, j) \vee \bigvee_{j' \in [t]} P_{1-b}(i, j, j')$ (which is from (35) or (36)) yields $\neg V(i, j, \ell) \vee T(\ell)^{1-b} \vee D_{i,j}$. We have derived such a clause for each $\ell \in [n], b \in \{0, 1\}$, so a cut on $T(\ell)$ gives $\neg V(i, j, \ell) \vee D_{i,j}$, and cutting this, for each $\ell \in [n]$, with (33) yields $D_{i,j}$.

Let us estimate the size of the refutation. The size of the base case is $O(trn^2 + r^2 + rnk)$, the total size of the induction steps is $O(st(n^3t + n^2tk + nt^2) + nt^2)$, and the size of the finish is $O(n^2 + nk)$. In total this is $O(trn^2 + tr^2 + trnk + st^2n^3 + st^2n^2k + st^2nk^2 + st^3n)$.

E Some results on trees and a proof of Theorem 11

► **Definition 20.** A partial assignment π to the variables of $\text{REF}_{s,t}^F$ is called *respectful* if for each $(i, j) \in [s] \times [t]$, either (i, j) is not mentioned in π , or $i \in [s] \setminus \{1\}$ and each of $D(i, j, \cdot, \cdot)$, $V(i, j, \cdot)$, $R(i, j, \cdot)$, $L(i, j, \cdot)$ is set by π , or $i = 1$ and both $D(1, j, \cdot, \cdot)$ and $I(j, \cdot)$ are set by π . In other words, respectful assignments are exactly the assignments of the form π_v where v is a node of a decision tree over $\text{REF}_{s,t}^F$.

If T is a decision tree over $\text{REF}_{s,t}^F$ and π is a respectful partial assignment, $T \upharpoonright \pi$ is obtained as follows: for each node v of T with a label (i, j) that is mentioned in π , contract the edge whose label determines an assignment to the variables with home pair (i, j) that is a subset of π , and delete all other edges leaving v (and delete their associated subtrees).

► **Lemma 21.** Let T be a decision tree over $\text{REF}_{s,t}^F$, let G be a DNF, and let π be a respectful partial assignment. If T strongly represents G , then $T \upharpoonright \pi$ strongly represents $G \upharpoonright \pi$.

Proof. For a leaf v of $T \upharpoonright \pi$ there is a unique leaf u of T such that $\pi_v = \pi_u \setminus \pi$, where π_u , π_v are defined as in Definition 7. Moreover, v has the same label as u , and π and π_u are compatible. Therefore, for a term $q \in G$ we have $q \upharpoonright (\pi \cup \pi_u) = q \upharpoonright (\pi \cup \pi_v) = (q \upharpoonright \pi) \upharpoonright \pi_v$. Also, for $b \in \{0, 1\}$, if $q \upharpoonright \pi_u = b$ then $q \upharpoonright (\pi \cup \pi_u) = b$. ◀

In the other direction, we have the following lemma.

► **Lemma 22.** Let T be a decision tree over $\text{REF}_{s,t}^F$, and let G be a DNF in the variables of $\text{REF}_{s,t}^F$. For each leaf v of T , let T_v be a decision tree that strongly represents $G \upharpoonright \pi_v$, where π_v is the path in T from the root to v . Moreover, assume that each label (i, j) of an internal node of T_v is a home pair of a variable of $G \upharpoonright \pi_v$. Then the tree T' obtained by appending to each leaf v of T the tree T_v strongly represents G .

Proof. This follows directly from the definitions. ◀

Proof of Theorem 11. Denote Π the $\text{Res}(k)$ refutation. For a line G in Π , let T_G be a decision tree over $\text{REF}_{s,t}^F$ of minimum height that strongly represents G . We can assume that no node of T_G is labelled with a pair (i, j) that is not a home pair of any variable of G .

For any respectful partial assignment π let C_π be the clause consisting of the following literals: $D(i, j, \ell, b)$ if and only if $\pi(D(i, j, \ell, b)) = 0$, $\neg D(i, j, \ell, b)$ if and only if $\pi(D(i, j, \ell, b)) = 1$, $\neg I(j, m)$ if and only if π sets $I(j, \cdot)$ to m , $\neg V(i, j, \ell)$ if and only if π sets $V(i, j, \cdot)$ to ℓ , $\neg L(i, j, j')$ if and only if π sets $L(i, j, \cdot)$ to j' , $\neg R(i, j, j')$ if and only if π sets $R(i, j, \cdot)$ to j' .

By induction on the lines of Π we show that for each line G of Π and for each $\pi \in \text{Br}_0(T_G)$, there is a resolution derivation $\Pi_G(\pi)$ of C_π from H together with the clauses (18) - (25), such that the index-width of $\Pi_G(\pi)$ is at most $3h$. The theorem then follows from $\{C_\pi : \pi \in \text{Br}_0(T_\emptyset)\} = \{C_\emptyset\} = \{\emptyset\}$.

Assume that G is an axiom $X \vee \neg X$. Then all the branches of T_G are labelled with 1, and so $\{C_\pi : \pi \in \text{Br}_0(T_G)\} = \emptyset$.

33:16 Failure of Feasible Disjunction Property and Automatability in k -DNF Resolution

Next assume that $G \in H$. Let $\pi \in \text{Br}_0(T_G)$. Since G is a clause, the node labels of T_G are exactly the pairs (i, j) mentioned in G . Note that since $G \upharpoonright \pi = 0$, for every (i, j) each literal of a variable in $D(i, j, \cdot, \cdot)$ that is in G is also in C_π . Suppose that π sets $V(i, j, \cdot)$ to $\ell \in [n]$. If there is a literal in G of a variable from $V(i, j, \cdot)$ such that the literal is not in C_π , then the literal must be $V(i, j, \ell')$ for some $\ell' \in [n]$ with $\ell' \neq \ell$. This follows from $G \upharpoonright \pi = 0$ and $\neg V(i, j, \ell) \in C_\pi$. Such literals $V(i, j, \ell')$ can be removed from G by resolving with the clause $\neg V(i, j, \ell) \vee \neg V(i, j, \ell')$ from (22). Similarly, we remove from G the literals in $G \setminus C_\pi$ of I, L, R -variables by resolving with the corresponding clauses from (23), (24), (25), respectively. We have thus obtained a resolution derivation $\Pi_G(\pi)$ of C_π from $\{G\}$ together with the clauses (22) - (25). Because the index-width of G is at most h , the same is true for the clauses in $\Pi_G(\pi)$.

Now assume that line G in Π is inferred from previously derived lines G_1, \dots, G_d for $d \in [2]$. By the induction hypothesis, we have for each $c \in [d]$ and for each $\pi \in \text{Br}_0(T_{G_c})$ a resolution derivation $\Pi_G(\pi)$ of C_π with the required properties. First construct a decision tree T as follows: if $d = 1$, T is T_{G_1} ; if $d = 2$, append to each branch $\pi \in \text{Br}_1(T_{G_1})$ the tree $T_{G_2} \upharpoonright \pi$. Observe that for each $\pi \in \text{Br}_0(T)$ there is $c \in [d]$ and $\pi' \in \text{Br}_0(T_c)$ such that $\pi' \subseteq \pi$, and C_π is a weakening of $C_{\pi'}$. Also, the index-width of C_π is at most $2h$, because so is the height of T . For a node v of T define a partial assignment π_v as in Definition 7.

Let $\sigma \in \text{Br}_0(T_G)$ be given. Inductively, from the leaves to the root of T , we show that if a node v of T is such that π_v is compatible with σ , then there is a resolution derivation $\Pi_G(\pi_v, \sigma)$ of $C_{\pi_v} \vee C_\sigma$ from H together with the clauses (18) - (25), such that the index-width of $\Pi_G(\pi_v, \sigma)$ is at most $3h$. When we reach the root of T , we will have obtained a derivation $\Pi_G(\emptyset, \sigma)$ of C_σ , and this is the derivation $\Pi_G(\sigma)$ we are after.

Assume that v is a leaf of T and π_v is compatible with σ . Then $\pi_v \in \text{Br}_0(T)$. This can be seen as follows. It is easy to check that the rules of $\text{Res}(k)$ have the property, called strong soundness, that any partial assignment that satisfies all premises of a rule also satisfies the conclusion of the rule. If we had $\pi_v \in \text{Br}_1(T)$, then for each $c \in [d]$, π_v contains some $\pi_c \in \text{Br}_1(T_{G_c})$, and so $G_c \upharpoonright \pi_v = G_c \upharpoonright \pi_c = 1$ because T_{G_c} strongly represents G_c . By strong soundness it follows that $G \upharpoonright \pi_v = 1$. But this means that π_v cannot be compatible with σ , because σ falsifies every term of G . So indeed $\pi_v \in \text{Br}_0(T)$. Further, we have that $C_{\pi_v} \vee C_\sigma$ is a weakening of C_{π_v} , which in turn is a weakening of $C_{\pi'}$ for some $\pi' \in \text{Br}_0(T_c)$ and some $c \in [d]$ such that $\pi' \subseteq \pi_v$, by the construction of T . By the inductive hypothesis we have a resolution derivation $\Pi_G(\pi')$ of $C_{\pi'}$ with the required properties. Because the index-width of C_{π_v} is at most $2h$, the index-width of $C_{\pi_v} \vee C_\sigma$ is at most $3h$. We have thus obtained a resolution derivation $\Pi_G(\pi_v, \sigma)$ of $C_{\pi_v} \vee C_\sigma$ with the required properties.

Now assume that v is labelled with a pair (i, j) and π_v is compatible with σ . We distinguish two cases. In the first case, assume that (i, j) is mentioned in σ . Then there is a child u of v such that $\pi_u \setminus \pi_v \subseteq \sigma$. Also, π_u is compatible with σ . By the induction hypothesis we therefore have a resolution derivation $\Pi_G(\pi_u, \sigma)$ of $C_{\pi_u} \vee C_\sigma$ with the required properties. Because $\pi_u \cup \sigma = \pi_v \cup \sigma$, we have $C_{\pi_u} \vee C_\sigma = C_{\pi_v} \vee C_\sigma$, and so we define $\Pi_G(\pi_v, \sigma)$ to be $\Pi_G(\pi_u, \sigma)$. In the second case, assume that (i, j) is not mentioned in σ . Then for each child u of v , π_u is compatible with σ . By the induction hypothesis, for each such u there is a resolution derivation $\Pi_G(\pi_u, \sigma)$ of $C_{\pi_u} \vee C_\sigma$ with the required properties. Notice that $C_{\pi_u} \vee C_\sigma = C_{\pi_u \setminus \pi_v} \vee C_{\pi_v} \vee C_\sigma$. We first construct a resolution refutation Π' of $\{C_{\pi_u \setminus \pi_v} : u \text{ is a child of } v\}$ together with the clauses (18) - (21) such that the index-width of Π' is 1. This is easy: since $\{C_{\pi_u \setminus \pi_v} : u \text{ is a child of } v\} = \{C_\alpha : \alpha \text{ is respectful and mentions just the pair } (i, j)\}$, we use (18), (20), (21) (resp. (19) if $i = 1$) to remove all the negated V, L, R -variables (resp. the negated I -variables) from the clauses C_α , and we refute the resulting clauses by a

refutation in the form of a complete binary tree to resolve all the D -variables. Now, having Π' , we define $\Pi_G(\pi_v, \sigma)$ as follows: add the literals of $C_{\pi_v} \vee C_\sigma$ to each clause of Π' other than an initial clause from (18), (20), (21), (19), and derive each initial clause $C_{\pi_u} \vee C_\sigma$ in the resulting derivation using the derivation $\Pi_G(\pi_u, \sigma)$. It is easy to see that $\Pi_G(\pi_v, \sigma)$ has the required properties. \blacktriangleleft

F Proof of Theorem 16

Denote the right hand side of the inequality (8) by $p_a(w)$. Let $k \geq 1$ be given and denote $\rho := \rho_k$. We prove the theorem by induction on a .

Base case: $a = 1$. G is a clause. If $c_i(G) \leq w$, then $\Pr[h_i(G \upharpoonright \rho) > w] = 0$ because we can build a decision tree strongly representing $G \upharpoonright \rho$ by querying the pairs from the smallest index-cover of G . If $c_i(G) > w$, we have $\Pr[h_i(G \upharpoonright \rho) > w] \leq \Pr[G \upharpoonright \rho \neq 1] \leq (1 - (1 - 2^{-k})/2)^{c_i(G)} \leq (1 - 1/4)^{c_i(G)} \leq e^{-c_i(G)/4} = 2^{-c_i(G)\gamma(1)} \leq 2^{-w\gamma(1)}$.

Induction step: Assume the theorem holds for $a - 1$, witnessed by $\delta(k, a - 1)$ and $n_0(k, a - 1)$. Find a positive $\delta(k, a) \leq \delta(k, a - 1)$ and an integer $n_0(k, a) \geq n_0(k, a - 1)$ such that

$$-\frac{\gamma(a-1)}{2}n + \left(2 \log t + \log n + \frac{\gamma(a-1)}{n^{a-2}}\right) \cdot \frac{\gamma(a-1)}{4} \leq -\gamma(a) \quad (51)$$

holds for any n, r, t satisfying (7) with $\delta(k, a)$ and $n_0(k, a)$ in place of δ and n_0 , respectively. Let G be an a -DNF, and let U be an index cover of G of size $c_i(G)$. We distinguish two cases based on $c_i(G)$.

Case 1: $c_i(G) > \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4}$. In this case we want to show that ρ satisfies G with a high probability. To this end, note that there are at least $c_i(G)/a$ many terms in G that are index-independent, that is, for no two of them there is a pair $(i, j) \in [s] \times [t]$ mentioned by both. (If every such set of terms was smaller than $c_i(G)/a$, take a maximal one and observe that the set of pairs mentioned by the terms forms an index-cover of G of cardinality smaller than $c_i(G)$, a contradiction.) It is easy to see that each of these index-independent terms is satisfied by ρ with independent probability at least 2^{-2a} . Therefore,

$$\begin{aligned} \Pr[h_i(G \upharpoonright \rho) > w] &\leq \Pr[G \upharpoonright \rho \neq 1] \leq (1 - 2^{-2a})^{c_i(G)/a} \leq 2^{-\frac{(\log e)}{a2^{2a}} c_i(G)} \leq 2^{-\frac{(\log e)}{a2^{2a}} \cdot \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4}} \\ &= 2^{-\frac{w}{n^{a-1}} \gamma(a)}. \end{aligned}$$

This finishes the inductive step for Case 1.

Case 2: $c_i(G) \leq \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4}$. Let $U' \subseteq U$, and let $\nu : \text{Var}_S(U) \cup \text{Var}(U \setminus U') \rightarrow \{0, 1\}$ satisfy the following conditions:

($\nu 1$) for each $(i, j) \in U'$ and each $u \in [k]$, $\nu(S_u(i, j)) = 1$,

($\nu 2$) for each $(i, j) \in U \setminus U'$ there is $u \in [k]$ with $\nu(S_u(i, j)) = 0$.

We have

$$\begin{aligned} &\Pr[h_i(G \upharpoonright \rho) > w \mid \rho \upharpoonright \text{dom}(\nu) = \nu] \\ &\leq \Pr[\exists \pi : \text{Var}(U') \rightarrow \{0, 1\}, \pi \text{ is respectful} \wedge h_i((G \upharpoonright \pi) \upharpoonright \rho) > w - |U'| \mid \rho \upharpoonright \text{dom}(\nu) = \nu] \\ &\leq \sum_{\substack{\pi : \text{Var}(U') \rightarrow \{0, 1\}, \\ \pi \text{ is respectful}}} \Pr[h_i((G \upharpoonright \pi) \upharpoonright \rho) > w - |U'| \mid \rho \upharpoonright \text{dom}(\nu) = \nu] \\ &= \sum_{\substack{\pi : \text{Var}(U') \rightarrow \{0, 1\}, \\ \pi \text{ is respectful}}} \Pr[h_i(((G \upharpoonright \pi) \upharpoonright \nu) \upharpoonright \rho) > w - |U'|] \\ &\leq (t^2 n 2^{2n})^{|U'|} p_{a-1}(w - |U'|). \end{aligned}$$

Here the first inequality follows from Lemma 22 and from $(G \upharpoonright \pi) \upharpoonright \rho = (G \upharpoonright \rho) \upharpoonright \pi$ (since $\text{dom}(\pi) \cap \text{dom}(\rho) = \emptyset$). The second inequality is obtained by the union bound. The equality follows since the events $h_i(((G \upharpoonright \pi) \upharpoonright \nu) \upharpoonright \rho) > w - |U'|$ and $\rho \upharpoonright \text{dom}(\nu) = \nu$ are independent (by the definition of ρ). And the last inequality is by the induction hypothesis and by the upper bound $t^2 n 2^{2n} = \max\{t^2 n 2^{2n}, r 2^{2n}\}$ (recall that $t \geq r$) over $(i, j) \in [s] \times [t]$ on the number of respectful partial assignments mentioning exactly the pair (i, j) .

Since the event $A \cap U = U'$ (where the random variable A is given by Definition 15) is the disjoint union of events $\rho \upharpoonright \text{dom}(\nu) = \nu$ over all ν satisfying conditions $(\nu 1)$ and $(\nu 2)$, the above calculation implies

$$\Pr[h_i(G \upharpoonright \rho) > w \mid A \cap U = U'] \leq (t^2 n 2^{2n})^{|U'|} p_{a-1}(w - |U'|). \quad (52)$$

Therefore,

$$\begin{aligned} \Pr[h_i(G \upharpoonright \rho) > w] &= \sum_{U' \subseteq U} \Pr[h_i(G \upharpoonright \rho) > w \wedge A \cap U = U'] \\ &= \sum_{U' \subseteq U} \Pr[h_i(G \upharpoonright \rho) > w \mid A \cap U = U'] \cdot \Pr[A \cap U = U'] \\ &\leq \sum_{U' \subseteq U} (t^2 n 2^{2n})^{|U'|} p_{a-1}(w - |U'|) \cdot 2^{-k|U'|} (1 - 2^{-k})^{|U \setminus U'|} \\ &= \sum_{q=0}^{c_i(G)} \binom{c_i(G)}{q} (t^2 n 2^{2n})^q p_{a-1}(w - q) \cdot 2^{-kq} (1 - 2^{-k})^{c_i(G) - q} \\ &\leq (t^2 n 2^{2n})^{c_i(G)} p_{a-1}(w - c_i(G)). \end{aligned} \quad (53)$$

Here the first inequality is by (52) and by the definition of ρ . The second inequality follows from $(t^2 n 2^{2n})^q p_{a-1}(w - q) \leq (t^2 n 2^{2n})^{c_i(G)} p_{a-1}(w - c_i(G))$ for $q \leq c_i(G)$. From (53), using the definition of $p_{a-1}(w - c_i(G))$ and the assumption $c_i(G) \leq \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4}$, we get

$$\begin{aligned} \log(\Pr[h_i(G \upharpoonright \rho) > w]) &\leq (2 \log t + \log n + 2n) c_i(G) - \frac{w - c_i(G)}{n^{a-2}} \gamma(a-1) \\ &= \left(2 \log t + \log n + 2n + \frac{\gamma(a-1)}{n^{a-2}}\right) c_i(G) - \frac{w \gamma(a-1)}{n^{a-2}} \\ &\leq \left(2 \log t + \log n + 2n + \frac{\gamma(a-1)}{n^{a-2}}\right) \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4} - \frac{w \gamma(a-1)}{n^{a-2}} \\ &= -\frac{w \gamma(a-1)}{2n^{a-2}} + \left(2 \log t + \log n + \frac{\gamma(a-1)}{n^{a-2}}\right) \frac{w}{n^{a-1}} \cdot \frac{\gamma(a-1)}{4} \\ &\leq -\frac{w}{n^{a-1}} \gamma(a), \end{aligned}$$

where the last inequality is equivalent to (51). This finishes the inductive step for Case 2, and the proof of the theorem.

G The width lower bound

► **Definition 23.** A partial assignment σ to the variables of $\text{REF}_{s,t}^F$ is called admissible if it satisfies all the following conditions.

(A1) For each $(i, j) \in [s] \times [t]$, $D(i, j, \cdot, \cdot)$ (resp. $V(i, j, \cdot)$, $I(j, \cdot)$, $L(i, j, \cdot)$, $R(i, j, \cdot)$) either is set to some clause (resp. some $\ell \in [n]$, some $m \in [r]$, some $j' \in [t]$, some $j' \in [t]$) by σ or contains no variable that is in $\text{dom}(\sigma)$.

- (A2) For each $(i, j) \in [s] \times [t]$, if $L(i, j, \cdot)$ or $R(i, j, \cdot)$ is set to some $j' \in [t]$, then both $D(i, j, \cdot, \cdot)$ and $D(i-1, j', \cdot, \cdot)$ are set.
- (A3) For each $(i, j) \in ([s] \setminus \{1\}) \times [t]$, $D(i, j, \cdot, \cdot)$ is set if and only if $V(i, j, \cdot)$ is set. For each $j \in [t]$, $D(1, j, \cdot, \cdot)$ is set if and only if $I(j, \cdot)$ is set.
- (A4) For each $(i, j) \in [s] \times [t]$, if $D(i, j, \cdot, \cdot)$ is set to a clause $C_{i,j}$, then $C_{i,j}$ is non-tautological and has at least $\min\{s-i, n\}$ many literals. If $D(i, j, \cdot, \cdot)$ is set to a clause $C_{i,j}$ with less than n literals and $V(i, j, \cdot)$ is set to some $\ell \in [n]$, then none of the literals of x_ℓ is in $C_{i,j}$.
- (A5) If $D(s, t, \cdot, \cdot)$ is set, it is set to the empty clause.
- (A6) For each $j \in [t]$, if $I(j, \cdot)$ is set, then σ satisfies all clauses in (11) with this j .
- (A7) For each $(i, j) \in ([s] \setminus \{1\}) \times [t]$, if $L(i, j, \cdot)$ (resp. $R(i, j, \cdot)$) is set, then σ satisfies all clauses in (13) and (15) (resp. (14) and (16)) with this (i, j) (i.e., those clauses that contain the literal $\neg L(i, j, j')$ (resp. $\neg R(i, j, j')$) for some $j' \in [t]$).

Proof of Theorem 17. Assume for a contradiction that there is a resolution refutation Π of $\text{REF}_{s,t}^F$ of index-width at most w . We will show that if there is an admissible partial assignment falsifying a clause E in Π obtained by the resolution rule from E_0 and E_1 , then there is an admissible partial assignment falsifying either E_0 or E_1 . This immediately (by induction) leads to a contradiction, since the empty assignment is admissible and falsifies the last (empty) clause in Π , and, by definition, no admissible partial assignment falsifies any clause of $\text{REF}_{s,t}^F$.

Let then σ be an admissible partial assignment falsifying a clause E in Π . Without loss of generality, assume that σ is a minimal (with respect to inclusion) admissible partial assignment with this property.

Let Q be the variable resolved on to obtain E from E_0 and E_1 . If $Q \in \text{dom}(\sigma)$, then σ already falsifies either E_0 or E_1 . So assume that $Q \notin \text{dom}(\sigma)$. We consider two cases.

Case 1. Suppose that for some $(i, j) \in [s] \times [t]$, $Q \in D(i, j, \cdot, \cdot)$ or $Q \in V(i, j, \cdot)$ (resp. $Q \in I(j, \cdot)$ and $i = 1$). Note that by (A1), (A2), and (A3), no variable from $D(i, j, \cdot, \cdot) \cup V(i, j, \cdot) \cup L(i, j, \cdot) \cup R(i, j, \cdot)$ (resp. $D(1, j, \cdot, \cdot) \cup I(j, \cdot)$) is in $\text{dom}(\sigma)$, and, moreover, for any $j' \in [t]$, it is not the case that $L(i+1, j', \cdot)$ or $R(i+1, j', \cdot)$ is set to j by σ . Therefore, we can extend σ to a partial assignment σ' as follows. Set $D(i, j, \cdot, \cdot)$ to any non-tautological clause containing n literals, unless $(i, j) = (s, t)$, in which case set $D(i, j, \cdot, \cdot)$ to the empty clause. In case $i \geq 2$, set $V(i, j, \cdot)$ to an arbitrary value $\ell \in [n]$; in case $i = 1$, set $I(j, \cdot)$ to any $m \in [r]$ such that the clause C_m is a subset of the clause to which we have set $D(1, j, \cdot, \cdot)$. (Here we use that F is unsatisfiable.) It is straightforward to check that σ' is admissible. Since $Q \in \text{dom}(\sigma')$, σ' falsifies $E \cup \{Q^{1-\sigma'(Q)}\}$, of which either E_0 or E_1 is a subset.

Case 2. Suppose that for some $(i, j) \in ([s] \setminus \{1\}) \times [t]$, $Q \in L(i, j, \cdot)$ (if $Q \in R(i, j, \cdot)$, we proceed in a completely analogous way). We may assume that $D(i, j, \cdot, \cdot)$ is set to some clause $C_{i,j}$ by σ and $V(i, j, \cdot)$ is set to some $\ell \in [n]$ by σ ; if not, set them both as described in Case 1. We now concentrate on the level $i-1$. Since the index-width of E is at most w and σ is a minimal admissible partial assignment falsifying E ,

$$|\{j' : D(i-1, j', \cdot, \cdot) \text{ is set by } \sigma\}| \leq 2w. \quad (54)$$

This is because $D(i-1, j', \cdot, \cdot)$ can be set by σ for two reasons: either $(i-1, j')$ is mentioned in E (which, together with (A2) and (A3), implies that $D(i-1, j', \cdot, \cdot)$ is set by σ) or there is some $j'' \in [t]$ such that a literal of a variable from $L(i, j'', \cdot)$ or $R(i, j'', \cdot)$ is in E (which forces σ to set $L(i, j'', \cdot)$ or $R(i, j'', \cdot)$, respectively, in order to falsify the literal) and σ happens to set $L(i, j'', \cdot)$ or $R(i, j'', \cdot)$, respectively, to j' (and therefore by (A2) $D(i-1, j', \cdot, \cdot)$ must be set by σ too).

We extend σ to a partial assignment σ' as follows. Set $L(i, j, \cdot)$ to any j' that is not from the set in (54). Such j' exists because $2w < t$. Set $D(i-1, j', \cdot, \cdot)$ to the clause $C_{i-1, j'} := (C_{i, j} \setminus \{\neg x_\ell\}) \cup \{x_\ell\}$, where $C_{i, j}$ and ℓ are as above. Finally, if $i \in \{3, \dots, s\}$, then either $C_{i-1, j'}$ has less than n literals and we set $V(i-1, j', \cdot)$ to any $\ell' \in [n]$ such that no literal of $x_{\ell'}$ is in $C_{i-1, j'}$, or $C_{i-1, j'}$ has n literals, in which case we set $V(i-1, j', \cdot)$ arbitrarily. If $i = 2$, then by (A4), (9), and the definition of $C_{i-1, j'}$ we know that $C_{i-1, j'}$ has n literals, and we set $I(j', \cdot)$ to any $m \in [r]$ such that $C_m \subseteq C_{i-1, j'}$. (Here we use that F is unsatisfiable.) This finishes the definition of σ' .

It is again easy to verify that σ' is admissible. Because $Q \in \text{dom}(\sigma')$, σ' falsifies $E \cup \{Q^{1-\sigma'(Q)}\}$, of which one of E_0, E_1 is a subset. \blacktriangleleft

H Proof of Theorem 18

Let $k \geq 1$ be given. Take δ and n_0 as given by Theorem 16 for $a = k$. If necessary, increase n_0 so that it satisfies

$$\beta(k)n_0 > k + 1. \quad (55)$$

Let n, r, s, t be integers satisfying (10), and let F satisfy the hypothesis of the theorem. Assume for a contradiction that there is a $\text{Res}(k)$ refutation Π of $\text{R}^k\text{REF}_{s,t}^F$ of length at most $2^{\beta(k)\frac{t}{n^{k-1}}}$.

Recall the random variable A from Definition 15. We have that with probability 2^{-k} ,

(a) $(s, t) \in A$.

By the Chernoff bound and the union bound, with probability at least $1 - se^{-t2^{-k}/8}$,

(b) for each $i \in [s]$ the cardinality of $A \cap (\{i\} \times [t])$ is at least $t/2^{k+1}$.

We have

$$se^{-t2^{-k}/8} = 2^{\log s - \frac{t \log e}{2^{k+3}}} \leq 2^{\log t - \frac{t \log e}{2^{k+3}}} \leq 2^{\log n_0 - \frac{n_0 \log e}{2^{k+3}}} < 2^{-(k+1)},$$

where we used $s \leq t$, $n_0 \leq s$ (from (10)), and $\frac{n_0 \log e}{2^{k+3}} - \log n_0 > \beta(k)n_0 > k + 1$ (by (55)).

By Theorem 16 and the union bound, with probability at least $1 - |\Pi| \cdot 2^{-\frac{t}{n^{k-1}2^{k+5}}\gamma(k)}$,

(c) for every line G in Π , $h_i(G \upharpoonright \rho_k) \leq t/2^{k+5}$.

We have

$$|\Pi| \cdot 2^{-\frac{t}{n^{k-1}2^{k+5}}\gamma(k)} \leq 2^{\beta(k)\frac{t}{n^{k-1}}} \cdot 2^{-\frac{t}{n^{k-1}2^{k+5}}\gamma(k)} = 2^{-\beta(k)\frac{t}{n^{k-1}}} \leq 2^{-\beta(k)n_0} < 2^{-(k+1)},$$

where we used $n^k \leq t$, $n_0 \leq n$ (from (10)), and (55).

It follows that there exists ρ_k such that (a), (b) and (c) hold. Fix any such ρ_k and denote it by ρ . We now restrict $\text{R}^k\text{REF}_{s,t}^F \upharpoonright \rho$ some more before we apply Theorem 11.

For each level $i \in [s]$ select any $t' := \lfloor t/2^{k+1} \rfloor - 2$ home pairs (i, j) of variables of $\text{R}^k\text{REF}_{s,t}^F \upharpoonright \rho$ (they exist thanks to (b)), making sure to include the pair (s, t) in the selection. Denote the set of selected pairs by B . Define a partial assignment $\nu : \text{Var}(\text{R}^k\text{REF}_{s,t}^F \upharpoonright \rho) \rightarrow \{0, 1\}$ by mapping all the variables with not selected home pairs so that they form an arbitrary resolution derivation from F , that is, so that ν satisfies every clause of $\text{R}^k\text{REF}_{s,t}^F \upharpoonright \rho$ that contains a literal of a variable in $\text{dom}(\nu)$. (This derivation may require two clauses per level, which is why we selected only $\lfloor t/2^{k+1} \rfloor - 2$ on each level.) Note that ν is respectful. Hence by (c) and Lemma 21 we have that for any line G in $\Pi \upharpoonright \rho$, $h_i(G \upharpoonright \nu) \leq t/2^{k+5}$.

Next, define a partial assignment λ as follows. For every $(i, j) \in B \setminus (\{1\} \times [t])$ and every $j' \in [t]$ such that $(i-1, j') \notin B$, map both $L(i, j, j')$ and $R(i, j, j')$ to 0. Let us verify that $((\text{R}^k\text{REF}_{s,t}^F \upharpoonright \rho) \upharpoonright \nu) \upharpoonright \lambda$ is $\text{REF}_{s,t'}^F$ up to a re-indexing of variables determined by a bijection

that maps, for each $i \in [s]$, the elements of $B \cap (\{i\} \times [t])$ to $(i, 1), \dots, (i, t')$. Thanks to Item a, clauses (41) are satisfied by ρ . All clauses (42) and (43) are satisfied: if $(i, j) \in B$ and $(i - 1, j') \notin B$, then the clause is satisfied by λ , otherwise it is satisfied by ρ or ν . Clauses (26) - (40) with $(i, j) \notin B$ are satisfied either by ρ (if $(i, j) \notin A$) or by ν . Clauses (26) - (40) with $(i, j) \in B$ become, after removing those clauses (28) - (31) that are satisfied by λ and after the re-indexing of variables, the clauses (11) - (25) with t replaced by t' . (Here notice that clauses (32) become (17) thanks to $(s, t) \in B$.) Hence $((R^k \text{REF}_{s,t}^F \upharpoonright \rho) \upharpoonright \nu) \upharpoonright \lambda$ is indeed $\text{REF}_{s,t'}^F$ up to the re-indexing of variables.

Let us now show that for a line G in $(\Pi \upharpoonright \rho) \upharpoonright \nu$ we have that $G \upharpoonright \lambda$ is, after the re-indexing of variables, strongly represented by a decision tree over $\text{REF}_{s,t'}^F$ of height at most $t/2^{k+5}$. As we already verified, $h_i(G) \leq t/2^{k+5}$, and therefore there is a tree T over $\text{REF}_{s,t}^F$ of minimum height which strongly represents G and whose height is at most $t/2^{k+5}$. Define a tree $T \upharpoonright \lambda$ by deleting all edges (and the corresponding subtrees) in T whose label is of the form $(C_{i,j}, \ell, j', j'')$ with $(i - 1, j') \notin B$ or $(i - 1, j'') \notin B$. $T \upharpoonright \lambda$ is, after relabelling its nodes and edges according to the re-indexing bijection, a decision tree over $\text{REF}_{s,t'}^F$. With every branch π of $T \upharpoonright \lambda$ we associate a partial assignment $\pi_{T \upharpoonright \lambda} : \text{Var}(((R^k \text{REF}_{s,t}^F \upharpoonright \rho) \upharpoonright \nu) \upharpoonright \lambda) \rightarrow \{0, 1\}$ defined via the re-indexing bijection and Definition 7, understanding the relabelled $T \upharpoonright \lambda$ as a tree over $\text{REF}_{s,t'}^F$. But every branch π of $T \upharpoonright \lambda$ is also a branch of T , hence Definition 7 with T (which is a tree over $\text{REF}_{s,t}^F$) says how π should be viewed as a partial assignment to $\text{Var}(\text{REF}_{s,t}^F)$; let us denote the partial assignment by π_T for clarity. It is easy to see from the definitions that for every branch π in $T \upharpoonright \lambda$, $\text{dom}(\lambda) \cap \text{dom}(\pi_{T \upharpoonright \lambda}) = \emptyset$ and $\pi_T \subseteq \lambda \cup \pi_{T \upharpoonright \lambda}$. It follows that $G \upharpoonright \lambda$ is strongly represented by $T \upharpoonright \lambda$. The tree $T \upharpoonright \lambda$ has, of course, height at most $t/2^{k+5}$.

We can now apply Theorem 11 taking $\text{REF}_{s,t'}^F$ (i.e., the re-indexed $((R^k \text{REF}_{s,t}^F \upharpoonright \rho) \upharpoonright \nu) \upharpoonright \lambda$) for H , t' for t , and $t/2^{k+5}$ for h , to obtain a resolution refutation of $\text{REF}_{s,t'}^F$ of index-width at most $3t/2^{k+5}$.

But we have

$$2 \cdot 3t/2^{k+5} < t/2^{k+2} < \lfloor t/2^{k+1} \rfloor - 2 = t',$$

where the second inequality follows from (10) and (55). Therefore, we can use Theorem 17, taking $3t/2^{k+5}$ for w and t' for t , to conclude that any resolution refutation of $\text{REF}_{s,t'}^F$ has index-width greater than $3t/2^{k+5}$. That is a contradiction.

I Proofs of Theorem 1 and Theorem 2

Proof of Theorem 1. Denote by F the well-known CNF $\neg\text{PHP}_n^{n+1}$ called the negation of the pigeonhole principle, expressing that a multi-valued function from $n + 1$ to n is injective. It consists of $r := n + 1 + (n^3 + n^2)/2$ clauses in $\tilde{n} := (n + 1)n$ variables.

Define $A_n := \text{SAT}_{\tilde{n}, r}^{\tilde{n}} \upharpoonright \gamma_F$, where γ_F is as in Proposition 19.

Since by [17, 19] there exists $\alpha > 0$ and an integer n_1 such that for every $n \geq n_1$, $\neg\text{PHP}_n^{n+1}$ has no $\text{Res}(k)$ refutations of size at most 2^{n^α} , the same is true for A_n . This is because by Proposition 19 there is a substitution τ such that $A_n \upharpoonright \tau$ is $\neg\text{PHP}_n^{n+1}$ together with some tautological clauses, and if Π is a $\text{Res}(k)$ refutation of A_n then $\Pi \upharpoonright \tau$ is a $\text{Res}(k)$ refutation of $A_n \upharpoonright \tau$. This shows Item i.

Define $B_{n,k} := R^k \text{REF}_{s,t}^F$, where we set $s := \tilde{n} + 1$ and $t := \tilde{n}^k$.

Let $\delta > 0$ and integer n_0 witness Theorem 18. Set $n_2 \geq n_0$ so that the hypotheses (10) with \tilde{n} in place of n hold with our choice of r, s, t (as functions of \tilde{n}) for all $\tilde{n} \geq n_2$. By that theorem, for every $\tilde{n} \geq n_2$, any $\text{Res}(k)$ refutation of $B_{n,k}$ has size greater than $2^{\beta(k)\tilde{n}}$. Item ii follows.

33:22 Failure of Feasible Disjunction Property and Automatability in k -DNF Resolution

Note that $\mathsf{R}^k\mathsf{REF}_{s,t}^{\tilde{n},r} \upharpoonright \gamma_F$ is $\mathsf{R}^k\mathsf{REF}_{s,t}^F$, because γ_F turns the clauses (6) into (26) (and the clauses satisfied by γ_F are removed). By Theorem 6 there is a $\mathsf{Res}(2)$ refutation of $\mathsf{SAT}_{s,t}^{\tilde{n},r} \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{\tilde{n},r}$ of size $O(k^2\tilde{n}^{3k+3})$. Hence the same holds for $A_n \wedge B_{n,k} = \mathsf{SAT}_{s,t}^{\tilde{n},r} \upharpoonright \gamma_F \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{\tilde{n},r} \upharpoonright \gamma_F$. This gives Item iii. \blacktriangleleft

Theorem 2 follows immediately from the more general Theorem 24 below. A function $T : \mathbb{N} \rightarrow \mathbb{N}$ is called *time-constructible* if there is an algorithm that when given 1^n (the string of n many 1's) computes $1^{T(n)}$ in time $O(T(n))$. We call a function $T : \mathbb{N} \rightarrow \mathbb{N}$ *subexponential* if $T(n) \leq 2^{n^{o(1)}}$.

► Theorem 24. *Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be time-constructible, non-decreasing and subexponential. If there is an integer $k \geq 1$ such that $\mathsf{Res}(k)$ is automatable in time T , then there are $c_1, c_2, c_3, c_4 > 0$ and an algorithm that when given as input a 3-CNF F in n variables decides in time $c_3(T(c_1n^{c_2k}) + n^k)^{c_4}$ whether F is satisfiable.*

Proof. Assume that for some integer $k \geq 1$ the system $\mathsf{Res}(k)$ is automatable in time T satisfying the assumptions of the theorem. Set r, s and t as functions of n as follows: $r := \binom{2n}{3}$, $s := n + 1$, $t := n^{k+3}$.

By Theorem 6 there are integers $c_1, c_2 > 0$ such that $\mathsf{SAT}^{n,r} \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{n,r}$ has a $\mathsf{Res}(2)$ refutation Π of size at most $c_1n^{c_2k}$; if necessary, increase c_1 and c_2 so that the size of Π plus the size of the formula $\mathsf{R}^k\mathsf{REF}_{s,t}^{n,r}$ is at most $c_1n^{c_2k}$.

Let $\delta > 0$ and integer $n_0 > 0$ witness Theorem 18. Let $n_1 > n_0$ be such that for all $n \geq n_1$,

$$r \leq t \leq 2^{\delta n} \tag{56}$$

and

$$2^{\beta(k)\frac{t}{n^{k-1}}} > T(c_1n^{c_2k}), \tag{57}$$

where $\beta(k)$ is as in Theorem 18. Here we use that T is subexponential.

Define algorithm M as follows. Given as input a 3-CNF F in n variables, check if $n \geq n_1$. If $n < n_1$, use brute force to decide if F is satisfiable or not, and output the answer. If $n \geq n_1$, compute the formula $\mathsf{R}^k\mathsf{REF}_{s,t}^F$ and run the automating algorithm on this formula for up to $T(c_1n^{c_2k})$ steps. If the automating algorithm returns a $\mathsf{Res}(k)$ refutation of $\mathsf{R}^k\mathsf{REF}_{s,t}^F$, then output “satisfiable”. Else output “unsatisfiable”.

Since both computing $\mathsf{R}^k\mathsf{REF}_{s,t}^F$ from F and checking whether the output of the automating algorithm is a $\mathsf{Res}(k)$ refutation of $\mathsf{R}^k\mathsf{REF}_{s,t}^F$ are polynomial-time procedures, and since T is time-constructible, it follows that there are $c_3, c_4 > 0$ such that the running time of M is at most $c_3(T(c_1n^{c_2k}) + n^k)^{c_4}$. It suffices to show that M gives the correct answer on 3-CNFs F in $n \geq n_1$ variables such that each clause of F has exactly three literals. Let F be such a 3-CNF, and let r' be the number of its clauses. We have $r' \leq r = \binom{2n}{3}$.

Assume first that F is satisfiable. Let γ_F and τ be as in Proposition 19, and let ν be a satisfying assignment for F . We have

$$(((\mathsf{SAT}^{n,r'} \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{n,r'}) \upharpoonright \gamma_F) \upharpoonright \tau) \upharpoonright \nu = ((\mathsf{SAT}^{n,r'} \upharpoonright \gamma_F) \upharpoonright \tau) \upharpoonright \nu \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{n,r'} \upharpoonright \gamma_F = \mathsf{R}^k\mathsf{REF}_{s,t}^F,$$

because by Proposition 19, $(\mathsf{SAT}^{n,r'} \upharpoonright \gamma_F) \upharpoonright \tau$ is F together with some tautological clauses in the variables x_1, \dots, x_n . Let Π' be the $\mathsf{Res}(2)$ refutation of $\mathsf{SAT}^{n,r'} \wedge \mathsf{R}^k\mathsf{REF}_{s,t}^{n,r'}$ given by Theorem 6. Then $\Pi'' := ((\Pi' \upharpoonright \gamma_F) \upharpoonright \tau) \upharpoonright \nu$ is a $\mathsf{Res}(2)$ refutation of $\mathsf{R}^k\mathsf{REF}_{s,t}^F$ (note that it is actually a resolution refutation), and we have

$$\begin{aligned}
\text{size}(\Pi'') + \text{size}(\text{R}^k \text{REF}_{s,t}^F) &\leq \text{size}(\Pi') + \text{size}(\text{R}^k \text{REF}_{s,t}^{n,r'}) \\
&\leq \text{size}(\Pi) + \text{size}(\text{R}^k \text{REF}_{s,t}^{n,r}) \\
&\leq c_1 n^{c_2 k}.
\end{aligned}$$

Because T is non-decreasing, the automating algorithm finds within the allotted time $T(c_1 n^{c_2 k})$ a $\text{Res}(k)$ refutation of $\text{R}^k \text{REF}_{s,t}^F$, and M outputs “satisfiable”.

Assume now that F is unsatisfiable. From our choices of r, s, t and n_1 and from (56) it follows that the hypotheses (10) of Theorem 18 are met for all $n \geq n_1$, and the same is true with r' in place of r . By that theorem, any $\text{Res}(k)$ refutation of $\text{R}^k \text{REF}_{s,t}^F$ has size greater than $2^{\beta(k) \frac{t}{n^{k-1}}}$. Thanks to (57) this implies that the automating algorithm cannot output any $\text{Res}(k)$ refutation of $\text{R}^k \text{REF}_{s,t}^F$ within the allotted time. M therefore outputs “unsatisfiable”. ◀

Search-To-Decision Reductions for Kolmogorov Complexity

Noam Mazor ✉

Tel Aviv University, Israel

Rafael Pass ✉

Tel Aviv University, Israel

Cornell Tech, New York, NY, USA

Abstract

A long-standing open problem dating back to the 1960s is whether there exists a *search-to-decision* reduction for the time-bounded Kolmogorov complexity problem – that is, the problem of determining whether the length of the shortest time- t program generating a given string x is at most s .

In this work, we consider the more “robust” version of the time-bounded Kolmogorov complexity problem, referred to as the GapMINKT problem, where given a size bound s and a running time bound t , the goal is to determine whether there exists a $\text{poly}(t, |x|)$ -time program of length $s + O(\log |x|)$ that generates x . We present the first non-trivial search-to-decision reduction R for the GapMINKT problem; R has a running-time bound of $2^{\epsilon n}$ for any $\epsilon > 0$ and additionally only queries its oracle on “thresholds” s of size $s + O(\log |x|)$. As such, we get that any algorithm with running-time (resp. circuit size) $2^{\alpha s} \text{poly}(|x|, t, s)$ for solving GapMINKT (given an instance (x, t, s)), yields an algorithm for finding a witness with running-time (resp. circuit size) $2^{(\alpha+\epsilon)s} \text{poly}(|x|, t, s)$.

Our second result is a polynomial-time search-to-decision reduction for the time-bounded Kolmogorov complexity problem in the average-case regime. Such a reduction was recently shown by Liu and Pass (FOCS’20), heavily relying on cryptographic techniques. Our reduction is more direct and additionally has the advantage of being *length-preserving*, and as such also applies in the exponential time/size regime.

A central component in both of these results is the use of Kolmogorov and Levin’s *Symmetry of Information Theorem*.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Kolmogorov complexity, search to decision

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.34

Related Version *Full Version*: <https://ecc.weizmann.ac.il/report/2024/003/> [18]

Funding *Noam Mazor*: Research partly supported by NSF CNS-2149305 and DARPA under Agreement No. HR00110C0086.

Rafael Pass: Supported in part by AFOSR Award FA9550-23-1-0387, AFOSR Award FA9550-23-1-0312, and an Algorand Foundation grant. This material is based upon work supported by DARPA under Agreement No. HR00110C0086. 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 United States Government, DARPA, AFOSR or the Algorand Foundation.

1 Introduction

In his historical account, Trakhtenbrot [22] describes efforts in the 1960s in the Russian Cybernetics program to understand problems requiring *brute-force search* to solve [22, 23, 24]. The so-called *Perebor* (Russian for brute-force search) conjectures refer to the conjectures that certain types of, what today are referred to as “meta-complexity”, problems require brute-force search to be solved. These include (a) the *Minimum Circuit Size problem*



© Noam Mazor and Rafael Pass;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 34; pp. 34:1–34:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



(MCSP) [12, 22] – finding the smallest Boolean circuit that computes a given function x , and (b) the *Time-Bounded Kolmogorov Complexity Problem* [14, 21, 2, 13, 6, 20] – computing the length, denoted $K^t(x)$ of the shortest program (evaluated on some particular Universal Turing machine U) that generates a given string x , within time t , where $t = \text{poly}(|x|)$ is a polynomial.

Our focus in this paper is on the Time-Bounded Kolmogorov Complexity problem. As explained by Trakhtenbrot, two versions of this problem were considered since the 1960s.

- **The “existential” (i.e., decisional) version:** Given a string x and a threshold s , determining whether $K^t(x)$ is less than “roughly” s .
- **The “constructive” (i.e., search) version:** Given a string x and a threshold s , if $K^t(x)$ is less than “roughly” s , finding a program π of length “roughly” s that certifies this.

Both of these problems are conjectured to require *brute-force search*: that is, to require algorithms with running time close to 2^n where $n = |x|$ is the size of the given instance x . This is referred to as the *Perebor conjecture* (with respect to the time-bounded Kolmogorov complexity problem) and can be viewed as an early precursor, and stronger form, of the $\mathbf{NP} \neq \mathbf{P}$ conjecture (as the problem trivially resides in \mathbf{NP}). In fact, not only no non-trivial uniform algorithms are known for the search versions¹, but there are also no non-trivial (uniform) algorithms (i.e., beating brute-force search) even if we have access to an oracle solving the decisional version: That is, the only *search-to-decision reduction* is simply to ignore the decision oracle and solve the search version using brute-force search.²

The central result of this paper is developing the first non-trivial search-to-decision reduction for a gap-version of time-bounded Kolmogorov complexity; more precisely, we develop such a reduction with running time $2^{\epsilon n}$ for every $\epsilon > 0$.

We additionally address search-to-decision reductions in the *average-case* regime (w.r.t. the uniform distribution over instances). There, recently, Liu and Pass demonstrated a polynomial time reduction [16], but the reduction is not *length preserving* and as such it cannot be applied in the exponential regime. As our second result, we present a new direct proof of the result of [16], but achieve also a length-preserving polynomial-time reduction (which thus also applies in the exponential regime).

1.1 Our Results

To explain our results, let us first recall the MINKT and GapMINKT problems.

The GapMINKT Problem

Following Ko [13], we let MINKT denote the set of strings $(x, 1^t, 1^s)$ such that $K^t(x) \leq s$. Since the notion of Kolmogorov complexity is highly dependent on the choice of the universal Turing machine, a natural – and more “robust” – variant of this problem allows for (a) some polynomial overhead in terms of the running time, and (b) some logarithmic slackness in terms of the threshold [15, 13]. Following Hirahara [9], we refer to Gap_pMINKT as the promise problem where:

¹ As we shall discuss shortly, two recent works [17, 11] demonstrate *circuits* (i.e., non-uniform algorithms) of size $2^{4n/5n+o(n)}$ that solve it.

² As just mentioned, in the non-uniform regime, non-trivial algorithms are known, but the same algorithm solves both the search and the decisional problem, so also in the non-uniform setting, the best known approach is to simply ignore the decisional oracle and solving the search problem from scratch.

- YES-instances consist of strings $(x, 1^t, 1^s)$ where $K^t(x) \leq s$;
 - NO-instances consist of strings $(x, 1^t, 1^s)$ where $K^{p(t, |x|)}(x) > s + \log p(t, |x|)$;
- and we say that an algorithm A solves GapMINKT if there exists some polynomial p such that A decides Gap_pMINKT. Furthermore, we say that A solves the search version of the problem, search-GapMINKT if there exists some polynomial p such that given any Gap_pMINKT YES-instance $(x, 1^t, 1^s)$, A outputs a program Π certifying that $(x, 1^t, 1^s)$ is not a NO-instance (i.e, the program can run in time $p(t, |x|)$ and have length at most $s + \log p(t, |x|)$).

Non-trivial Search-to-Decision for GapMINKT

Our first (and main) result, is a non-trivial search-to-decision reduction for GapMINKT.

► **Theorem 1 (Informal).** *For any $\varepsilon > 0$, and every polynomial τ , there exists a randomized oracle-aided algorithm F such that for every A that decides Gap _{τ} MINKT, F^A solves search-GapMINKT.*

Moreover, on input $(x, 1^t, 1^s)$ F runs in time $2^{\varepsilon s} \text{poly}(|x|, t, s)$, and only queries A on inputs $(y, 1^{t'}, 1^{s'})$ with $|y| = \text{poly}(|x|, t, s)$, $t' = \text{poly}(|x|, t, s)$, and $s' \leq s + \log \text{poly}(|x|, t, s)$.

F^A solves search-GapMINKT with gap $\tau^{O(1/\varepsilon)}$. We remark that the reduction is not fully length preserving – the reduction invokes its oracle on statements x' that are longer than the original statement x , and at first sight, it may thus seem that the reduction is not useful in the regime of exponential hardness.

The key point, however, is that it only invokes the oracle on thresholds s' that are of roughly the same size as the original threshold s . Therefore, since the hardness of the GapMINKT problem most naturally should be thought to be a function of the threshold s (as there is a trivial $\text{poly}(|x|, t, s)2^s$ algorithm, namely brute-force search), this reduction still yields a non-trivial search-to-decision reduction in the exponential regime:

► **Corollary 2.** *For any $\varepsilon > 0, \alpha > 0, \tau \in \text{poly}$, assume that there is an algorithm that solves Gap _{τ} MINKT in time (resp. size) $2^{\alpha s} \text{poly}(|x|, t, s)$. Then there exists an algorithm that solves search-GapMINKT in time (resp. size) $2^{(\alpha+\varepsilon)s} \cdot \text{poly}(|x|, t, s)$ on inputs $(x, 1^t, 1^s)$.*

An Average-case Search-to-Decision Reduction

We turn to considering the average-case regime. Here we provide a polynomial-time reduction that additionally is *length-preserving* and as such directly also applies in the exponential regime. (This is in contrast to the earlier average-case search-to-decision reduction of [16] that did not apply in the exponential regime.)

► **Theorem 3.** *For every $p, t \in \text{poly}$ there exists $\hat{p} \in \text{poly}$ and $t' \in \text{poly}$, and an efficient oracle-aided algorithm F such that the following holds. Let A be an algorithm that computes $K^{t'}$ with probability $1 - 1/\hat{p}$ on the uniform distribution. Then F^A solves search- K^t with probability $1 - 1/p$ on the uniform distribution.*

Moreover, on input $x \in \{0, 1\}^n$, F makes only queries of the form $x||y$ with $y \in O(\log n)$.

As corollaries, we thus get:

► **Corollary 4 (reproving [16]).** *For every $p, t \in \text{poly}$ there exists $\hat{p}, t' \in \text{poly}$ such that the following holds: if there exists a polynomial time (reps. $2^{o(|x|)}$ time) algorithm that computes $K^{t'}$ with probability $1 - 1/\hat{p}$ over the uniform distribution, then exists polynomial time (resp. $2^{o(|x|)}$ time) algorithm that solves search- K^t with probability $1 - 1/p$ on the uniform distribution. Moreover, the same holds also in the non-uniform setting (i.e., w.r.t. polynomial-size and respectively $2^{o(|x|)}$ size algorithms).*

► **Corollary 5 (new).** *For every $p, t \in \text{poly}$ there exists $\hat{p}, t' \in \text{poly}$ such that the following holds: if there exists a constant $\alpha > 0$ and a $2^{\alpha \cdot |x|} \text{poly}(|x|)$ time (resp. size) algorithm that computes $K^{t'}$ with probability $1 - 1/\hat{p}$ over the uniform distribution, then there exists a $2^{\alpha \cdot |x|} \text{poly}(|x|)$ time (resp. size) algorithm that solves search- K^t w.p $1 - 1/p$ over the uniform distribution.*

We note that while our reduction improves on [16] in the length-preserving aspect (and additionally is significantly simpler), it also has some disadvantages: in particular, in [16] an oracle for $K^{t'}$ for *any* polynomial t' can be used to solve search- K^t for any other polynomial t , whereas in our case, the reduction only works as long as t is sufficiently larger than t' . Additionally, the same thing holds also with respect to the error probability polynomials \hat{p}, p . The reasons for these “amplifications” is that [16] passes through cryptographic techniques (hardness amplification [25], and constructions of pseudorandom generators [7]) that blow up the input size.

1.2 Related Works

While, as far as we know, no non-trivial search-to-decision reductions were previously known for GapMINKT in the worst-case regime, there are several works that consider variants of this question:

Slightly Subexponential Search-to-Decision for MFSP: An elegant work by Ilango considers a formula size variant of the classic Perebor conjecture problem, MFSP, where the goal is to find the shortest formula computing some given function. He demonstrates a search-to-decision reduction with running-time $2^{0.67n}$ for MFSP. As far as we know, this is the first result to demonstrate any non-trivial search-to-decision reduction for a Perebor-style problem. (We note that in contrast, we here consider the standard time-bounded Kolmogorov complexity problem, and we also get a smaller running time of $2^{\epsilon n}$ for any $\epsilon > 0$.) Ilango also gets an improved running time of $2^{n/\log \log n}$ if only requiring an algorithm that succeeds on *most* (i.e., a $1 - 1/o(1)$) fraction of instances. In contrast, in this setting, we get a polynomial running time.

Average-case Search-to-Decision for MINKT: Liu and Pass [16] show a *polynomial-time* algorithm that solves the search-GapMINKT) *on average* over the uniform (over x , and for every t, s) given access to an oracle that solves GapMINKT on average. Our second result is a strict strengthening of this result since our reduction is length-preserving (i.e., it only queries its oracle on input lengths that are $O(\log n)$ longer), and as such it also applies in the exponential regime (whereas the result of [16] only apply in the polynomial to subexponential regimes).

Conditional and Non-black-box Search-to-Decision for GapMINKT: An intriguing work by Hirahara [10] presents a *non-black-box* search to decision reduction for GapMINKT in the polynomial regime, under standard derandomization assumptions. More precisely, assuming that E does not have subexponential-size circuits, he shows that if GapMINKT has a (deterministic, wlog due to the assumption) polynomial-time decider, then search-GapMINKT has a polynomial-time algorithm. His result does not extend to the non-uniform setting, or to algorithms running in time even just $n^{\log n}$ due to the fact that the code of the GapMINKT attacker gets incorporated into the witness for the search-GapMINKT problem. In contrast, ours is unconditional; on the other hand, ours is only meaningful in the exponential regime (as the running time of the reduction is subexponential).

A different paper by Hirahara [8] gets an unconditional non-black-box search-to-decision reduction for the polynomial regime for $\text{Gap}_\tau\text{MINKT}$ w.r.t $\tau = 2^{\sqrt{n}}$.³ This result also does not apply in the non-uniform setting, but does extend to the subexponential (but not exponential) regime.

Search-to-Decision Reductions w.r.t. Black-box Solvers: In a very recent work, the current authors consider *black-box* solvers for the MINKT problem that solve the problem no matter what the underlying Universal Turing Machine U is, given black-box access to it. A polynomial-size black-box search-to-decision reduction is demonstrated with respect to such attackers. In contrast, we here consider all, and not just black-box, solvers.

Non-uniform Algorithms Beating Perebor: As mentioned above, two independent recent works [17, 11] develop algorithms solving the MINKT problem using a circuit of size $2^{4n/5}\text{poly}(n)$, disproving the “non-uniform” version of the Perebor conjecture. These algorithms directly also work for the search version of the problem and as such, even in the non-uniform regime, it was not known how to make use of an GapMINKT oracle to solve the search version better than simply solving it from scratch.

1.3 Proof Overview

We provide a brief overview of the proofs of Theorem 1 and 3, starting with Theorem 1, which proceeds in two steps.

Worst-case Search-to-Decision Reduction for “Shallow” Instances

As an intermediary step, which may be of independent interest, we start by providing a search-to-decision reduction whose running time is a function of the so-called *computational depth* of the instance x we are reducing from (i.e., that we want to find a witness for). Recall that the computational depth [1] of an instance x is defined as $cd^t(x) = K^t(x) - K(x)$. Note that by a standard counting argument, we have that “computationally deep” strings (i.e., string x such that $cd^t(x) > O(\log(|x|))$) are rare.

We start by presenting a search-to-decision reduction with running time

$$2^{cd^t(x)} \text{poly}(|x|, t, s)$$

(which thus for most strings runs in polynomial time). The key idea behind the reduction is the following. Given a string x , and a minimal-length t -time program Π generating x , the $K^{t'}$ -complexity of the string $x||\Pi$, for $t' = \text{poly}(|x|, t)$, is not significantly higher than the K^t -complexity of the string x – since the string $x||\Pi$ also can be generated by a self-printing variant of Π . Furthermore, the above argument also holds even if we concatenate not only the whole of Π but even just a prefix of it.

Thus, if we have access to a GapMINKT oracle, we ought to be able to find Π “bit-by-bit” by simply concatenating a bit to x and checking if the $K^{t'}$ -complexity remains below $s + O(\log |x|)$. In more detail, we keep track of a set \mathcal{S} of candidates y (whose prefix is x) and at each iteration concatenate each bit $b \in \{0, 1\}$ to y and check whether the $K^{t'}$ -complexity of $y||b$ remains small, and if so adding $y||b$ to the set \mathcal{S} . By the argument above and a standard induction, we have that at iteration i , $y = x||\Pi_{\leq i}$ (where $\Pi_{\leq i}$ denotes the i first bit of Π) must be in the set \mathcal{S} , so we can finally find Π by simply going over all the elements $y = x||\Pi'$ of \mathcal{S} and checking whether Π' generates x .

³ The results is actually a bit stronger – the running time only increases by a polynomial, but the gap increases by \sqrt{n} , as opposed to the desired $O(\log n)$.

The problem, of course, is that the set \mathcal{S} could contain lots of other elements. This is where computational depth enters the picture. To see why, let us first start by showing that if we had been dealing with Kolmogorov complexity, as opposed to t -bounded Kolmogorov complexity, then the size of \mathcal{S} can never be more than of polynomial size (in $|x|, t$). In fact, this follows almost directly from Kolmogorov and Levin’s celebrated *symmetry of information* (SoI) theorem [26] which states that for any strings a, b , we have that⁴

$$K(a||b) \geq K(a) + K(b|a) - O(\log(|a| + |b|)).$$

Indeed, recall that \mathcal{S} consists of all strings $y = x||\Pi$ whose K -complexity is roughly that of x ; by the SoI theorem, setting $a = x$ and $b = \Pi$, we get that $K(\Pi|x) \leq O(\log|x|)$ and thus there can be at most $\text{poly}(|x|)$ such strings.⁵

Finally, note that if considering a string x whose computational depth is d , then $K^t(x) - K(x) \leq d$, and as such for each element $x||\Pi$ that remains in \mathcal{S} , we have that

$$\begin{aligned} K(x||\Pi) - K(x) &\leq K^t(x||\Pi) - K(x) \\ &\leq K^t(x) + O(\log|x|) - K(x) \\ &\leq K^t(x) + O(\log|x|) - (K^t(x) - cd^t(x)) \\ &\leq d + O(\log|x|) \end{aligned}$$

Thus, by the SoI theorem, we then get that $K(\Pi|x) \leq d + O(\log|x|)$, and therefore we have that $|\mathcal{S}| \leq 2^d \text{poly}(|x|)$. As such, the running of our algorithm becomes $2^{cd^t(x)} \text{poly}(|x|, t, s)$, as desired.

Dealing with Deep Instances

Note that given any instances x whose computational depth is bounded by $\epsilon|x|$, then the running time of the above algorithm becomes $2^{\epsilon|x|} \text{poly}(|x|, t, s)$, as desired. If not, and in case the algorithm’s running time becomes larger than this, we must have that the set \mathcal{S} produced is bigger than $2^{\epsilon|x|}$. Our key idea now is to simply stop the algorithm once the size of \mathcal{S} reaches $2^{\epsilon|x|}$, and at this point selecting a *random element* in $x' \in \mathcal{S}$, and restarting the algorithm on x' instead (since a program generating x' can easily be modified to a program generating x). The reason for doing this is that since the set \mathcal{S} is “big”, by choosing a random element, we are guaranteed that the *actual* (i.e., not time-bounded) Kolmogorov complexity of the chosen string x' is roughly ϵn larger than that of x , yet since all strings in \mathcal{S} have roughly the same time-bounded Kolmogorov complexity ($s + O(\log|x|)$), we must have that the computational depth of x' is at least ϵn smaller than that of x . In essence, by picking this random element, we are able to get a new instance x' such that (a) the witness for x' is also a valid witness for x , yet (b) the computational depth of x' is $\epsilon|x|$ smaller than that of x .

By iteratively continuing this process, we eventually (after $1/\epsilon$ steps) end up with an element with small computational depth and thus manage to find a witness in the desired running time.

⁴ Recall that the conditional Kolmogorov complexity of b given a , denoted by $K(b|a)$ is the minimal length of a program that outputs b given input a .

⁵ This result may be of independent interest. It shows a polynomial-time “list-to-decision” reduction for Kolmogorov-complexity – that is, a polynomial-time algorithm that given access to a decision oracle outputs a polynomial-length list of candidate witnesses, one of which is correct. The reason why this does not yield a *search-to-decision* reduction is that we cannot, in polynomial-time, determine if a witness is correct by running it.

Let us highlight why this approach only gives an algorithm with subexponential running time: The issue is that each time we pick a random element $x' \in \mathcal{S}$, the time-bounded Kolmogorov complexity of the element may increase by $O(\log |x|)$, so we can only afford a constant number of iterations, which is why we need to make sure that we can eliminate a constant fraction of the computational depth in each step. (An additional reason is that the running-time t' blows up as a polynomial of t in each iteration, so again, we can only afford a constant number of iterations.)

Search-to-Decision in the Average-case Regime

We turn to discussing our search-to-decision reduction in the average-case regime. The goal is to show how to use an oracle that (decides, or equivalently, computes) K^t with high probability on the uniform distribution to find a $K^{t'}$ witness with high probability over the uniform distribution for a polynomially related t' .

Towards this, we will show a reduction that again works in the worst-case regime, but only on computationally shallow instances – that is, instances x with computational depth $O(\log |x|)$. This reduction will improve on the one above in the sense that it is *length preserving*; additionally, due to the length-preserving aspect of the reduction, it will also follow that if we only require the reduction to work with high probability (over the uniform distribution) over instances, then it suffices for the oracle to also work with high probability.

The idea is to, given an instance x , consider strings $y = x||i||\Pi_i$, where Π_i is the i th “chunk” (of length $O(\log |x|)$) of the smallest time-bounded program Π generating x ; such strings still have roughly the same time-bounded Kolmogorov complexity as x , and by the same argument based on symmetry-of-information, we can argue that there cannot be more than polynomially many strings y that have x as a prefix and also have roughly the same time-bounded Kolmogorov complexity as x . This enables us to recover a small set of candidates for (most) of the “coordinates” of Π . But, even if there are just 2 candidates for each such coordinate, there will still be too many options to try out, as the number of coordinates is polynomial in $|\Pi|/|\Pi_i|$ which can be as large as $|x|/\log|x|$.

To solve this problem, we will rely on the notion of a *list-recoverable error-correcting code* [5, 3] – in essence, a type of an error-correcting code (ECC) from which we recover a polynomial-length list of candidate messages (one of which is guaranteed to be the true one) given a polynomial-length candidate list for each symbol of the encoding. Roughly speaking, we find all strings $y = x||i||z_i$ that have small time-bounded Kolmogorov complexity, and then apply the list-recoverable procedure of the ECC. By the existence of *efficient* list-recoverable codes [5] (where both the encoding and decoding can be done efficiently), we are still guaranteed that when z_i is the i th symbol of the encoding of Π , then y indeed has small time-bounded Kolmogorov complexity; next, the above symmetry-of-information based argument will ensure that we can only have a “small” number of candidates for most coordinates i , and as such, the list-recovering procedure will indeed find some short program Π .

There is just one catch with this argument: using the above SoI based argument we will get a too weak bound on the number of possible candidates for each symbol. We note, however, that we can use the same argument to bound the *total* number of strings of the form (x, i, z) with small time-bounded Kolmogorov complexity, and as such, use an averaging argument to argue that for, say 90% of the coordinates, we get a sufficiently small list of symbols. The issue remaining is that we can no longer rely on the list-recoverability property to recover the message (as we no longer have a short list for *every* symbol of the codeword). Luckily, there exist list-recoverable codes satisfying exactly this property (i.e.,

that we can recover a polynomial-length list of messages, even if we only have a bound on the set of symbols, for a constant fraction of the coordinates) – indeed, as shown in [5, 4], the Reed-Solomon code also satisfies such list-recoverability “with errors”.

To finally see why this reduction also works in the average-case regime, first recall that computationally deep strings are rare, so the reduction will work with high probability over x , as long as the oracle works on all instances. Next, note that the reduction, given an instance x , only queries its oracles on instances of roughly the same length as x , and that have x as a prefix, which suffices to argue that we only need an oracle that works with high probability.

2 Preliminaries

2.1 Notations

All logarithms are taken in base 2. We use calligraphic letters to denote sets and distributions, uppercase for random variables, and lowercase for values and functions. Let poly stand for the set of all polynomials. Given a vector $v \in \Sigma^n$, let v_i denote its i^{th} entry, let $v_{<i} = (v_1, \dots, v_{i-1})$ and $v_{\leq i} = (v_1, \dots, v_i)$. For $x, y \in \{0, 1\}^*$, we let xy and $x||y$ denote the concatenation of the strings x and y . An oracle-aided algorithm A is an algorithm with an oracle access. Given a (randomized) function \mathcal{O} , we use $A^{\mathcal{O}}$ to denote the algorithm A when using \mathcal{O} as the oracle.

2.2 Distributions and Random Variables

When unambiguous, we will naturally view a random variable as its marginal distribution. The support of a finite distribution \mathcal{P} is defined by $\text{Supp}(\mathcal{P}) := \{x : \Pr_{\mathcal{P}}[x] > 0\}$. For a (discrete) distribution \mathcal{P} , let $x \leftarrow \mathcal{P}$ denote that x was sampled according to \mathcal{P} . Similarly, for a set \mathcal{S} , let $x \leftarrow \mathcal{S}$ denote that x is drawn uniformly from \mathcal{S} .

2.3 Kolmogorov Complexity

Roughly speaking, the t -time-bounded Kolmogorov complexity, $K^t(x)$, of a string $x \in \{0, 1\}^*$ is the length of the shortest program $\Pi = (M, y)$ such that, when simulated by a universal Turing machine, Π outputs x in t steps. Here, a program Π is simply a pair of a Turing Machine M and an input y , where the output of Π is defined as the output of $M(y)$. When there is no running time bound (i.e., the program can run in an arbitrary number of steps), we obtain the notion of Kolmogorov complexity.

In the following, let $U(\Pi, 1^t)$ denote the output of Π when emulated on U for t steps. We now define the notion of Kolmogorov complexity with respect to the universal TM U .

► **Definition 6.** Let $t \in \mathbb{N}$ be a number. For all $x \in \{0, 1\}^*$, define

$$K_U^t(x) = \min_{\Pi \in \{0, 1\}^*} \{|\Pi| : U(\Pi, 1^t) = x\}$$

where $|\Pi|$ is referred to as the description length of Π . Similarly, for every $z \in \{0, 1\}^*$ define

$$K_U^t(x | z) = \min_{\Pi \in \{0, 1\}^*} \{|\Pi| : U(\Pi(z), 1^t) = x\}.$$

When there is no time bound, we define

$$K_U(x) = \min_{\Pi \in \{0, 1\}^*} \{|\Pi| : \exists t \in \mathbb{N} \text{ s.t. } U(\Pi, 1^t) = x\}$$

and

$$K_U(x | z) = \min_{\Pi \in \{0, 1\}^*} \{|\Pi| : \exists t \in \mathbb{N} \text{ s.t. } U(\Pi(z), 1^t) = x\}.$$

It is well known that for every x , $K^t(x) \leq |x| + c$, for some constant c depending only on the choice of the universal TM U .

► **Fact 7.** *For every universal TM U , there exists a constant c such that for every $x \in \{0, 1\}^*$, and for every t such that $t(n) > 0$, $K_U^t(x) \leq |x| + c$.*

We will also use the following fact, which states that we can efficiently encode a pair (x, y) with a small overhead.

► **Fact 8.** *There exists $q \in \text{poly}$ such that the following holds for every $x, y \in \{0, 1\}^*$,*

$$K^{q(|xy|)}(x, y) \leq |x| + |y| + \log|x| + 2 \log \log|x| + O(1).$$

We will use the following bound on the Kolmogorov complexity of strings sampled from the uniform distribution.

► **Lemma 9.** *For any universal TM U , any string $x \in \{0, 1\}^*$ and any set S , it holds that*

$$\Pr_{y \leftarrow S}[K_U(y | x) < \log|S| - i] \leq 2^{-i}.$$

In this paper, unless otherwise stated, we fix some universal Turing machine U that can emulate any program Π with polynomial overhead, and let $K^t = K_U^t$ and $K = K_U$.

The computational depth of a string is the difference between its Kolmogorov complexity and its time-bounded Kolmogorov complexity.

► **Definition 10** (Computational depth [1]). *For $x \in \{0, 1\}^*$ and $t \in \mathbb{N}$, the computational depth of x is defined to be $cd^t(x) = K^t(x) - K(x)$.*

Since, by a simple counting argument, most strings $x \in \{0, 1\}^n$ have $K^t(x)$ close to n , it holds that most strings have small computational depth.

► **Fact 11.** *For every $n \in \mathbb{N}$ and every $t \in \mathbb{N}$, $\Pr_{x \leftarrow \{0, 1\}^n}[cd^t(x) > i] \leq 2^{-i}$.*

We will also use the Symmetry of Information lemma.

► **Theorem 12** (Symmetry of Information [26]). *There exists a constant $c \in \mathbb{N}$ such that for every $x, y \in \{0, 1\}^*$,*

$$K(x) + K(y | x) + c \log(|x| + |y|) \geq K(x|y) \geq K(x) + K(y | x) - c \log(|x| + |y|)$$

We next define MINKT and GapMINKT.

► **Definition 13** (MINKT). *MINKT is the following promise problem:*

- $\mathcal{Y} = \{(x, 1^t, 1^s) : K^t(x) \leq s\}$
- $\mathcal{N} = \{(x, 1^t, 1^s) : K^t(x) > s\}$

We say that an algorithm A solves search-MINKT if A finds a program Π such that $|\Pi| \leq s$ and $U(\Pi, 1^t) = x$, for every $(x, 1^t, 1^s) \in \mathcal{Y}$.

We remark that MINKT is actually a language (every possible input is either in \mathcal{Y} or in \mathcal{N}), and we define it as a promise problem for easy comparison with GapMINKT.

► **Definition 14** (GapMINKT). *Let $\tau \in \text{poly}$ be a polynomial such that $\tau(t, |x|) \geq t$ for every t, x . Then Gap $_{\tau}$ MINKT is the following promise problem:*

- $\mathcal{Y} = \{(x, 1^t, 1^s) : K^t(x) \leq s\}$
- $\mathcal{N}_{\tau} = \{(x, 1^t, 1^s) : K^{\tau(t, |x|)}(x) > s + \log \tau(t, |x|)\}$

We say that an algorithm A decides GapMINKT if there exists $\tau \in \text{poly}$ such that A decides Gap $_{\tau}$ MINKT.

We say that a (randomized) algorithm A solves search-GapMINKT if there exists $\tau \in \text{poly}$ such that A (with probability $1/2$) finds a program Π such that $|\Pi| \leq s + \log \tau(t, |x|)$ and $U(\Pi, 1^{\tau(t, |x|)}) = x$, for every $(x, 1^t, 1^s) \in \mathcal{Y}$.

3 Decision-to-Search for Shallow Instances

In this part we prove our search-to-decision reduction for inputs with small computational depth.

► **Theorem 15.** *There exists an oracle-aided algorithm F such that the following holds. Let A be an oracle that decides GapMINKT. Then F^A solves search-MINKT.*

Moreover, on input $(x, 1^t, 1^s)$, F runs in time $2^{cd^t(x)} \text{poly}(|x|, t, s)$, and only queries A on inputs $(y, 1^{t'}, 1^{s'})$ with $|y| \leq |x| + s$, $t' \in \text{poly}(|x|, t)$, and $s' \leq s + O(\log(|x| + t))$.

Directly from Theorem 15 we get the following corollary.

► **Corollary 16.** *The following holds:*

- *Assume that there is a poly-time (resp. poly size) algorithm that solves GapMINKT. Then there exists an algorithm that solves search-GapMINKT in time (resp. size) $2^{cd^t(x)} \cdot \text{poly}(|x|, t, s)$ on inputs $(x, 1^t, 1^s)$.*
- *Assume that for some $\alpha > 0$ there is an algorithm that solves GapMINKT in time (resp. size) $2^{\alpha \cdot s} \text{poly}(|x|, t, s)$. Then there exists an algorithm that solves search-GapMINKT in time (resp. size) $2^{cd^t(x) + \alpha \cdot s} \cdot \text{poly}(|x|, t, s)$ on inputs $(x, 1^t, 1^s)$.*

The proof of Theorem 15 is almost immediate from the following lemma, in which the running time of the algorithm that solves MINKT is larger for high values of the threshold s .

► **Lemma 17.** *There exists an oracle-aided algorithm F' such that the following holds. Let A be an oracle that decides GapMINKT. Then F'^A solves search-MINKT.*

Moreover, on input $(x, 1^t, 1^s)$, F' runs in time $2^{s - K(x)} \text{poly}(|x|, t, s)$, and queries A on inputs $(y, 1^{t'}, 1^{s'})$ with $|y| \leq |x| + s$, $t' \in \text{poly}(t)$, and $s' \leq s + O(\log(|x| + t))$.

Proof of Theorem 15. Let F be the algorithm that given $(x, 1^t, 1^s)$ runs F' on input $(x, 1^t, 1^{s'})$ for every $s' = 1, \dots, s$, until the first execution that outputs a program Π with $U(\Pi, 1^t) = x$. The theorem follows since the algorithm halts when $s' = K^t(x)$. ◀

We next prove Lemma 17. In the proof of Lemma 17 we will use the following claim.

▷ **Claim 18.** *There exists a polynomial $q \in \text{poly}$ and a constant c_0 , such that the following holds for every $x \in \{0, 1\}^*$ with $|x| \geq 2$, and every $t \in \mathbb{N}$. Let Π a program of length $K^t(x)$ such that $U(\Pi, 1^t) = x$. Then for every $i \leq K^t(x)$, $K^{q(t, |x|)}(x || \Pi_{\leq i}) \leq K^t(x) + c_0 \log|x|$.*

Proof of Claim 18. Let U' be an efficient program such that $U'(\Pi, i) = U(\Pi) || \Pi_{\leq i}$ (where we encode Π, i using Fact 8), and let q be a polynomial such that $q(t, |x|)$ is an upper bound on the running time of U' where t is a bound on the running time of $U(\Pi)$ (recall that $|\Pi| = K^t(x) \leq |x| + O(1)$).

Then $K^{q(t, |x|)}(U(\Pi) || \Pi_{\leq i}) \leq |(\Pi, i)| + O(1) \leq |\Pi| + 3 \log|\Pi| + O(1) \leq |\Pi| + 3 \log(|x| + c) + O(1)$, where the last inequality holds by Fact 7, for some constant $c \in \mathbb{N}$. ◀

To prove Lemma 17, let $q \in \text{poly}$ and c_0 be the polynomial and constant promised by Claim 18, and consider the following algorithm that finds a minimal K^t -witness.

► **Algorithm 19** (F').

Oracle: GapMINKT decider A .

Input: $(x, 1^t, 1^s)$ for $x \in \{0, 1\}^*$, $t, s \in \mathbb{N}$.

1. Set $\mathcal{S}_0 = \{x\}$ and $k = s + c_0 \log|x|$.
2. For every $i = 1, 2, \dots, s$:
 - a. Compute $\mathcal{S}_i = \{yb : y \in \mathcal{S}_{i-1}, b \in \{0, 1\} \text{ and } A(yb, 1^{q(t, |x|)}, 1^k) = \text{Yes}\}$
 - b. If exists $y \in \mathcal{S}_i$ such that $y = x||\Pi$ and $U(\Pi, 1^t) = x$, output Π and terminate.
3. Output \perp .

We will show that the correctness of the above algorithm follows directly by Claim 18. To bound the running time of Algorithm 19, we will use the following claim.

▷ **Claim 20.** On input $(x, 1^t, 1^s)$, Algorithm 19 runs in time $2^{s-K(x)} \cdot \text{poly}(|x|, t, s)$.

Before proving Claim 20, let us use Claim 18 and Claim 20 to prove Lemma 17.

Proof of Lemma 17. We start with the correctness of Algorithm 19. Let $(x, 1^t, 1^s)$ be the input for the algorithm, and assume that $K^t(x) \leq s$. Let Π be a program of minimal length such that $U(\Pi, 1^t) = x$. By Claim 18, the correctness of the oracle A , and by a simple induction, $x||\Pi_{\leq i}$ is in the set \mathcal{S}_i for every $i \leq |\Pi| \leq s$. Therefore, $x||\Pi$ is in $\mathcal{S}_{K^t(x)}$, and thus Algorithm 19 outputs a correct answer.

By Claim 20, Algorithm 19 runs in time $2^{s-K(x)} \cdot \text{poly}(|x|, t, s)$. Finally, it is not hard to see that Algorithm 19 makes only queries of the form $(yb, 1^{q(t, |x|)}, 1^k)$ with $k = s + c_0 \log|x|$, and $|yb| \leq |x| + s$. ◀

3.1 Proving Claim 20

We will use the following lemma, that bounds the number of k -bit strings y such that xy has low Kolmogorov complexity

► **Lemma 21.** *There exists a constant $c \in \mathbb{N}$ such that the following holds for every $x \in \{0, 1\}^*$ and for every $k, \ell \in \mathbb{N}$.*

$$\left| \left\{ y \in \{0, 1\}^{\leq k} : K(xy) \leq \ell \right\} \right| \leq 2^{\ell+1-K(x)} \cdot (|x| + k)^c$$

Proof of Lemma 21. Let c be the constant from Theorem 12. By a simple counting argument, there are at most $2^{\ell+c \log(|x|+k)+1-K(x)}$ strings $y \in \{0, 1\}^{\leq k}$ such that $K(y | x) \leq \ell + c \log(|x| + k) - K(x)$. It thus enough to show that for every $y \in \{0, 1\}^{\leq k}$ with $K(y | x) > \ell + c \log(|x| + k) - K(x)$, it holds that $K(xy) > \ell$, which is true By Theorem 12. ◀

We are now ready to prove Claim 20.

Proof of Claim 20. The running time of *Algorithm 19* is bounded by $|\bigcup \mathcal{S}_i| \cdot \text{poly}(|x|, t, s)$, and thus it is enough to bound the size of $\bigcup \mathcal{S}_i$. Toward this goal, let $\tau \in \text{poly}$ be the polynomial for which A decides $\text{Gap}_\tau \text{MINKT}$. We get that

$$\begin{aligned} \left| \bigcup \mathcal{S}_i \right| &\leq \left| \left\{ y \in \{0, 1\}^{\leq s} : A(xy, 1^{q(t, |x|)}, 1^{s+c_0 \log|x|}) = \text{Yes} \right\} \right| \\ &\leq \left| \left\{ y \in \{0, 1\}^{\leq s} : K^{\tau(q(t, |x|), |x|)}(xy) \leq s + c_0 \log|x| + \log \tau(q(t, |x|), |x|) \right\} \right| \\ &\leq \left| \left\{ y \in \{0, 1\}^{\leq s} : K(xy) \leq s + c_0 \log|x| + \log \tau(q(t, |x|), |x|) \right\} \right| \\ &\leq 2^{s+c_0 \log|x|+\log \tau(q(t, |x|), |x|)+1-K(x)} \cdot \text{poly}(|x| + s) \\ &= 2^{s-K(x)} \cdot \text{poly}(|x|, t, s), \end{aligned}$$

34:12 Search-To-Decision Reductions for Kolmogorov Complexity

where the second inequality holds by the correctness of A , the third since $K(xy) \leq K^t(xy)$ and the last inequality holds by Lemma 21. \triangleleft

3.2 A List-to-Decision Reduction for K -complexity

We note that Algorithm 19 also gives “list-to-decision” reduction for Kolmogorov-complexity: given access to an oracle that decides the threshold problem of Kolmogorov-complexity, a simple variant of Algorithm 19 outputs a list of polynomial length, containing the witness.

► **Theorem 22.** *There exists an efficient oracle-aided algorithm F such that the following holds. Let A be an oracle that given $x \in \{0, 1\}^*$ and $s \in \mathbb{N}$, decides if $K(x) \leq s$. Then F^A outputs a list \mathcal{L} , such that $|\mathcal{L}| \in \text{poly}(|x|)$, and \mathcal{L} contains a K -witness for x : that is, there exists $\Pi \in \mathcal{L}$ for which $|\Pi| = K(x)$ and $U(\Pi) = x$.*

As in Theorem 15, the same holds when the oracle A only solves the gap version of the threshold problem (given x and s , A decides if $K(x) \leq s$ or $K(x) \geq s + O(\log|x|)$).

The algorithm is as follows.

► **Algorithm 23 (F).**

Oracle: A .

Input: $x \in \{0, 1\}^*$.

1. Use A to compute $s = K(x)$.
2. Set $\mathcal{S}_0 = \{x\}$ and $k = s + c_0 \log|x|$.
3. For every $i = 1, 2, \dots, s$:
 - a. Compute $\mathcal{S}_i = \{yb : y \in \mathcal{S}_{i-1}, b \in \{0, 1\} \text{ and } A(yb, 1^k) = \text{Yes}\}$
4. Output \mathcal{S}_s .

Proof of Theorem 22. Fix an input $x \in \{0, 1\}^*$, and let Π be a K -witness for x . By Claim 18 and simple induction, there exists a constant c_0 such that $\Pi \in \mathcal{S}_s$. By Lemma 21, it holds that $|\mathcal{S}_s| \in \text{poly}(|x|, K(x)) = \text{poly}|x|$.

Finally, by Lemma 21 we get that $|\mathcal{S}_i| \in \text{poly}(|x|, i)$, which implies that Algorithm 23 runs in polynomial time. \blacktriangleleft

4 Decision-to-Search Everywhere

In this part we prove our main search-to-decision reduction for GapMINKT.

► **Theorem 24.** *Let $\varepsilon > 0$ be a constant. Then there exists a randomized oracle-aided algorithm F such that the following holds for every $\tau \in \text{poly}$. Let A be an oracle that decides Gap $_\tau$ MINKT. Then $F_\tau^A = F^A(\tau, \cdot, \cdot, \cdot)$ solves search-GapMINKT.*

Moreover, on input $(x, 1^t, 1^s)$ F_τ^A runs in time $2^{\varepsilon s} \text{poly}(|x|, t, s)$, and only queries A on inputs $(y, 1^{t'}, 1^{s'})$ with $|y| = \text{poly}(|x|, t, s)$, $t' = \text{poly}(|x|, t, s)$, and $s' \leq s + \log \text{poly}(|x|, t, s)$.

Directly from Theorem 24 we get the following corollary.

► **Corollary 25.** *The following holds for every $\tau \in \text{poly}$ and $\varepsilon > 0$: Assume that for some $\alpha > 0$ there is an algorithm that solves Gap $_\tau$ MINKT in time (resp. size) $2^{\alpha \cdot s} \text{poly}(|x|, t, s)$. Then there exists an algorithm that solves search-GapMINKT in time (resp. size) $2^{(\alpha + \varepsilon)s} \cdot \text{poly}(|x|, t, s)$ on inputs $(x, 1^t, 1^s)$.*

We next prove Theorem 24. In the following, let $q \in \text{poly}$, $c_0 \in \mathbb{N}$ be the polynomial and constant from Claim 18, and let c be the constant from Theorem 12. We start with the following algorithm, that with high probability outputs a program Π such that x is a prefix of the output of Π . We later change the algorithm such that the output will be a program that outputs x .

► **Algorithm 26** (Find).

Parameters: $\epsilon > 0, \tau \in \text{poly}$

Oracle: $\text{Gap}_\tau \text{MINKT}$ decider A .

Input: $(x, 1^t, 1^s)$ for $x \in \{0, 1\}^*$, $t, s \in \mathbb{N}$.

1. Set $x^1 = x$, $t^1 = t$ and $s^1 = s$.
2. For every $j = 1, \dots, \lceil 1/\epsilon \rceil + 1$:
 - a. Set $\mathcal{S}_0^j = \{x^j\}$, and $k^j = s^j + c_0 \log |x^j|$.
 - b. Set $r^j = \epsilon s + (k^j - s^j) + c \log(|x^j| + s^j) + \log \tau(q(t^j, |x^j|), |x^j| + s^j) + \log 4/\epsilon$.
 - c. For every $i = 1, 2, \dots, s^j$:
 - i. Compute $\mathcal{S}_i^j = \{yb : y \in \mathcal{S}_{i-1}^j, b \in \{0, 1\} \text{ and } A(yb, 1^{q(t^j, |x^j|)}, 1^{k^j}) = \text{Yes}\}$
 - ii. If exists $y \in \mathcal{S}_i^j$ such that $y = x^j || \Pi$ and $U(\Pi, 1^{t^j}) = x_j$, output Π and terminate.
 - iii. If $|\mathcal{S}_i^j| \geq 2^{r^j}$, set $\mathcal{S}^j = \mathcal{S}_i^j$ and move to Item 2d.
 - d. Randomly choose $x^{j+1} \leftarrow \mathcal{S}^j$.
 - e. Set $t^{j+1} = \tau(q(t^j, |x^j|), |x^{j+1}|)$ and $s^{j+1} = k^j + \log \tau(q(t^j, |x^j|), |x^{j+1}|)$.
3. Output \perp .

We start with a simple observation on the parameters in Claim 28.

► **Claim 27.** For every $j \leq \lceil 1/\epsilon \rceil + 1$ it holds that $t^j \in \text{poly}(|x|, t, s)$, $|x^j| \in \text{poly}(|x|, t, s)$, $s^j = s + \log(\text{poly}(|x|, t, s))$, $k^j = s + \log(\text{poly}(|x|, t, s))$, and $r^j = \epsilon \cdot s + \log(\text{poly}(|x|, t, s))$.

Proof of Claim 27. The claim holds since ϵ is a constant, τ and q are polynomials, and by the definition of t^j, x^j, s^j, k^j and r^j . \triangleleft

We next bound the running time of Algorithm 26.

► **Claim 28.** On input $(x, 1^t, 1^s)$, Algorithm 26 runs in time $2^{\epsilon s} \cdot \text{poly}(|x|, t, s)$. Moreover, Algorithm 26 only queries A on inputs $(y, 1^{t'}, 1^{s'})$ with $|y| = \text{poly}(|x|, t, s)$, $t' = \text{poly}(|x|, t, s)$, and $s' \leq s + \log \text{poly}(|x|, t, s)$.

Proof of Claim 28. Fix an input $(x, 1^t, 1^s)$. Similarly to the proof of Claim 20, the running time of the j -th iteration in Step 2 of Algorithm 26 is at most

$$\left| \bigcup_i \mathcal{S}_i^j \right| \cdot \text{poly}(|x^j|, t^j, s^j) \leq s^j \cdot 2^{r^j} \cdot \text{poly}(|x^j|, t^j, s^j).$$

It thus enough to show that $t^j \in \text{poly}(|x|, t, s)$, $|x^j| \in \text{poly}(|x|, t, s)$, $s^j = s + \log(\text{poly}(|x|, t, s))$, and $r^j = \epsilon \cdot s + \log(\text{poly}(|x|, t, s))$ for every $j \leq \lceil 1/\epsilon \rceil + 1$, which holds by Claim 27. \triangleleft

To see that Algorithm 26 indeed outputs a program that outputs x , we have the following claim.

► **Claim 29.** There exists a constant $c \in \mathbb{N}$ such that the following holds. Assume that on input $(x, 1^t, 1^s)$, Algorithm 26 enters the j -th iteration in Step 2. Then,

34:14 Search-To-Decision Reductions for Kolmogorov Complexity

1. x is a prefix of x^j ,
2. $K^{t^j}(x^j) \leq s^j$, and,
3. With probability at least $1 - j \cdot \epsilon/4$, $K(x^j) \geq (j-1) \cdot \epsilon \cdot s + (s^j - s)$.

Proof of Claim 29. The first item holds by the definition of the algorithm. The second item holds since in Algorithm 26, x^j is in the set \mathcal{S}_i^{j-1} only if $A(x^j, 1^{q(t^{j-1}, |x^{j-1}|)}, 1^{k^{j-1}}) = \text{Yes}$, which implies by the correctness of A that

$$K^{\tau(q(t^{j-1}, |x^{j-1}|), |x^j|)}(x^j) \leq k^{j-1} + \log \tau(q(t^{j-1}, |x^{j-1}|), |x^j|),$$

and since $t^j = \tau(q(t^{j-1}, |x^{j-1}|), |x^j|)$, $s^j = k^{j-1} + \log \tau(q(t^{j-1}, |x^{j-1}|), |x^j|)$.

The proof of the last item is by induction on j . Assume that

$$K(x^j) \geq (j-1) \cdot \epsilon \cdot s + (s^j - s).$$

Observe that by definition of the algorithm, it holds that x^j is a prefix of every element in \mathcal{S}^j . That is, we can write $\mathcal{S}^j = x^j \|\mathcal{S}'^j$ for a set \mathcal{S}'^j of size at least 2^{r^j} .

Thus, $x^{j+1} = x^j \|\mathcal{z}$, for $\mathcal{z} \in \{0, 1\}^{\leq s^j}$ which is randomly chosen from a set of size at least 2^{r^j} . Using Lemma 9, it holds that with probability at least $1 - \epsilon/4$ that $K(\mathcal{z} \mid x^j) \geq r^j - \log 4/\epsilon$. by Symmetry of Information (Theorem 12) we get that

$$\begin{aligned} K(x^{j+1}) &\geq K(x^j) + r^j - \log 4/\epsilon - c \log(|x^j| + s^j) \\ &\geq (j-1) \cdot \epsilon \cdot s + (s^j - s) + (\epsilon s + (k^j - s^j) + \log \tau(q(t^j, |x^j|), |x^j| + s^j)) \\ &\geq j \cdot \epsilon \cdot s + (k^j + \log \tau(q(t^j, |x^j|), |x^j| + s^j)) - s \\ &\geq j \cdot \epsilon \cdot s + (k^j + \log \tau(q(t^j, |x^j|), |x^{j+1}|)) - s \\ &= j \cdot \epsilon \cdot s + s^{j+1} - s \end{aligned}$$

The proof now follows by the union bound. ◁

We can now prove Theorem 24.

Proof of Theorem 24. We start with the definition of the algorithm F . Let F be the algorithm that first executes Algorithm 26, to get a program Π such that $U(\Pi)_{\leq |x|} = x$. Then, F outputs a program Π' that simulates Π , and outputs the $|x|$ first bits of its output. It follows that $U(\Pi') = x$, and $|\Pi'| \leq |\Pi| + O(\log |x|)$. Moreover, the running time of Π' is bounded by a polynomial of the running time of Π .

Next, by Claim 28, F runs in the stated time. We now prove the correctness of F . First, recall that by Fact 7, we can assume without loss of generality that $s \leq |x| + O(1)$. Next, observe that by Claim 29, in every iteration j it holds that x is a prefix of x^j , and that $K^{t^j}(x^j) \leq s^j$. Thus, by the correctness of Algorithm 19, if for every iteration i , \mathcal{S}_i^j is not larger than 2^{r^j} , then Algorithm 26 outputs a program of length at most

$$s^j = s + \log \text{poly}(|x|, t, s) = s + \log \text{poly}(|x|, t)$$

that outputs x^j in time

$$t^j = \text{poly}(|x|, t, s) = \text{poly}(|x|, t).$$

We are left to deal with the case that in every iteration j of Step 2, \mathcal{S}_i^j is larger than 2^{r^j} for some i . In this case, by the third item of Claim 29, with probability at least $1 - (\lceil 1/\epsilon \rceil + 1)\epsilon/4 \geq 1/2$, in the last iteration we get that

$$K(x^{\lceil 1/\epsilon \rceil + 1}) \geq (\lceil 1/\epsilon \rceil + 1)\epsilon s + (s^{\lceil 1/\epsilon \rceil + 1} - s) > s^{\lceil 1/\epsilon \rceil + 1},$$

which is in contradiction to the second item of Claim 29, as $K(x^{\lceil 1/\epsilon \rceil + 1}) \leq K^t(x^{\lceil 1/\epsilon \rceil + 1})$ for every $t \in \mathbb{N}$. ◀

5 Decision-to-Search for MINKT using List Recoverable Codes

In this part we use list recoverable codes to get length-preserving decision-to-search for instances with small computational depth. List recoverable codes [3] are defined next.

► **Definition 30.** For $\Sigma = \{\Sigma_n\}_{n \in \mathbb{N}}$ and functions $m: \mathbb{N} \rightarrow \mathbb{N}$, $p: \mathbb{N} \rightarrow [0, 1]$, $\ell: \mathbb{N} \rightarrow \mathbb{N}$ and $L: \mathbb{N} \rightarrow \mathbb{N}$, an ensemble $\text{Enc} = \{\text{Enc}_n: \{0, 1\}^n \rightarrow \Sigma_n^{m(n)}\}_{n \in \mathbb{N}}$ is an efficient (p, ℓ, L) -list-recoverable code if Enc can be computed in polynomial time, and there exists an efficient procedure Rec such that the following holds for every $n \in \mathbb{N}$. Given sets $\mathcal{S}_1, \dots, \mathcal{S}_{m(n)} \subseteq \Sigma_n$ such that $|\mathcal{S}_i| \leq \ell(n)$ for every i , $\text{Rec}(\mathcal{S}_1, \dots, \mathcal{S}_{m(n)})$ outputs a list \mathcal{R} of size at most $L(n)$, containing all $x \in \{0, 1\}^n$ with

$$|\{i \in [m(n)]: \text{Enc}(x)_i \notin \mathcal{S}_i\}| \leq p(n) \cdot m(n).$$

As shown in [5, 4], Reed-Solomon codes [19] are list recoverable with parameters that are suitable for our needs. In particular:

► **Theorem 31** ([5, 4]). For every efficiently computable $w \in O(\log n)$, there exists an efficient $\text{Enc}: \{0, 1\}^n \rightarrow [m(n)]^{m(n)}$, for $m(n) = 2^{w(n)}$, such that Enc is an (p, ℓ, L) list recoverable code, for any $p \leq 1 - \sqrt{\ell(n) \cdot n/m(n)}$ and $L(n) \in O(\ell(n) \cdot m(n))$.

Moreover, $\text{Enc}: \{0, 1\}^n \rightarrow [m(n)]^{m(n)}$ runs in time $\text{poly}(n, w(n))$.

We now state the main theorem of this part.

► **Theorem 32.** For every $d \in \mathbb{N}$, there exists an oracle-aided algorithm F such that the following holds. Let A be an oracle that decides MINKT. Then F^A solves search-MINKT on inputs $(x, 1^t, 1^s)$ with $cd^t(x) \leq d \log |x|$.

Moreover, on input $(x, 1^t, 1^s)$, F runs in time $\text{poly}(|x|, t, s)$, and only queries A on inputs $(x||y, 1^{t'(|x|, t)}, 1^{s'})$ with $|y| = r(|x|, t) \leq \log \text{poly}(|x|, t)$, $t' \in \text{poly}$ and $s' \leq s + \log \text{poly}(|x|, t)$.

In the following, let $\text{Enc} = \{\text{Enc}_n: \{0, 1\}^n \rightarrow \Sigma_n^{m(n)}\}$ be an efficient (p, ℓ, L) list recoverable code, and Rec the efficient reconstruction algorithm of Enc , for parameters $\Sigma_n, m(n), p, \ell, L$ we will choose later. Let $q \in \text{poly}$ and $c_0, c_1 \in \mathbb{N}$ be a polynomial and constants to be chosen later. We will show that the following algorithm returns a short program that produces x , when the input s is exactly equal to $K^t(x)$, and then show how to get rid of this assumption. For $i \in [m(n)]$, we let $\langle i \rangle \in \{0, 1\}^{\lceil \log m(n) \rceil}$ be the binary representation of i . Let c_K be the constant from Fact 7 such that $K^t(x) \leq |x| + c_K$ for every x .

► **Algorithm 33** (Find).

Oracle: MINKT decider A .

Parameters: $\tau \in \text{poly}$, $d \in \mathbb{N}$.

Input: $(x, 1^t, 1^s)$ for $x \in \{0, 1\}^*$, $t, s \in \mathbb{N}$.

1. Let $n = |x| + c_K$.
2. For every $i \in [m(n)]$, compute the set

$$\mathcal{S}_i = \left\{ y \in \Sigma_n: A(x||\langle i \rangle||y, 1^{q(t, |x|)}, 1^{s+\log m(n)+c_1 \log \log m(n)}) = \text{Yes} \right\}.$$

3. For every $i \in [m(n)]$ such that $|\mathcal{S}_i| > \ell(n)$, set $\widehat{\mathcal{S}}_i = \emptyset$. Otherwise, set $\widehat{\mathcal{S}}_i = \mathcal{S}_i$.
4. Compute $\text{Rec}(\widehat{\mathcal{S}}_1, \dots, \widehat{\mathcal{S}}_{m(n)})$, and let $\mathcal{R} \subseteq \{0, 1\}^n$ be the output.
5. Find a string $\Pi \in \{0, 1\}^s$ such that $\Pi||0^{n-s} \in \mathcal{R}$ and $U(\Pi, 1^t) = x$.
6. Output Π .

34:16 Search-To-Decision Reductions for Kolmogorov Complexity

Note that in the above algorithm, the set \mathcal{S}_i is the set of all possible values (in Σ_n) for the i th symbol in the encoding of Π . We start by showing that the size of $\mathcal{S}_1, \dots, \mathcal{S}_{m(n)}$ is not too large.

▷ **Claim 34.** There exists a constant $c_2 \in \mathbb{N}$ such that the following holds for every $c_1, d, w \in \mathbb{N}$. Let $x \in \{0, 1\}^*$ such that $cd^t(x) \leq d \log|x|$. Then there are at most

$$M = 2^{w+(c_1+c_2)(\log 2w)+d \log|x|+c_2 \log|x|}$$

pairs $(i, \alpha) \in [2^w] \cdot [2^w]$ such that $K(x||\langle i \rangle||\alpha) \leq K^t(x) + w + c_1(\log w)$.

Proof. Immediate by Lemma 21. ◁

Next, we will use the following claim to show that Π is in the set \mathcal{R} .

▷ **Claim 35.** For every $w: \mathbb{N} \rightarrow \mathbb{N}$, and every efficient code $\text{Enc}: \{0, 1\}^n \rightarrow [2^{w(n)}]^{2^{w(n)}}$, there exists a polynomial q such that the following holds for every $x \in \{0, 1\}^*$ with $w(|x|) \geq 2$, and every $t \in \mathbb{N}$. Let Π a program of length $K^t(x)$ such that $U(\Pi, 1^t) = x$. Then for $n = |x| + c_K$ and every $i \in [2^{w(n)}]$,

$$K^{q(t, |x|)}(x||\langle i \rangle||\text{Enc}(\Pi||0^{n-|\Pi|})_i) \leq K^t(x) + w(n) + c_1 \log w(n)$$

for some universal constant c_1 .

Proof. Let Π' be the program that given input (Π, i) , first simulates $U(\Pi)$ to get an output x , and then computes $n = |x| + c_K$ and $\text{Enc}(\Pi||0^{n-|\Pi|})_i$. Let $q \in \text{poly}$ be the bound of the running time of Π' . Then, using Fact 8, $K^{q(t, |x|)}(x||\langle i \rangle||\text{Enc}(\Pi)_i) \leq K^t(x) + w(n) + 4 \log w(n)$. ◁

We are now ready to prove Theorem 32.

Proof of Theorem 32. We will show that on input $(x, 1^t, 1^{s'})$ Algorithm 33 returns a program Π such that $U(\Pi, 1^t) = x$, if $s' = K^t(x)$. The theorem then follows by considering the algorithm that enumerates over all possible values of $s' < s$.

Let c_1 and c_2 be the constants promised by Claim 35 and Claim 34 respectively. Let $\text{Enc}: \{0, 1\}^n \rightarrow [2^{w(n)}]^{m(n)}$ be an efficient (p, ℓ, L) list recoverable code, for $w(n) = 2(d + c_1 + c_2) \log n + 10$, $p(n) = 1/10$, $\ell(n) = 2^{(c_1+c_2) \log w(n) + (d+c_2) \log n + 5}$, $L(n) \in \text{poly}$ and $m(n) = 2^{w(n)}$. By Theorem 31 such a code exists as

$$\ell(n) \cdot n/m(n) = n \cdot 2^{(c_1+c_2) \log w(n) - (d+2c_1+c_2) \log n - 5} \leq 2^{-5} \leq (1-p)^2$$

for any large enough n . Finally, let q be the polynomial promised by Claim 35 with respect to the code Enc .

We next show that when using Enc as the code in Algorithm 33, Algorithm 33 outputs a minimal program Π that produces the input x . By the list recoverable property of Enc , it is enough to show that $\Pi||0^{n-s}$ is in the list outputted by Rec . It thus enough to show that (1) $\text{Enc}(\Pi)_i \in \mathcal{S}_i$ for every $i \in [m(n)]$ and (2), $\mathcal{S}_i = \widehat{\mathcal{S}}_i$ for at least $(1 - 1/10)m(n)$ indexes $i \in [m(n)]$.

(1) follows immediately by Claim 35 and the definition of \mathcal{S}_i . For (2), by Claim 34 the total size $\sum_i |\mathcal{S}_i|$ of the sets \mathcal{S}_i is at most

$$2^{w+(c_1+c_2) \log 2w+d \log n+c_2 \log n} = m(n) \cdot \ell(n) \cdot 2^{-5}.$$

By Markov, we get that there are at most $2^{-5} \cdot m(n) < 1/10 \cdot m(n)$ indexes i such that $|\mathcal{S}_i| > \ell(n)$. For all other i 's it holds that $\mathcal{S}_i = \widehat{\mathcal{S}}_i$, which concludes the proof. ◀

5.1 Decision-to-Search on Average

► **Definition 36.** For a function $t: \mathbb{N} \rightarrow \mathbb{N}$, we say that an algorithm A computes K^t if for every $x \in \{0, 1\}^*$, $A(x) = K^{t(|x|)}(x)$. We say that A computes K^t with error ϵ if for every $n \in \mathbb{N}$, $\Pr_{x \leftarrow \{0, 1\}^n} [A(x) = K^{t(|x|)}(x)] \geq 1 - \epsilon(n)$.

We say that A solves search- K^t with error ϵ if A outputs a program Π such that $U(\Pi, 1^{t(|x|)}) = x$ and $|\Pi| = K^{t(|x|)}(x)$ with probability $1 - \epsilon(n)$ over $x \leftarrow \{0, 1\}^n$.

We prove the following theorem.

► **Theorem 37.** For every $p, t \in \text{poly}$ there exists $\hat{p} \in \text{poly}$ and $t' \in \text{poly}$, and an efficient oracle-aided algorithm F such that the following holds. Let A be an that computes $K^{t'}$ with error $1/\hat{p}$. Then F^A solves search- K^t with error $1/p$.

Moreover, on input $x \in \{0, 1\}^n$, F makes only queries of the form $x||y$ with $y \in O(\log n)$.

To prove Theorem 37, we will use the following theorem, which is followed by the same proof as Theorem 32.

► **Theorem 38.** For every $d \in \mathbb{N}$ and $t \in \text{poly}$, there exists $t' \in \text{poly}$ and an oracle-aided algorithm F such that the following holds. Let A be an oracle that computes $K^{t'}$. Then F^A solves search- K^t on inputs $(x, 1^t, 1^s)$ with $cd^t(x) \leq d \log|x|$.

Moreover, on input x , F runs in time $\text{poly}(|x|)$, and only queries A on inputs $x||y$ with $|y| = r(|x|, t) \leq \log \text{poly}(|x|, t)$.

Proof. This follows by a similar proof to Theorem 32, together with the observation that MINKT can be decided on inputs $(x, 1^{t(|x|)}, 1^s)$ using oracle that computes K^t . ◀

Proof of Theorem 37. We start with the assumption that the oracle A is deterministic, and later show how to eliminate this assumption. Fix p and t , and let t' and F be the polynomial and algorithm promised by Theorem 32. Let r be the parameters of F as defined in Theorem 32, and let d be such that $|x|^d > 2p(|x|)$ for every x with $|x| > 2$. Finally, let $\hat{p}(|x|) = 8p(|x|) \cdot 2^{r(|x|, t(|x|))}$, and let A be an oracle the computes $K^{t'}$ with error $1/\hat{p}$ (when the probability is taken only over the input x of A).

Let \hat{A} be an oracle that computes $K^{t'}$ correctly on every input. By Definition 36, on a random $x \leftarrow \{0, 1\}^n$ A and \hat{A} agree with probability $1 - 1/\hat{p}(x)$. By Theorem 32, $F^{\hat{A}}$ solves search- K^t on every input with $cd^{t(|x|)}(x) \leq d \log|x|$, and thus, by Fact 11, $F^{\hat{A}}$ solves search- K^t with error at most $1/|x|^d \leq 1/(2p(|x|))$. It is thus enough to show that

$$\Pr_{x \leftarrow \{0, 1\}^n} [F^A(x) = F^{\hat{A}}(x)] \geq 1 - 1/2p(|x|). \quad (1)$$

To see Equation (1), let \mathcal{B}_n be the set of all $x \in \{0, 1\}^n$ such that $F^A(x) \neq F^{\hat{A}}(x)$. By definition of \mathcal{B} , on every input $x \in \mathcal{B}_n$, there must be some query $x||y$ that F makes to the oracle, such that $A(x||y) \neq \hat{A}(x||y)$. Let $\mathcal{R}_n \subseteq \{0, 1\}^n$ be the set of all x 's, such that there exists $y \in \{0, 1\}^{r(n, t(n))}$ with $A(x||y) \neq \hat{A}(x||y)$. We get that $|\mathcal{R}_n| \geq |\mathcal{B}_n|$, and,

$$1/\hat{p}(n + r(n, t(n))) \geq \Pr_{z \leftarrow \{0, 1\}^{n+r(n, t(n))}} [A(z) \neq \hat{A}(z)] \geq |\mathcal{R}_n| \cdot 2^{-n-r(n, t(n))}.$$

Thus, we get that

$$|\mathcal{B}_n| \leq 2^{n+r(n, t(n))+1} \cdot 1/\hat{p}(n + r(n, t(n))) \leq 2^{n+r(n, t(n))+1} \cdot 1/\hat{p}(n),$$

which implies that $|\mathcal{B}_n|/2^n \leq 1/4p(n)$, which concludes the claim.

We next move to deal with randomized algorithm A . That is, A that err with probability $1/\widehat{p}$, where the probability is now taken both over the input and the internal randomness of A . To deal with such a randomized oracle A , we observe that by standard amplification, it is enough to replace Equation (1), with $\Pr_{x \leftarrow \{0,1\}^n} \left[\Pr \left[F^A(x) \neq F^{\widehat{A}}(x) \right] > 1/2 \right] \leq 1/4p(|x|)$. We can thus let \mathcal{B}_n be the set of all $x \in \{0,1\}^n$ such that $\Pr \left[F^A(x) \neq F^{\widehat{A}}(x) \right] > 1/2$, and $\mathcal{R}_n \subseteq \{0,1\}^n$ be the set of all x 's, such that with probability larger than $1/2$ (over the randomness of A) there exists $y \in \{0,1\}^{r(n,t(n))}$ with $A(x||y) \neq \widehat{A}(x||y)$. We get that $|\mathcal{R}_n| \geq |\mathcal{B}_n|$, and,

$$1/\widehat{p}(n+r(n,t(n))) \geq \Pr_{z \leftarrow \{0,1\}^{n+r(n,t(n))}} \left[A(z) \neq \widehat{A}(z) \right] \geq 1/2 \cdot |\mathcal{R}_n| \cdot 2^{-n-r(n,t(n))},$$

and the claim follows as in the deterministic case. \blacktriangleleft

We get the following two corollaries. The first was already proven in [16].

► **Corollary 39** (reproving [16]). *For every $p, t \in \text{poly}$ there exists $\widehat{p}, t' \in \text{poly}$ such that the following holds:*

- *Assume that there is a poly-time (resp. poly size) algorithm that computes $K^{t'}$ with error $1/\widehat{p}$. Then there exists a poly-time (resp. poly-size) algorithm that solves search- K^t with error $1/p$.*
- *Assume that there is an algorithm that computes $K^t(x)$ with error $1/\widehat{p}$ in time (resp. size) $2^{o(|x|)}$. Then there exists an algorithm that solves search- K^t with error $1/p$ in time (resp. size) $2^{o(|x|)}$.*

The second shows that the same holds also in the exponential regime.

► **Corollary 40.** *For every $p, t \in \text{poly}$ there exists $\widehat{p}, t' \in \text{poly}$ such that the following holds: Assume that there exists a constant $\alpha > 0$ and an algorithm that computes $K^{t'}$ with error $1/\widehat{p}$ in time (resp. size) $2^{\alpha \cdot |x|} \text{poly}(|x|)$. Then there exists an algorithm that solves search- K^t with error $1/p$ in time (resp. size) $2^{\alpha \cdot |x|} \text{poly}(|x|)$.*

6 Decision-to-Search for GapMINKT using List Recoverable Codes

In this part we show that Theorem 32 holds even when we start with an oracle that solves GapMINKT instead of MINKT.

► **Theorem 41.** *For every $\tau \in \text{poly}$ and $d \in \mathbb{N}$, there exists an oracle-aided algorithm F such that the following holds. Let A be an oracle that decides Gap $_{\tau}$ MINKT. Then F^A solves search-MINKT on inputs $(x, 1^t, 1^s)$ with $cd^t(x) \leq d \log|x|$.*

Moreover, on input $(x, 1^t, 1^s)$, F runs in time $\text{poly}(|x|, t, s)$, and only queries A on inputs $(x||y, 1^{t'(|x|, t)}, 1^{s'})$ with $|y| = \ell(|x|, t) \leq \log \text{poly}(|x|, t)$, $t' \in \text{poly}$ and $s'(|x|, t) \leq s + \log \text{poly}(|x|, t)$.

To prove Theorem 41 we will show that Algorithm 33 with different parameters works. When the oracle A only decides GapMINKT, the size of the sets \mathcal{S}_i can be larger. The first claim shows that it still cannot be too large.

▷ **Claim 42.** There exists a constant $c_2 \in \mathbb{N}$ such that the following holds for every $c_1, d, w \in \mathbb{N}$, and every $\tau, q \in \text{poly}$. Let $x \in \{0,1\}^*$ such that $cd^t(x) \leq d \log|x|$. Then there are at most

$$M = 2^{w+(c_1+c_2)(\log 2w)+d \log|x|+c_2 \log|x|+\log \tau(q(t,|x|),|x|)}$$

pairs $(i, \alpha) \in [2^w] \cdot [2^w]$ such that $K(x||i||\alpha) \leq K^t(x) + w + c_1(\log w) + \log \tau(q(t, |x|), |x|)$.

Proof. Immediate by Lemma 21. \triangleleft

To get non-trivial bound from Claim 42, we will need to choose out code alphabet size, 2^w , to be larger than $\tau(q(t, |x|), |x|)$. Since the polynomial q is going to be dependent on the code evaluation time, we will need to use codes where the running time of the code does not grow too fast with the alphabet size. By Theorem 31, there is a family of list recoverable codes, such that we can choose the size of the output alphabet Σ , and the running time of the encoder only depends on $\log|\Sigma|$. This property is used in the next claim, which shows that $\Pi|0^{n-s}$ is in the set \mathcal{R} .

▷ **Claim 43.** There exists a constant c_1 and polynomial q' , such that the following holds for every $w: \mathbb{N} \rightarrow \mathbb{N}$, every $x \in \{0, 1\}^*$ with $|x| \geq 16$, and every $t \in \mathbb{N}$. Let Π a program of length $K^t(x)$ such that $U(\Pi, 1^t) = x$, and let $\text{Enc}: \{0, 1\}^n \rightarrow [2^{w(n)}]^{2^{w(n)}}$ be the code from Theorem 31. Then for $n = |x| + c_K$ and for every $i \in [2^{w(n)}]$,

$$K^{q'(t, |x|, w(n))}(x| \langle i \rangle | \text{Enc}(\Pi|0^{n-|\Pi|})_i) \leq K^t(x) + w(n) + c_1 \log w(n).$$

Proof. The proof is similar to the proof of Claim 35, using the observation that the running time of Π' is polynomial in $t, |x|$ and $w(n)$ by the choice of Enc . ◀

We are now ready to prove Theorem 41.

Proof of Theorem 41. We will show that on input $(x, 1^t, 1^{s'})$ Algorithm 33 returns a program Π such that $U(\Pi, 1^t) = x$, if $s' = K^t(x)$. The theorem then follows by considering the algorithm that enumerates over all possible values of $s' < s$.

Let c_1 and q' be the constant and polynomial promised by Claim 43, and let c_2 be the constant from Claim 42. Let c_0 be such that for $w(n) = c_0 \lceil \log n + t(n) \rceil$,

$$\begin{aligned} & w(|x| + c_K) \\ & \geq 2((c_1 + c_2) \log w(|x| + c_K) + (d + c_2) \log |x| \\ & \quad + \log \tau(q'(t(|x|), n, w(|x| + c_K)), |x| + 2w(|x| + c_K))), \end{aligned} \tag{2}$$

and, $2^{w(|x|+c_K)/3} > |x|$. Finally, let $q(t, |x|) = q'(t, |x|, w)$.

The proof now continues similar to the proof of Theorem 32. ◀

References

- 1 Luis Antunes, Lance Fortnow, Dieter Van Melkebeek, and N Variyam Vinodchandran. Computational depth: concept and applications. *Theoretical Computer Science*, 354(3):391–404, 2006.
- 2 Gregory J. Chaitin. On the simplicity and speed of programs for computing infinite sets of natural numbers. *J. ACM*, 16(3):407–422, 1969.
- 3 Venkatesan Guruswami and Piotr Indyk. Expander-based constructions of efficiently decodable codes. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 658–667. IEEE, 2001.
- 4 Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory. *Draft available at <https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/web-coding-book.pdf>*, 2(1), 2012.
- 5 Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 28–37. IEEE, 1998.
- 6 J. Hartmanis. Generalized kolmogorov complexity and the structure of feasible computations. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, pages 439–445, 1983. doi:10.1109/SFCS.1983.21.

- 7 Johan Hastad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, pages 1364–1396, 1999.
- 8 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within np. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 247–258. IEEE, 2018.
- 9 Shuichi Hirahara. Non-disjoint promise problems from meta-computational view of pseudorandom generator constructions. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
- 10 Shuichi Hirahara. Symmetry of information from meta-complexity. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
- 11 Shuichi Hirahara, Rahul Ilango, and Ryan Williams. Beating brute force for compression problems. Technical Report TR23-171, Electronic Colloquium on Computational Complexity, 2023.
- 12 Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 73–79, 2000.
- 13 Ker-I Ko. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986. doi:10.1016/0304-3975(86)90081-2.
- 14 A. N. Kolmogorov. Three approaches to the quantitative definition of information. *International Journal of Computer Mathematics*, 2(1-4):157–168, 1968.
- 15 Leonid A. Levin. Universal’nyĕ perebornyĕ zadachi (Universal search problems : in Russian). *Problemy Peredachi Informatsii*, pages 265–266, 1973.
- 16 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. IEEE, 2020.
- 17 Noam Mazor and Rafael Pass. The non-uniform perebor conjecture for time-bounded kolmogorov complexity is false. *15th Innovations in Theoretical Computer Science*, 2024.
- 18 Noam Mazor and Rafael Pass. Search-to-decision reductions for kolmogorov1 complexity. *Electronic Colloquium on Computational Complexity*, 2024.
- 19 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.
- 20 Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 330–335, 1983.
- 21 R.J. Solomonoff. A formal theory of inductive inference. part i. *Information and Control*, 7(1):1–22, 1964. doi:10.1016/S0019-9958(64)90223-2.
- 22 Boris A Trakhtenbrot. A survey of russian approaches to perebor (brute-force searches) algorithms. *Annals of the History of Computing*, 6(4):384–400, 1984.
- 23 Sergey Yablonski. The algorithmic difficulties of synthesizing minimal switching circuits. *Problemy Kibernetiki*, 2(1):75–121, 1959.
- 24 Sergey V Yablonski. On the impossibility of eliminating perebor in solving some problems of circuit theory. *Doklady Akademii Nauk SSSR*, 124(1):44–47, 1959.
- 25 A. C. Yao. Protocols for secure computations. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 160–164, 1982.
- 26 Alexander K Zvonkin and Leonid A Levin. The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms. *Russian Mathematical Surveys*, 25(6):83, 1970.

Finer-Grained Hardness of Kernel Density Estimation

Josh Alman  

Department of Compute Science, Columbia University, New York, NY, USA

Yunfeng Guan  

Department of Compute Science, Columbia University, New York, NY, USA

Abstract

In batch Kernel Density Estimation (KDE) for a kernel function $f : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$, we are given as input $2n$ points $x^{(1)}, \dots, x^{(n)}, y^{(1)}, \dots, y^{(n)} \in \mathbb{R}^m$ in dimension m , as well as a vector $v \in \mathbb{R}^n$. These inputs implicitly define the $n \times n$ kernel matrix K given by $K[i, j] = f(x^{(i)}, y^{(j)})$. The goal is to compute a vector $w \in \mathbb{R}^n$ which approximates Kw , i.e., with $\|Kw - v\|_\infty < \varepsilon \|w\|_1$. For illustrative purposes, consider the Gaussian kernel $f(x, y) := e^{-\|x-y\|_2^2}$. The classic approach to this problem is the famous Fast Multipole Method (FMM), which runs in time $n \cdot O(\log^m(\varepsilon^{-1}))$ and is particularly effective in low dimensions because of its exponential dependence on m . Recently, as the higher-dimensional case $m \geq \Omega(\log n)$ has seen more applications in machine learning and statistics, new algorithms have focused on this setting: an algorithm using discrepancy theory, which runs in time $O(n/\varepsilon)$, and an algorithm based on the polynomial method, which achieves inverse polynomial accuracy in almost linear time when the input points have bounded square diameter $B < o(\log n)$.

A recent line of work has proved fine-grained lower bounds, with the goal of showing that the “curse of dimensionality” arising in FMM is necessary assuming the Strong Exponential Time Hypothesis (SETH). Backurs et al. [NeurIPS 2017] first showed the hardness of a variety of Empirical Risk Minimization problems including KDE for Gaussian-like kernels in the case with high dimension $m = \Omega(\log n)$ and large scale $B = \Omega(\log n)$. Alman et al. [FOCS 2020] later developed new reductions in roughly this same parameter regime, leading to lower bounds for more general kernels, but only for very small error $\varepsilon < 2^{-\log^{\Omega(1)}(n)}$.

In this paper, we refine the approach of Alman et al. to show new lower bounds in all parameter regimes, closing gaps between the known algorithms and lower bounds. For example:

- In the setting where $m = C \log n$ and $B = o(\log n)$, we prove Gaussian KDE requires $n^{2-o(1)}$ time to achieve additive error $\varepsilon < \Omega(m/B)^{-m}$, matching the performance of the polynomial method up to low-order terms.
- In the low dimensional setting $m = o(\log n)$, we show that Gaussian KDE requires $n^{2-o(1)}$ time to achieve ε such that $\log \log(\varepsilon^{-1}) > \tilde{\Omega}((\log n)/m)$, matching the error bound achievable by FMM up to low-order terms. To our knowledge, no nontrivial lower bound was previously known in this regime.

Our approach also generalizes to any parameter regime and any kernel. For example, we achieve similar fine-grained hardness results for any kernel with slowly-decaying Taylor coefficients such as the Cauchy kernel. Our new lower bounds make use of an intricate analysis of the “counting matrix”, a special case of the kernel matrix focused on carefully-chosen evaluation points. As a key technical lemma, we give a novel approach to bounding the entries of its inverse by using Schur polynomials from algebraic combinatorics.

2012 ACM Subject Classification Theory of computation \rightarrow Problems, reductions and completeness

Keywords and phrases Kernel Density Estimation, Fine-Grained Complexity, Schur Polynomials

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.35

Funding Supported in part by a grant from the Simons Foundation (Grant Number 825870 JA) and a Google Research Scholar award.

Acknowledgements We would like to thank Amol Aggarwal for constructive discussions on Schur polynomials, and anonymous reviewers for helpful suggestions.



© Josh Alman and Yunfeng Guan;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 35; pp. 35:1–35:21

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



1 Introduction

In computational statistics and learning theory, many applications reduce to solving the following problem: Given a set of points $X \subseteq \mathbb{R}^m$ sampled from some unknown distribution \mathcal{D} , estimate the probability density at a query point $y \in \mathbb{R}^m$ (or multiple such points $Y \subseteq \mathbb{R}^m$). This problem is known as the density estimation problem and has attracted interest in theoretical computer science in recent years. One of the most common methods for this problem is Kernel Density Estimation (KDE), which tries to approximate the distribution by a sum of kernel functions centered at each data point x . More concretely, if a kernel function $k : \mathbb{R}^m \times \mathbb{R}^m \rightarrow [0, 1]$ is appropriately picked, then the *Kernel Density* $\text{KDF}(y) := \frac{1}{n} \sum_{x \in X} k(x, y)$ is a reasonably good approximation of \mathcal{D} at point y .¹ In this work, we will focus on the popular class of radial kernels, i.e., kernels of the form $K(x, y) = f(\|x - y\|_2^2)$ for some function $f : \mathbb{R}_{\geq 0} \rightarrow [0, 1]$. Some prominent radial kernels include the Gaussian kernel with $f(u) = e^{-u}$, Rational Quadratic kernel with $f(u) = 1/(1 + u)^\sigma$ for constant σ , and t -Student kernel with $f(u) = 1/(1 + u^\rho)$ for constant ρ .

By virtue of its excellent statistical properties, Kernel Density Estimation has found numerous applications in computational statistics for tasks like mean estimation, classification, and outlier detection; see, for instance, the surveys [18, 13] and [24, Chapter 1]. The prevalence of kernel methods in machine learning has also led to many new applications of Kernel Density Estimation [22]. One popular recent example is in “attention computation”, the time bottleneck in computations involving transformers and other large language models; this is known to be essentially equivalent to Kernel Density Estimation with the Gaussian kernel [3]².

In light of its wide applicability, tremendous effort has also been put into designing efficient algorithms for the KDE problem. The straightforward algorithm shows that for a given y , $\text{KDF}(y)$ can be computed in $O(n)$ time, assuming f is efficiently computable.³ If one aims at an exact result, this simple algorithm seems to be optimal. However, a number of advances starting from the celebrated Fast Multipole Method [14] have successfully brought the running time down to $n^{o(1)}$ in various parameter regimes (after preprocessing the set X), provided an approximation to the Kernel Density suffices. Before introducing these algorithmic ideas, we first give a formal definition of the (approximate) KDE problem. For simplicity, we present here the batched version, which asks to compute the KDF value for a collection of query points simultaneously. (We also compare the batched version and the data structure version in Section 1.5.)

► **Definition 1** ((Approximate) Kernel Density Estimation $\text{KDE}_f(n, m, \varepsilon, B)$). *Let $f : [0, B] \rightarrow [0, 1]$ be a real function and define the (kernel) function $k(x, y) = f(\|x - y\|_2^2)$.*

Suppose we are given as input $2n$ points $x^{(1)}, \dots, x^{(n)}, y^{(1)}, \dots, y^{(n)} \in \mathbb{R}^m$ with the guarantee that $\|x^{(i)} - y^{(j)}\|_2^2 \leq B$ for all $i, j \in [n]$. Define the kernel matrix $K \in [0, 1]^{n \times n}$ by $K[i, j] = f(\|x^{(i)} - y^{(j)}\|_2^2)$ for $i, j \in [n]$. Then given additionally a vector $u \in \mathbb{R}^n$, one needs to output a vector $v \in \mathbb{R}^n$ such that $\|v - Ku\|_\infty \leq \varepsilon \|u\|_1$.

¹ For example, when $m = 1$ and $k(x, y) = \mathbb{1}[|x - y| \leq 1]$, the KDF is the histogram of the dataset X .

² The parameters of KDE correspond directly to the parameters in attention: n is the number of “tokens” in the query sequence, and m is the dimension of the vectors which encode tokens.

³ In this work we always assume $f(\|x - y\|_2^2)$ can be exactly computed in $O(1)$ time. Most of the previous works adopt this assumption.

1.1 Algorithms

As a computational problem, KDE has its running time dependent on four parameters: the number of data points / query points n , the dimension of vectors m , the *additive* error of approximation ε , and the maximum pairwise (square) distance B . Multiple parameter regimes arise from their interplay, and the KDE problem tends to have rather different behavior across the regimes. Of course, the choice of the kernel function can also substantially change the complexity of the problem; in the following discussion we take the Gaussian kernel as a running example.

Low dimensional KDE: $m = o(\log n)$. The first nontrivial algorithm for the KDE problem is Greengard and Roklin's Fast Multipole Method [14]. In this method, one partitions the space into bounded regions, and then Taylor expands the kernel around centers of these regions. In this way a truncation of the Taylor series yields a approximation of high accuracy when the data points lie in distant cells. For the Gaussian kernel, the Fast Multipole Method runs in time $O(n \log^{O(m)}(n/\varepsilon))$, which is exceptionally good in low dimensions (e.g., $m = O(1)$ in the original physical context⁴ of [15]). However, in higher dimensional regimes, this approach suffers from an exponential dependence on m , which is inherent in the (deterministic) space partitioning procedure (essentially building a quad-tree), and shared by other tree-based methods.

Moderate dimensional KDE: $m = \Theta(\log n)$. One way to avoid the exponential dependence on m of the Fast Multipole Method is to use the method of polynomial approximation directly, without combining it with space partitioning. Alman and Aggarwal [1] pinned down the optimal degree of a polynomial that approximates $f(u) = e^{-Bu}$ over $u \in [0, 1]$ by analyzing its Chebyshev truncation. A polynomial approximation of e^{-Bu} then allows one to approximate the Gaussian kernel matrix using a matrix consisting of only polynomial entries. Such a matrix admits a decomposition as a product of two matrices of dimension $n \times n^{o(1)}$, for which the matrix-vector product can be performed efficiently. As a result, they gave an algorithm for Gaussian KDE running in $n^{1+o(1)}$ time when $m = O(\log n)$, $B = o(\log n)$ and $\varepsilon = 1/\text{poly}(n)$.

High dimensional KDE: $m = \omega(\log n)$. Another technique often used in the study of KDE is random sampling. For example, to compute $\frac{1}{n} \sum_{x \in X} k(x, y)$, one can sample a subset $S \subseteq X$ so that $\frac{1}{|S|} \sum_{x \in S} k(x, y)$ is a close approximation to $\text{KDF}(y)$ when $|S|$ is sufficiently large. Simple calculation shows that $|S| = O(\log n/\varepsilon^2)$ suffices, and a $\tilde{O}(n/\varepsilon^2)$ time (randomized) algorithm follows. We note that this algorithm has no dependence on m and works for arbitrarily high dimensions. This folklore random sampling algorithm stood unchallenged until recently Phillips and Tai [20] devised an $O(n/\varepsilon)$ -time algorithm by building a small coresets based on discrepancy theory. They show that a clever subsampling scheme yields a smaller $S \subseteq X$ which has the same accuracy as its counterpart when used in the random sampling algorithm. We in addition note that much effort [10, 11, 5, 7, 8, 9] has been dedicated to the *relative* error setting, combining sampling schemes with techniques from high-dimensional geometry (such as hashing-based space partitioning). See Section 1.5 below where we compare the additive error and relative error settings and survey these algorithms in more detail.

⁴ The Fast Multipole Method was originally introduced to solve the n -body problem from physics.

1.2 Lower bounds

Despite the great variety of algorithms developed over the years, we still lack a comprehensive understanding of the complexity of the KDE problem. In this work, our goal is to complement the distinct algorithms targeting different parameter regimes with (nearly) matching running time lower bounds, and explain how the complexity of KDE is affected by parameters.

The first and most influential known lower bound was developed by Backurs, Indyk and Schmidt [6]. This work establishes a reduction from the Bichromatic Closest Pair (BCP) problem to a collection of empirical risk minimization problems including KDE. As BCP is a standard hard problem in fine-grained complexity assuming the Strong Exponential Time Hypothesis (SETH), the reduction leads to conditional lower bounds on the running time of KDE. To explain in detail, we first give the formal statement of the BCP problem.

► **Problem 2** (Bichromatic Closest Pair). *Hamming-BCP(n, m): Given two sets $X = \{x^{(1)}, \dots, x^{(n)}\}, Y = \{y^{(1)}, \dots, y^{(n)}\} \in \{0, 1\}^m$, compute $\min_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2$.*

We then observe that terms in the KDE result $\|K \times \mathbf{1}\|_1$ can be grouped according to the pairwise distance between $x^{(i)}$ and $y^{(j)}$.

$$\|K \times \mathbf{1}\|_1 = \sum_{i=1}^n \sum_{j=1}^n f(\|x^{(i)} - y^{(j)}\|_2^2) = \sum_{p=0}^m f(p) \cdot \#\{(i, j) \in [n]^2 : \|x^{(i)} - y^{(j)}\|_2^2 = p\}.$$

This identity provides a way of extracting minimum pairwise distance from the KDE result. Indeed, if $\min_{i, j} \|x^{(i)} - y^{(j)}\|_2^2 \geq p+1$, then $\|K \times \mathbf{1}\|_1 \leq n^2 f(p+1)$ (assuming f is decreasing); otherwise, there exists a pair $(x^{(i)}, y^{(j)})$ with $\|x^{(i)} - y^{(j)}\|_2^2 \leq p$ and $\|K \times \mathbf{1}\|_1 \geq f(p)$. In this way, if f is decreasing quickly enough, one can decide whether $\min_{i, j} \|x^{(i)} - y^{(j)}\|_2^2 \leq p$ based on a sufficiently accurate approximation of $\|K \times \mathbf{1}\|_1$.

This relatively simple reduction gives strong running time lower bounds in the moderate/high dimensional regime. In the original paper, [6] shows that when $m = \Omega(\log n)$ and $B = \Omega(\log n)$, it requires $n^{2-o(1)}$ time to approximate the Gaussian KDE to $\varepsilon = 2^{-\text{poly} \log n}$. This result is later improved by Alman and Aggarwal [1] (also in [11] for KDE with relative error), who combine the same reduction with the hardness result of approximate BCP [21] and show that even approximating to accuracy $\varepsilon = 1/\text{poly}(n)$ requires $n^{2-o(1)}$ time.

However, we note that this reduction relies on a strong premise that $f(p)/f(p+1) > n^2$, i.e., the kernel function has *rapid decay*. (The variant in [1] relies on a similar condition.) This fails to hold for many kernels of interest, including all smooth kernels (such as the Rational Quadratic kernels and t -Student kernels) and small-scale Gaussian kernels with small $B < o(\log n)$.

To get around this barrier, Alman, Chu, Schild and Song [2] extends the reduction of [6], by solving *multiple* KDE instances and, roughly speaking, combining their answers to extract *more* information about the pairwise distances. Concretely, we define the *distance vector* (in terms of two sets of points) and *counting matrix* (in terms of the function f) as follows.

► **Definition 3.** *Let $X = \{x^{(1)}, \dots, x^{(n)}\}, Y = \{y^{(1)}, \dots, y^{(n)}\} \in \{0, 1\}^m$ be two sets of points. We define the distance vector $w = [\#\{(i, j) : \|x^{(i)} - y^{(j)}\|_2^2 = p\}]_{p \in [m]}$, and for $\alpha_1, \dots, \alpha_m \in [0, 1]$ define the counting matrix $M = [f(\alpha_\ell \cdot p)]_{\ell, p \in [m]}$.*

Consider the matrix-vector multiplication $M \times w$. By the same argument as in the reduction of [6], we observe each entry in $\tilde{w} := M \times w$ can be computed by some KDE instance $K_\ell \times \mathbf{1}$ (K_ℓ is associated with appropriately scaled X and Y):

$$\tilde{w}[\ell] = \sum_{i=1}^n \sum_{j=1}^n f(\|\sqrt{\alpha_\ell} x^{(i)} - \sqrt{\alpha_\ell} y^{(j)}\|_2^2) = \sum_{p=0}^m f(\alpha_\ell p) \cdot \#\{(i, j) : \|x^{(i)} - y^{(j)}\|_2^2 = p\}.$$

Once \tilde{w} is obtained from KDE subroutines, one can then approximate the distance vector by simply computing $M^{-1} \times \tilde{w}$. If this approximation to w has (additive) error bounded by $1/3$, then a subsequent rounding step yields the exact distance vector, and automatically a solution to the BCP problem.

To analyze this reduction, we define a key quantity

$$\tau(M) := \max_{0 \neq v \in \mathbb{R}^m} \frac{\|M^{-1}v\|_\infty}{\|v\|_\infty}.$$

Suppose $\|K_\ell \times \mathbf{1}\|_\infty \leq \varepsilon \|\mathbf{1}\|_1$ for all $\ell \in [m]$. Then

$$\|M^{-1}\tilde{w} - w\|_\infty = \|M^{-1}(\tilde{w} - Mw)\|_\infty \leq \tau(M) \cdot \max_{\ell \in [m]} \|K_\ell \times \mathbf{1}\|_1 \leq \tau(M) \cdot n^2 \varepsilon.$$

Therefore, any algorithm that approximates KDE instances to accuracy $\varepsilon = 1/(3n^2\tau(M))$ in $n^{2-\Omega(1)}$ time amounts to a truly subquadratic BCP algorithm and refutes SETH. (In this paper below, we slightly modify this approach to improve the dependence on n in ε from quadratic to linear; see Section 1.4 for more details.)

To complete the hardness result, it remains to bound the quantity $\tau(M)$ as a function of m and B . [2] makes a generic statement relating $\tau(M)$ to the approximability of f by low-degree polynomials. In particular, it is shown that for all three kernels – Rational Quadratic kernel, t -Student kernel and small-scale Gaussian kernel – performing $\Omega(\log n)$ -dimensional KDE requires $n^{2-o(1)}$ time to achieve accuracy $\varepsilon = 2^{-\text{poly} \log n}$.

The proof of [2] for this bound on $\tau(M)$ is highly technical, and requires new tools from analysis and linear algebra. For the sake of coherence, we defer an overview of the details to Section 1.4. A main weakness of [2] is that it only gives hardness for very small ε , which is an inherent consequence of their approach to bounding $\tau(M)$. One of the main technical contributions of our paper, which we discuss in more detail shortly, is an improved approach to bounding $\tau(M)$ which yields exponentially better bounds on ε for many kernel functions of interest.

1.3 Our contribution

In this work we give stronger negative results for the KDE problem, and pin down its complexity in each parameter regime. We mainly focus on the Gaussian kernel, and on two of the most used smooth kernels – the Rational Quadratic kernel and the t -Student kernel. That said, our approach is general and would apply to any other kernel of interest after some calculations (See Section 1.5 for further discussion). To give a unified presentation, we formulate both the positive and negative results as upper bounds and lower bounds on $1/\varepsilon$. More specifically, we will answer the following questions.

► **Question 4.** *Fix a kernel function f and a parameter regime determined by $m = o(\log n)$ (resp. $\Theta(\log n)$, $\omega(\log n)$) and $B = o(\log n)$ (resp. $\Omega(\log n)$). What is the range of $1/\varepsilon$ achievable in $n^{1+o(1)}$ time? What is the range of $1/\varepsilon$ that requires $n^{2-o(1)}$ time?*

For simplicity, in the following discussion we understand “Easy” as being achievable in $n^{1+o(1)}$ time, and understand “Hard” as requiring $n^{2-o(1)}$ time.

Gaussian kernel. In the regime $m = \Theta(\log n)$, $B = o(\log n)$, the polynomial method [1] gives the best known positive result: Gaussian KDE is Easy when $1/\varepsilon < (m/B)^{o(m)}$. The best known negative result due to [2] establishes the Hardness of KDE when $1/\varepsilon > 2^{\text{poly} \log n}$. In this work we improve the negative result and show that the polynomial method is optimal up to a low-order $2^{O(m)}$ factor in $1/\varepsilon$.

► **Theorem 5.** *Assuming SETH, for every $q \in (0, 1)$, there exist $C_1, C_2 > 0$ such that when $m > C_1 \log n$ and $1/\varepsilon > (m/B)^m \cdot C_2^m$, GaussianKDE(n, m, B, ε) cannot be solved in $O(n^{2-q})$ time.*

In the low dimensional regime⁵, $c^{\log^* n} < m < o(\log n)/(\log \log n)$, the Fast Multipole Method has stood unchallenged for over three decades. Using this method, Gaussian KDE is Easy when $\log \log(1/\varepsilon) < o(\log n)/m$. It is natural to conjecture that a substantial improvement is impossible. However, to the best of our knowledge, no previous hardness result was known for Gaussian kernel KDE in this regime.⁶ In this work we give the first negative result against the Fast Multipole Method, and in particular show that the $\log \log(1/\varepsilon)$ achieved by the Fast Multipole Method is optimal up to a roughly logarithmic factor in m .

► **Theorem 6.** *Assuming SETH, for every $q > 0$, there exist $C_1, C'_1, C_2 > 0$ such that when $C_1^{\log^* n} < m < C'_1(\log n)/(\log \log n)$ and $\log \log(1/\varepsilon) > (\log n)/m \cdot (\log m) \cdot C_2^{\log^* n}$, GaussianKDE(n, m, B, ε) cannot be solved in $O(n^{2-q})$ time.*

Apart from the two major improvements above, our techniques also lead to new results in other regimes. As a straightforward corollary of Theorem 5, we show $1/\varepsilon > ((\log n)/B)^{\Omega(\log n)}$ is Hard for high-dimensional $m = \omega(\log n)$, small-scale $B = o(\log n)$ regime, improving the $1/\varepsilon > 2^{\text{poly} \log n}$ bound in [2]. For the large scale regime, we note the hardness result in [1] requires $(B/\log n)$ to tend to infinity alongside $(m/\log n)$. This inherent dependence between B and m is an inevitable consequence of the rapid decay condition. In comparison, as our new reduction is free of such restrictions, new hardness results can be developed as well in the regime where $B = \Theta(\log n)$ is fixed and only $m/\log n$ tends to infinity.

We summarize all the known upper and lower bounds for Gaussian KDE in Table 1.

■ **Table 1** Summary of known results for Gaussian KDE, incorporating our new Theorems 5 and 6. The new hardness in high dimensions follows from our Theorem 5 and a straightforward reduction from moderate to high dimension. The stated hardness results for large scale and moderate or high dimension were previously known [6, 1], although we improve the constant C in these cases.

	small scale $B = o(\log n)$	large scale $B = \Omega(\log n)$
low dimension $c^{\log^* n} < m < o\left(\frac{\log n}{\log \log n}\right)$	Easy: $\log \log(1/\varepsilon) < o((\log n)/m)$ Hard (new): $\log \log(1/\varepsilon) > \tilde{\Omega}(\log n)/m$	
moderate dimension $m = C \log n$	Easy: $1/\varepsilon < (m/B)^{o(m)}$ Hard (new): $1/\varepsilon > \Omega(m/B)^m$	Easy: $1/\varepsilon < n^{1-q}$ Hard: $1/\varepsilon > n^C$ for some $C > 1$
high dimension $m > \omega(\log n)$	Easy: $1/\varepsilon < n^{1-q}$ Hard (new): $1/\varepsilon > ((\log n)/B)^{\Omega(\log n)}$	Easy: $1/\varepsilon < n^{1-q}$ Hard: $1/\varepsilon > n^C$ for some $C > 1$

Rational Quadratic kernel and t -Student kernel. For the Rational Quadratic kernel $f(x) = 1/(1+x)^\sigma$ and t -Student kernel $f(x) = 1/(1+x^\rho)$ parameterized by absolute constants $\sigma, \rho \geq 1$, we give similar lower bound results. In the moderate to high dimensional regime $d = \Omega(\log n)$, the best known negative result is again that $1/\varepsilon = 2^{\text{poly} \log n}$ is Hard by [2]. We show that this can be improved to $1/\varepsilon = \text{poly}(n)$. This parallels the improvement

⁵ We use \log^* to denote the very slowly-growing iterated logarithm function.

⁶ For non-Lipschitz kernels, some hardness results for very low error in low dimensions were established in [2].

by [1] for the large-scale Gaussian kernel from $1/\varepsilon = 2^{\text{poly} \log n}$ to $1/\varepsilon = \text{poly}(n)$ (and we recall that these kernels do not decrease quickly enough to prove this using the approach of [6, 1]).

► **Theorem 7.** *For Rational Quadratic kernel $f(x) = 1/(1+x)^\sigma$ and t -Student kernel $f(x) = 1/(1+x^\rho)$ parameterized by absolute constants $\sigma, \rho \geq 1$, the following holds. Assuming SETH, for every $q > 0$, there exists $C_1, C_2 > 0$ such that if $m > C_1 \log n, 1/\varepsilon > n^{C_2}$, then $\text{KDE}_f(n, m, B = 1, \varepsilon)$ cannot be solved in $O(n^{2-q})$. Here C_2 is dependent on σ or ρ .*

This result implies that KDE for Rational Quadratic kernel and t -Student kernel are strictly harder than Gaussian KDE: for $B = O(1)$ and $m = \Theta(\log n)$, Gaussian KDE is Easy when $1/\varepsilon < m^{o(m)}$ whereas KDE for Rational Quadratic kernel and t -Student kernel are Hard for $1/\varepsilon > 2^{\Omega(m)}$. Interestingly, this is in sharp contrast to the phenomenon in the study of KDE with relative error, where KDE for smooth kernels are seemingly easier to solve. We discuss this difference in detail in Section 1.5.

In the low dimensional regime, we also prove negative results against the Fast Multipole Method. For both Rational Quadratic kernel and t -Student kernel, the Fast Multipole Method has similar bound $O(n \log^{O(m)}(n/\varepsilon))$ on running time. We complement this algorithm with a matching lower bound up to a $\tilde{O}(\log m)$ factor in $\log \log(1/\varepsilon)$.

► **Theorem 8.** *For Rational Quadratic kernel $f(x) = 1/(1+x)^\sigma$ and t -Student kernel $f(x) = 1/(1+x^\rho)$ parameterized by absolute constants $\sigma, \rho \geq 1$, the following holds. Assuming SETH, for every $q > 0$, there exist $C_1, C'_1, C_2 > 0$ such that when $C_1^{\log^* n} < m < C'_1(\log n)/(\log \log n)$ and $\log \log(1/\varepsilon) > (\log n)/m \cdot (\log m) \cdot C_2^{\log^* n}$, $\text{KDE}_f(n, m, B = 1, \varepsilon)$ cannot be solved in $O(n^{2-q})$ time. Here C_2 is dependent on σ or ρ .*

1.4 Techniques

As discussed in Section 1.2, [2] established an upper bound on $1/\varepsilon$ which hinges on a key quantity $\tau(M)$, and the central ingredient of their reduction is a bound on this quantity. Therefore we start by sketching the proof [2] developed for this bound. By definition,

$$\tau(M) = \max_{\|v\|_\infty=1} \|M^{-1}v\|_\infty \leq \max_{\|v\|_\infty=1} \max_{s \in [m]} \sum_{t=1}^m |M^{-1}[s, t]| |v[t]| \leq m \max_{s, t \in [m]} |M^{-1}[s, t]|.$$

By Cramer's rule, we write $M^{-1}[s, t] = \frac{\det(M_{t-}^{s-})}{\det(M)}$, where M_{t-}^{s-} is the matrix obtained by removing the s -th row and t -th column of M . Thus it suffices to bound $\det(M)$ and $\det(M_{t-}^{s-})$. We here make use of a common matrix decomposition technique in the study of the polynomial method. If f has Taylor series $f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(0)}{k!} x^k$ convergent over $[0, 1]$, then

$$\det(M) = \det \left[\sum_{k=0}^{\infty} \frac{f^{(k)}(0)}{k!} \alpha_s^k \beta_t^k \right]_{s, t \in [m]} = \det \left(\left[\alpha_s^k \cdot \frac{f^{(k)}(0)}{k!} \right]_{m \times \mathbb{N}} \times \left[\beta_t^k \right]_{\mathbb{N} \times m} \right).$$

One tool for computing determinants of the form $\det(A \times B)$, where A and B are rectangular matrices, is the Cauchy-Binet formula (See Section 2.3 for details), which gives

$$\det(M) = \sum_{0 \leq n_1 < \dots < n_m} \left(\prod_{k=1}^m \frac{f^{(n_k)}(0)}{n_k!} \right) \det \left[\alpha_s^{n_k} \right]_{s, k \in [m]} \det \left[\beta_t^{n_k} \right]_{t, k \in [m]}. \quad (1)$$

Observing that the determinants involved are effectively $(m!)$ -term polynomials in α and β , [2] then views the entire sum as a power series and applies a standard (yet technical) analysis to derive a bound on $\tau(M)$.

In this work, we extend this approach in four aspects.

Direction 1: Schur polynomials. First, we improve on the analysis of the series (1). Although this series is a rather concrete representation of $\det(M)$, the analysis of the $(m!)$ -term polynomials and infinite sum is still a strenuous task and tends not to lead to tight bounds. In this work, we make further inspections of the structure of series (1) and observe that all the determinants $\det[\alpha_s^{n_k}]_{s,k \in [m]}, \det[\beta_t^{n_k}]_{t,k \in [m]}$ have a special structure – they are known as *generalized Vandermonde matrices*. Such matrices have been extensively studied in Algebraic Combinatorics under the name *Schur polynomials*, and are central to the theory of symmetric polynomials. For the classical theory and application of Schur polynomials we point to Chapter 7 of [23]. In recent years, Schur polynomials have also found various applications in computer science, such as in quantum computation [16, 19] and geometric complexity theory [17].

One key property of Schur polynomials lies in its two equivalent definitions. The algebraic definition by Cauchy establishes connection between Schur polynomials and generalized Vandermonde matrices, while the combinatorial definition by Littlewood gives a concrete specification of the coefficients of Schur polynomials.

► **Definition 9 (Schur polynomials).** *Let $m > 0$ be an integer and $\lambda_1 \leq \dots \leq \lambda_m$ be positive integers. We define the Schur polynomial s_λ on variables (u_1, \dots, u_m) by*

$$s_\lambda(u) = s_\lambda(u_1, \dots, u_m) = \frac{\det[u_i^{\lambda_k + (k-1)}]_{i,k \in [m]}}{\prod_{j>i} (u_j - u_i)}. \tag{Cauchy}$$

Equivalently, the Schur polynomial $s_\lambda(u)$ can be defined by

$$s_\lambda(u) = \sum_{T \in \mathcal{T}} \prod_{i=1}^m u_i^{t(T)_i}, \tag{Littlewood}$$

where \mathcal{T} is the set of all semi-standard Young tableaux of shape λ on alphabet $[m]$, and $t(T) \in \mathbb{N}^m$ is the type of T . (See Section 4 for the definition of Young tableaux and associated parameters.)

Based on this property, we are able to represent $\det[\alpha_s^{n_k}]$ and $\det[\beta_t^{n_k}]$ by “neater” polynomials whose nonzero coefficients are uniformly 1, and whose monomials can be enumerated using some combinatorial objects. One can thereby obtain stronger bounds through more straightforward analysis. Moreover, many known results regarding the asymptotic behavior of Schur polynomials also turn out useful in providing guidance on proof strategies.

Direction 2: Special counting matrices. We also observe that for many kernels of interest, the counting matrix M itself has a special structure. For the t -Student kernel $f = 1/(1 + x^\rho)$, its associated counting matrix $M_f = [1/(1 + \alpha_s^\rho \beta_t^\rho)]_{s,t \in [m]}$ is known as a (scaled) Cauchy matrix, as is M_{t-}^{s-} . For the Gaussian kernel $f = e^{-x}$, the associated counting matrix $M_f = [e^{\alpha_s \beta_t}]_{s,t \in [m]}$ is a Vandermonde matrix, and M_{t-}^{s-} is (a relatively simple example of) a generalized Vandermonde matrix. Closed-form formulas are known for determinants of such special matrices. One may thus get around the Cauchy-Binet expansion and bounds can be deduced via a direct argument.

Direction 3: Grouping vector pairs. Regarding the reduction per se, we show the reduction in [2] can be modified so that $\varepsilon = 1/(3n\tau(M))$ suffices, in contrast to the aforementioned $\varepsilon = 1/(3n^2\tau(M))$ lower bound. The main idea is to perform the reduction on $\{x^{(i)}\} \times Y$ for each $x^{(i)} \in X$ separately. Note that for fixed $i \in [n]$,

$$(K \times \mathbf{1})[i] = \sum_{j \in [n]} f(\|x^{(i)} - y^{(j)}\|_2^2) = \sum_{p=0}^m f(p) \cdot \#\{j : \|x^{(i)} - y^{(j)}\|_2^2 = p\}.$$

By the same argument as before, one now recovers the components $w^i = [\#\{j \in [m] : \|x^{(i)} - y^{(j)}\|_2^2 = p\}]_{p \in [m]}$ of w for respective $i \in [m]$. We note, in this new reduction, it suffices to approximate single entries $(K \times \mathbf{1})[i]$, which is arguably a simpler task than approximating $\|K \times \mathbf{1}\|_1$, as an error n times as large may accumulate in the latter case. Formalizing this idea in Section 3, we successfully shave a factor of n .

This improvement on the dependence of n in addition raises an interesting question. In the high dimensional regime, [20] showed that for any positive definite kernel f , one can compute $\text{KDE}_f(n, m, \varepsilon, B)$ in truly subquadratic time with accuracy $1/\varepsilon < n^{1-\delta}$ for any constant $\delta > 0$. Against this algorithm, our improved reduction produces a lower bound for $1/\varepsilon > \Omega(n) \cdot \tau(M)$, leaving a gap of simply $\tau(M)$. This brings within reach a potential tightness result: the optimality of [20] can be (in part) established as long as one can bound $\tau(M) = O(1)$ for some positive definite kernel f . Such a result is arguably hard to obtain from the previous lower bounds by either [6] or [2].

Direction 4: Low-dimensional BCP. To extend the negative result to the low-dimensional regime, we combine our main reduction with a variant of the BCP problem. Williams [25] and Chen [12] showed that the BCP problem for vectors with *integer entries* remains hard even in extremely low dimensions $d = 2^{O(\log^* n)}$. (See Section 2.1 for a formal statement). With slight modification, our main reduction can use KDE subroutines to recover the distance vector for not only datapoints in $\{0, 1\}^m$ but those in \mathbb{Z}^m (though a larger counting matrix is required). A hardness result for KDE in low dimensions thus follows from similar analysis. Moreover, looking into the proofs of [25] and [12], we notice that they effectively showed a stronger trade-off between the dimension of vectors and magnitude of vector entries. Translated into the setting of KDE, this is a trade-off between dimension m and approximation error ε .

1.5 Discussion

Additive vs. Relative Error. In the previous discussion we focused mainly on the KDE problem with additive error. In recent years, much effort has also been dedicated to algorithm design in the setting with relative error, primarily in the moderate to high dimensional regime $d = \Omega(\log n)$. In this setting, the running time of KDE algorithms normally depends not only on the relative error parameter ε_R but also on a lower bound of the kernel value $\mu = \min_{x \in [0, B]} f(x)$. The folklore random sampling algorithm runs in time $O(n\varepsilon_R^{-2}\mu^{-1})$. For the Gaussian kernel, Charikar and Siminelakis [10] made the first major improvement by designing a $O(n\varepsilon_R^{-2}\mu^{-0.5})$ -time algorithm using a LSH-based Importance Sampling scheme. Later Charikar et al. [8] presented an improved implementation of Importance sampling that achieves $O(n\varepsilon_R^{-2}\mu^{-0.173})$ running time. For smooth kernels (including the Rational Quadratic kernel and t -Student kernel), the first non-trivial improvement was due to Backurs et al. [5], who presented an algorithm running in $n\varepsilon_R^{-2}\text{poly log}(\mu^{-1})$ time using tree-based space partitioning techniques. Recently, Charikar, Kapralov and Waingarten [9] combined this result with the discrepancy based sampling scheme by Phillips and Tai [20] and achieved $n\varepsilon_R^{-1}\text{poly log}(\mu^{-1})$ running time, improving the dependence on ε_R .

Interestingly, our new lower bounds in Section 1.3 exhibits a sharp contrast between the additive and relative error setting. The best known KDE algorithms stated above suggest that KDE for smooth kernels are likely easier than that for Gaussian-like kernels in the

35:10 Finer-Grained Hardness of Kernel Density Estimation

relative error setting. However, comparing between Theorem 5 and Theorem 7, we observe an opposite trend. For example, for $B = O(1)$ and $m = \Theta(\log n)$, Gaussian KDE is Easy when $1/\varepsilon < m^{o(m)}$ whereas KDE for Rational Quadratic kernel and t -Student kernel are Hard even for $1/\varepsilon = 2^{\Omega(m)}$. This difference suggests that the discrepancy between two formulations is likely inherent and they should be treated with respective care.

Dynamic vs. Batched KDE. As we see from the reduction of [6] and [2], the BCP problem naturally reduces to the batched version of KDE, and thereby we take batched KDE as the primary formulation in this work for simplicity. In the literature, the KDE problem is equally often phrased in its dynamic version, e.g., in [10, 20, 9]. In the dynamic KDE problem for kernel $k(x, y)$, one is given a dataset $X \subset \mathbb{R}^m$ and a vector $w \in \mathbb{R}^n$, and asked to design a data structure \mathcal{A} that preprocesses X and outputs an approximation to the Kernel Density $\sum_{x \in X} k(x, q)w[x]$ for each query point $q \in \mathbb{R}^m$. Given a KDE data structure, one can easily build a batched KDE algorithm in time $T(\text{preprocessing}) + n \cdot T(\text{query})$. Hence any hardness result proved for batched KDE automatically holds for dynamic KDE as well. It is not clear whether there is a reduction in the reverse direction, and it remains an open problem to determine whether the batched version is strictly easier. Nonetheless, all the known algorithms including the Fast Multipole method, Polynomial method and sampling-based methods, are data structures or can be modified to solve the dynamic problem.

2 Preliminaries

2.1 SETH and known hard problems

We now introduce several variants of the Bichromatic Closest Pair problem.

► **Problem 10** (Hamming (Exact) Bichromatic Closest Pair). *Hamming-BCP(n, m): Given two sets $A = \{x^{(1)}, \dots, x^{(n)}\}, B = \{y^{(1)}, \dots, y^{(n)}\} \subset \{0, 1\}^m$, compute $\min_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2$.*

► **Theorem 11** ([4]). *Assuming SETH, for every $q \in (0, 1)$, there exists $C > 0$ such that if $m > C \log n$, then Hamming-BCP(n, m) cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$.*

Similarly, one can define the Hamming approximate BCP problem and its decision version.

► **Problem 12** (Hamming Approximate BCP). *Hamming-Apx-BCP(n, m, μ): Given two sets A, B as in Problem 10 as well as $\mu \in \mathbb{R}_+$, output $d \in \mathbb{R}$ such that $\min_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2 \leq d \leq (1 + \mu) \min_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2$.*

► **Theorem 13** ([21]). *Assuming SETH, for every $q > 0$, there exist $C > 0, \mu > 0$ such that if $m > C \log n$, then Hamming-Apx-BCP(n, m, μ) cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$.*

In the low dimensional regime $m = o(\log n)$, we consider the hardness of the ℓ_2 BCP problem.

► **Problem 14** (ℓ_2 (Exact) Bichromatic Closest Pair). *ℓ_2 -BCP(n, m, D): Given two sets $A = \{x^{(1)}, \dots, x^{(n)}\}, B = \{y^{(1)}, \dots, y^{(n)}\} \in \mathbb{Z}^m$ such that $\max_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2 \leq D$, compute $\min_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2$.*

► **Theorem 15** ([12]). *Assuming SETH, for every $q > 0$, there exists $C_1, C_2 > 0$ such that if $m > C_1^{\log^* n}$ and $D > m^{C_2^{\log^* n} \cdot (\log n)/m}$, then ℓ_2 -BCP(n, m, D) cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$.*

To unify the hardness results for BCP in different dimension regimes, we view the Hamming BCP problem as an ℓ_2 BCP problem with $D = m$. Formally, we combine Theorem 13 and Theorem 15 as follows.

► **Theorem 16.** *Assuming SETH, for every $q > 0$, there exists $C > 0$ such that ℓ_2 -BCP(n, m, D) cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$ if either of the following holds: (1) $m > C \log n, D = m$, or (2) $m > C^{\log^* n}, D > m^{C^{\log^* n} \cdot (\log n)/m}$.*

2.2 Kernels of interest

In this work we focus primarily on three kernels $k(x, y) = f(\|x - y\|_2^2)$:

- Gaussian kernel $f(x) = e^{-x}$;
- Rational Quadratic kernel $f(x) = 1/(1 + x)^\sigma$ for $\sigma \geq 1$ a parameter;
- t -Student kernel $f(x) = 1/(1 + x^\rho)$ for $\rho \geq 1$ a parameter.⁷

Rational Quadratic kernel and t -Student kernel are two typical kernels with mild decay (as opposed to the rapid decay of Gaussian kernel). This property is abstracted by Backurs et al. [5] in the definition of a smooth kernel. Here we only focus on decreasing radial kernels.

► **Definition 17 (smooth kernel).** *A decreasing radial kernel $k(x, y) = f(\|x - y\|_2^2)$ is (L, t) smooth if for any $0 < a < b$, $\frac{f(a)}{f(b)} \leq L \left(\frac{b}{a}\right)^t$.*

By calculation one can verify that Rational Quadratic kernel and t -Student kernel are respectively $(1, 1)$ - and $(1, \rho)$ -smooth.

Positive definite kernels. Most commonly-studied kernels are positive definite. We will find that our lower bound approach takes a particularly nice form for such kernels.

► **Definition 18 (Positive definite kernel).** *A kernel $k : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ is positive definite if for any n points $x_1, \dots, x_n \in \mathbb{R}^m$, the Gram matrix $G = [k(x_i, x_j)]_{i, j \in [n]}$ is always positive definite.*

For radial kernels, we have the following concise characterization of positive definite kernels.

► **Definition 19.** *Let $f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real function. We say f is absolutely monotone if $G \in C^\infty(\mathbb{R}_{\geq 0})$ and $f^{(k)}(t)$ for all $k \in \mathbb{N}$ and $t \geq 0$, and we say f is completely monotone if $G \in C^\infty(\mathbb{R}_{\geq 0})$ and $(-1)^k \cdot f^{(k)}(t) \geq 0$ for all $k \in \mathbb{N}$ and $t \geq 0$.*

► **Theorem 20 (Schoenberg's characterization).** *Let $f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real function. Then the kernel $k(x, y) = f(\|x - y\|_2^2)$ is positive definite if and only if f is completely monotone on $\mathbb{R}_{\geq 0}$.*

2.3 Tools from linear algebra

As in prior work, we will make use of the Cauchy-Binet formula.

► **Lemma 21 (Cauchy-Binet formula).** *Let $k > 0$ be an integer, and for functions $A : [k] \times \mathbb{N} \rightarrow \mathbb{R}$ and $B : \mathbb{N} \times [k] \rightarrow \mathbb{R}$, define the matrix $C \in \mathbb{R}^{k \times k}$ by, for $i, j \in [k]$, $C[i, j] = \sum_{\ell=0}^\infty A[i, \ell]B[\ell, j]$. If the sum converges absolutely for all i, j , then*

$$\det(C) = \sum_{1 \leq \ell_1 < \dots < \ell_k} \det(A[\ell_1, \dots, \ell_k]) \cdot \det(B[\ell_1, \dots, \ell_k]).$$

Here $A[\ell_1, \dots, \ell_k]$ (resp. $B[\ell_1, \dots, \ell_k]$) is the $k \times k$ matrix obtained from A (resp. B) by taking the columns (resp. rows) ℓ_1, \dots, ℓ_k .

⁷ t in the name of the kernel in principle should be the name of the parameter. We here use ρ as parameter while keeping the name t -Student kernel unchanged.

3 Main reduction from BCP

Most of our new lower bounds are based on the reduction below from Hamming or ℓ_2 Exact Bichromatic Closest Pair to KDE. This reduction generalizes a framework developed in [2] to accommodate different parameter regimes. In this section we will give the outline of the reduction and establish an upper bound on $1/\varepsilon$ determined by the quantity $\tau(M)$.

► **Definition 22.** Let $D > 0$ be an integer. Fix vectors $\alpha, \beta \in \mathbb{R}^D$ and $c \in \mathbb{R}$. Suppose $c + \alpha_\ell \beta_p \in [0, 1]$ for all $\ell, p \in [D]$. Then for function $f : [0, 1] \rightarrow \mathbb{R}$, the counting matrix $M = M(D; f, \alpha, \beta)$ is a $D \times D$ matrix defined by $M[\ell, p] = f(c + \alpha_\ell \beta_p)$, $\ell, p \in [D]$.

► **Theorem 23.** Let $\alpha \in \mathbb{R}^D$ be a fixed vector and $\beta \in \mathbb{R}^D$ be the identity vector defined by $\beta_p = p$. If $\text{KDE}_f(n, m, \varepsilon)$ can be solved in $T(n, m, \varepsilon)$ time, then ℓ_2 -BCP(n, m, D) with $m = n^{o(1)}$, $D = n^{o(1)}$ can be solved in $T(n, m + 1, (3n\tau(M))^{-1}) \cdot n^{o(1)} + n^{1+o(1)}$ time, where M is the $D \times D$ counting matrix associated with f, α, β and $\tau(M) = \max_{0 \neq b \in \mathbb{R}^D} \frac{\|M^{-1}b\|_\infty}{\|b\|_\infty}$.

Proof. Given two sets $X = \{x^{(1)}, \dots, x^{(n)}\}, Y = \{y^{(1)}, \dots, y^{(n)}\} \subseteq \mathbb{Z}^m$ such that $\max_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2 \leq D$, let $W \in \mathbb{N}^{D \times n}$ denote the distance count matrix defined by $W[p, i] = \#\{j \in [n] : \|x^{(i)} - y^{(j)}\|_2^2 = p\}$. Then the matrix product $U = M \times W$ gives

$$\begin{aligned} U[\ell, i] &= \sum_{p=1}^D M[\ell, p] \cdot W[p, i] = \sum_{p=1}^D f(c + \alpha_\ell \beta_p) \sum_{j \in [n]} \mathbb{1}[\|x^{(i)} - y^{(j)}\|_2^2 = p] \\ &= \sum_{j \in [n]} f\left(c + \alpha_\ell \cdot \beta \left[\|x^{(i)} - y^{(j)}\|_2^2\right]\right). \end{aligned} \quad (2)$$

We note that when β is the identity vector, the summation (2) can be formulated as a KDE instance. More specifically, let $\tilde{x}_\ell^{(i)}, \tilde{y}_\ell^{(j)} \in \mathbb{R}^{m+1}$ be defined by $\tilde{x}_\ell^{(i)}[k] = \sqrt{\alpha(\ell)}x^{(i)}[k]$ if $k \in [m]$ and $\tilde{x}_\ell^{(i)}[m+1] = \sqrt{c}$; $\tilde{y}_\ell^{(j)}[k] = \sqrt{\alpha(\ell)}y^{(j)}[k]$ if $k \in [m]$ and $\tilde{y}_\ell^{(j)}[m+1] = 0$. Then $U[\ell, i] = \sum_{j \in [n]} f(\|\tilde{x}_\ell^{(i)} - \tilde{y}_\ell^{(j)}\|_2^2)$. Therefore we have the following algorithm for ℓ_2 -BCP(n, m, D) using KDE_f as a subroutine. On input $X = \{x^{(1)}, \dots, x^{(n)}\}, Y = \{y^{(1)}, \dots, y^{(n)}\}$ such that $\max_{i, j \in [n]} \|x^{(i)} - y^{(j)}\|_2^2 \leq D$:

1. For $\ell \in [D]$, construct vectors $\tilde{x}_\ell^{(i)}, \tilde{y}_\ell^{(j)}, i, j \in [n]$. Then approximate the ℓ -th row of U : $\hat{U}[\ell] \approx U[\ell] = K_\ell \times \mathbf{1}$ using the KDE_f oracle.
2. Compute $\hat{W} = M^{-1} \times \hat{U}$, and round each entry to the closest integer.

We claim that if we call the KDE_f subroutine with $\varepsilon = (3n\tau(M))^{-1}$, then the distance count matrix W is exactly recovered after the rounding step. Indeed, for fixed $i \in [n]$, letting $W[\cdot, i], U[\cdot, i]$ respectively denote the i -th column of matrix W and U , we have

$$\|\hat{W}[\cdot, i] - W[\cdot, i]\|_\infty = \|M^{-1}(\hat{U}[\cdot, i] - U[\cdot, i])\|_\infty \leq \tau(M) \cdot \|\hat{U}[\cdot, i] - U[\cdot, i]\|_\infty.$$

If the KDE_f subroutine guarantees that $\|\hat{U}[\ell] - U[\ell]\|_\infty \leq (3n\tau(M))^{-1} \cdot \|\mathbf{1}\|_1 = (3\tau(M))^{-1}$ for all $\ell \in \{1, \dots, D\}$, then the entry-wise difference between \hat{W} and W is bounded by $1/3$.

The D calls to the subroutine then take in total $D \cdot T(n, m, (3n\tau(M))^{-1})$ time. It takes in addition $D \cdot O(nm) = n^{1+o(1)}$ operations to construct the vectors and $O(D^\omega)$ time for step 2.⁸ ◀

⁸ We always assume $f(x)$ can be exactly computed in constant time for any $x \in [0, 1]$.

We now combine the reduction above and the hardness result of BCP in Theorem 16. The hardness of the KDE problem follows.

► **Theorem 24.** *Assuming SETH, for every $q > 0$, there exists $C \geq 0$ such that $\text{KDE}_f(n, m, (3n\tau(M))^{-1})$ cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$. if either of the following holds: (1) $m > C \log n, D = m$ or (2) $m > C^{\log^* n}, D > m^{C^{\log^* n} \cdot (\log n)/m}$. Here M is the $D \times D$ counting matrix in Definition 22.*

4 Schur polynomials

Let \mathbb{F} be a field. For matrices $A, B \in \mathbb{F}^{m \times n}$, let $A \circ B$ denote the Hadamard product of A and B , i.e. the $m \times n$ matrix with entries $(A \circ B)[i, j] = A[i, j] \cdot B[i, j], i \in [m], j \in [n]$. For matrix $A \in \mathbb{F}^{m \times n}$ we define the Hadamard powers $A^{\circ 1} = A$ and $A^{\circ(k+1)} = A^{\circ k} \circ A$ for integer $k \geq 1$. Similarly one can define the Hadamard product and Hadamard power for vectors $u \in \mathbb{F}^m$. Moreover, given a vector $u \in \mathbb{F}^m$ and a tuple $r = (r_1, \dots, r_n) \in \mathbb{N}^n$, we denote by $u^{\circ r}$ the $m \times n$ matrix

$$\left(\begin{array}{c|c|c|c} u^{\circ r_1} & u^{\circ r_2} & \dots & u^{\circ r_n} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline u_m^{\circ r_1} & u_m^{\circ r_2} & \dots & u_m^{\circ r_n} \end{array} \right) = \begin{pmatrix} u_1^{r_1} & u_1^{r_2} & \dots & u_1^{r_n} \\ \vdots & \vdots & \ddots & \vdots \\ u_m^{r_1} & u_m^{r_2} & \dots & u_m^{r_n} \end{pmatrix}.$$

In particular, when $m = n$, we call $u^{\circ r}$ a generalized Vandermonde matrix. This is a natural generalization of the (usual) Vandermonde matrix $u^{\circ \delta}$ associated with the tuple $\delta = (0, 1, \dots, m-1)$.

The concept of Schur polynomials was first proposed by Cauchy and defined as the ratio of two generalized Vandermonde determinants. In what follows we denote by $\mathbb{N}_{<}^m = \{x \in \mathbb{N}^m : x_1 < \dots < x_m\}$ the set of m -tuples composed of distinct entries in ascending order, and define $\mathbb{N}_{\leq}^m = \{x \in \mathbb{N}^m : x_1 \leq \dots \leq x_m\}$ similarly.

► **Definition 25** (Cauchy's definition of Schur polynomials). *Let $m > 0$ be an integer and $\lambda = (\lambda_1, \dots, \lambda_m) \in \mathbb{N}_{\leq}^m$ an integer tuple. We define the Schur polynomial on variables (u_1, \dots, u_m) by*

$$s_\lambda(u) = s_\lambda(u_1, \dots, u_m) = \frac{\det(u^{\circ(\lambda+\delta)})}{\det(u^{\circ \delta})} = \frac{\det(u^{\circ(\lambda+\delta)})}{V(u)}.$$

Here $V(u) = \det(u^{\circ \delta}) = \prod_{1 \leq i < j \leq m} (u_j - u_i)$ is the Vandermonde determinant.

It is known that Schur polynomial has an equivalent definition by Littlewood using Young tableaux.

► **Definition 26.** *Let $m > 0$ be an integer, and $\lambda = (\lambda_1, \dots, \lambda_m) \in \mathbb{N}_{\leq}^m$ an integer tuple. A semi-standard Young tableau (SSYT) of shape λ on alphabet $[m]$ is a left-aligned two-dimensional rectangular array T of cells, with λ_i cells in the i -th row ($i \in [m]$), from bottom to top, such that*

- each cell in T is assigned with an entry from $1, \dots, m$;
- entries weakly decrease in each row, from left to right;
- entries strictly decrease in each column, from top to bottom.

Moreover, for a SSYT T , we define the type $\mathfrak{t}(T) = (t_1, \dots, t_m) \in \mathbb{N}^m$ of T , where t_j is the number of cells in T assigned with entry $j \in [m]$.

35:14 Finer-Grained Hardness of Kernel Density Estimation

► **Definition 27** (Littlewood's definition of Schur polynomials). *Let $m > 0$ be an integer. For integer tuple $\lambda = (\lambda_1, \dots, \lambda_m) \in \mathbb{N}_{\leq}^m$, the Schur polynomial $s_\lambda(u)$ can be equivalently defined by*

$$s_\lambda(u) = \sum_{T \in \mathcal{T}} u^{t(T)},$$

where \mathcal{T} is the set of all SSYT of shape λ on alphabet $[m]$.

5 Direct calculation of $\tau(M)$: Gaussian kernel and t -Student kernel

5.1 Gaussian kernel

In this section we show upper bounds for

$$\tau(M) = \max_{b \in \mathbb{R}^D: b \neq 0} \frac{\|M^{-1}b\|_\infty}{\|b\|_\infty} \leq D \cdot \max_{s, t \in [D]} |M^{-1}[s, t]|$$

where M is the counting matrix associated with the Gaussian kernel $f(x) = e^{-Bx}$.⁹ By standard facts in linear algebra, we can rewrite $|M^{-1}[s, t]| = \left| \frac{\det(M_{t-}^{s-})}{\det(M)} \right|$ with M_{t-}^{s-} the submatrix of M consisting of all entries but those in row s or column t .

For the simplicity of analysis, we study the counting matrix M of dimension $(D+1)$. A slight modification of the main reduction can employ such a matrix to recover the distance matrix with a redundant row: $W[p, i] = \#\|x^{(i)} - y^{(j)}\|_2^2 = p, i \in [n], p = \{0, 1, \dots, D\}$.

► **Theorem 28.** *Let $\alpha \in \mathbb{R}^{D+1}$ be a fixed vector and $\beta \in \mathbb{R}^{D+1}$ be the identity vector. Let M be the $(D+1) \times (D+1)$ counting matrix associated with Gaussian kernel $f = e^{-Bx}$ and α, β . Then there exists a vector α such that $\tau(M) \leq \left(\frac{5e}{1-e^{-B/D}} \right)^D$.*

Proof. Let $x_i = \exp(-B\alpha_i)$, then $M = [x_i^j : i, j \in \{0, \dots, D\}]$ is a Vandermonde matrix with $|\det(M)| = |\det(x^{\circ(0, \dots, D)})| = V(x)$. On the other hand, we observe that M_{t-}^{s-} can be viewed as a generalized Vandermonde matrix. Making use of both the algebraic and combinatorial definition of Schur polynomials, we have¹⁰

$$|\det(M_{t-}^{s-})| = |\det((x^{s-})^{\circ(0, \dots, t-1, t+1, \dots, D)})| = V(x^{s-}) \sum_{R \subseteq \{0, \dots, D\} \setminus \{s\}} \prod_{r \in R} x^r,$$

in which x^{s-} denotes the vector $(x_0, \dots, x_{s-1}, x_{s+1}, \dots, x_D)$ and $\sum_{R \subseteq \{0, \dots, D\} \setminus \{s\}} \prod_{r \in R} x^r \leq \prod_{i \in \{0, \dots, D\} \setminus \{s\}} (1 + x_i) \leq 2^D$. Therefore

$$\max_{s, t} |\det(M_{t-}^{s-})| \leq 2^D \cdot \max_s V(x^{s-}) = 2^D \cdot V(x) \left(\min_s \prod_{i \in [D] \setminus \{s\}} |x_i - x_s| \right)^{-1}.$$

We now pick $0 = \alpha_0 < \dots < \alpha_{D-1} < \alpha_D = 1/D$ such that $|x_{i+1} - x_i| = |\exp(-B\alpha_{i+1}) - \exp(-B\alpha_i)| = \frac{1}{D}(1 - e^{-B/D})$. Let $r = \frac{1}{D}(1 - e^{-B/D})$, then

⁹ By a straightforward reduction, $\text{KDE}_f(n, m, \varepsilon, B)$ is equivalent to $\text{KDE}_g(n, m, \varepsilon, B = 1)$ where $g = f(Bx)$.

¹⁰ The Schur polynomial here in fact equals an elementary symmetric polynomial.

$$\begin{aligned} \prod_{i \in \{0, \dots, D\} \setminus \{s\}} |x_i - x_s| &= \left(\prod_{i=0}^{s-1} (s-i)r \right) \left(\prod_{i=s+1}^D (i-s)r \right) \\ &= r^D \cdot s!(D-s)! \geq r^D \cdot \left(\frac{D}{2e}\right)^{D/2.2} = \left(\frac{1-e^{-B/D}}{2e}\right)^D. \end{aligned}$$

Combining the calculations together, we have

$$\tau(M) \leq D \cdot \max_{s,t} \left| \frac{\det(M_{t-}^{s-})}{\det(M)} \right| \leq D \cdot 2^D \left(\min_s \prod_{i \in [D] \setminus \{s\}} |x_i - x_s| \right)^{-1} \leq \left(\frac{5e}{1-e^{-B/D}} \right)^D. \blacktriangleleft$$

The following hardness result for Gaussian KDE therefore arises from a combination of the general KDE hardness Theorem 24 and the bound on $\tau(M)$ above.

► **Theorem 29** (Hardness of Gaussian KDE). *Let $f(x) = e^{-Bx}$ be the Gaussian kernel with $B \geq 1$. Then assuming SETH, for every $q > 0$, there exists $C_1, C'_1, C_2, C_3, C'_3, C_4 \geq 0$ such that $\text{KDE}_f(n, m, \varepsilon)$ cannot be solved in time $O(n^{2-q})$ if either of the following holds:*

- (1) $m > C_1 \log n, B < C'_1 \log n, 1/\varepsilon > (C_2 m/B)^m$, or
- (2) $C_3 \log^* n < m < C'_3 (\log n)/(\log \log n), \log \log(1/\varepsilon) > (\log n)/m \cdot (\log m) \cdot C_4^{\log^* n}$.

Proof. For regime (1), first pick appropriate C'_1 so that $B/m < 1$.

$$3n\tau(M) = 3n \left(\frac{5e}{1-e^{-B/m}} \right)^m \leq 3n \left(\frac{5e}{B/(em)} \right)^m = 3 \cdot 2^{C_1^{-1}m} \cdot \left(\frac{5e^2m}{B} \right)^m \leq \left(\frac{C_2m}{B} \right)^m$$

for some constant $C_2 > 0$. For regime (2), we have $B/D \leq 1$ and thus

$$3n \left(\frac{5e}{1-e^{-B/D}} \right)^D \leq 3n \left(\frac{5e^2D}{B} \right)^D \leq 3n(5e^2D)^D.$$

If $m < (\log n)/(\log \log n)$, $\log D = (\log n)/m \cdot (\log m) \cdot C^{\log^* m} = (\log \log n)^2(1-o(1)) \cdot C^{\log^* m} > \log \log n$. $\log(3n\tau(M)) \leq \log(3n) + D \log(5e^2D) < C'D \log(5e^2D)$. For some $C' > 0$. Thus, $\log \log(3n\tau(M)) \leq \log D + \log \log(5e^2D) \leq (\log n)/m \cdot (\log m) \cdot C_4^{\log^* m}$. ◀

5.2 t -Student kernel

For t -Student kernels $f(x) = 1/(1+x^p)$, we prove a similar bound on $\tau(M)$. The key observation here is that both M and M_{s-}^t are (scaled) Cauchy matrices. For vectors $a, b \in \mathbb{R}^n$, the Cauchy matrix associated with a, b is defined by $M[i, j] = 1/(a_i + b_j), i, j \in [n]$. The following closed-form formula is known for its determinant.

► **Theorem 30** (Cauchy determinant). *Let $n \geq 1$ be an integer. For $a, b \in \mathbb{R}^n$, let $M = [1/(a_i + b_j)]_{i,j \in [n]}$ be the associated Cauchy matrix. Then*

$$\det(M) = \frac{\prod_{1 \leq i < j \leq n} (a_i - a_j)(b_i - b_j)}{\prod_{i,j \in [n]} (a_i + b_j)}.$$

► **Corollary 31.** *For $a, b \in \mathbb{R}^n$ with $a_i, b_j \in (0, 1), \forall i, j \in [n]$, and $s, t \in [n]$, we have*

$$\left| \frac{\det[(1+a_i b_j)^{-1}]_{i,j \in [n]}}{\det[(1+a_i b_j)^{-1}]_{i \in [n]^{s-}, j \in [n]^{t-}}} \right| = \left| \frac{\prod_{i \in [n]^{s-}} (a_i - a_s) \prod_{j \in [n]^{t-}} (b_j - b_t)}{\prod_{i \in [n]} (1+a_i b_t) \prod_{j \in [n]^{t-}} (1+a_s b_j)} \right|.$$

Here $[n]^{s-} = [n] \setminus \{s\}, [n]^{t-} = [n] \setminus \{t\}$.

35:16 Finer-Grained Hardness of Kernel Density Estimation

Before proving the bound on $\tau(M)$, we gather several simple inequalities to be used.

► **Lemma 32.**

(I) Let $r \geq 1, a, b > 0$ be real numbers. Then $(a + b)^r \geq a^r + b^r$.

(II) Let $a, b \geq 0$ be integers. Then $a!b! \geq ((\lfloor \frac{a+b}{2} \rfloor)!)^2$.

Proof.

(I) Taking the derivative, one can show $f(x) = (1 + x)^r - x^r$ is increasing in x for $x > 0$. Thus $f(b/a) = (1 + b/a)^r - (b/a)^r \geq f(0) = 1$. The inequality follows.

(II) For $a = b = 0$, the inequality is trivial. If not, we have $\binom{a+b}{a} \leq \binom{a+b}{\lceil (a+b)/2 \rceil}$, $a!b! \geq (\lceil \frac{a+b}{2} \rceil)! (\lfloor \frac{a+b}{2} \rfloor)! \geq ((\lfloor \frac{a+b}{2} \rfloor)!)^2$. ◀

► **Theorem 33.** Let $\rho \geq 1$ be a real number. Let $\alpha \in \mathbb{R}^D$ be a fixed vector and $\beta \in \mathbb{R}^D$ be the identity vector. Let M be the $D \times D$ counting matrix associated with t -Student kernel $f(x) = 1/(1 + x^\rho)$ and α, β . Then there exists a vector α such that $\tau(M) \leq (7e)^{2\rho D}$.

Proof. The counting matrix is $M = [f(\alpha_i \beta_j)]_{i,j \in [D]} = \left[\frac{1}{1 + (\alpha_i \beta_j)^\rho} \right]_{i,j \in [D]}$. For simplicity we assume D is odd. Let $s, t \in [D]$. By corollary 31, we have

$$\left| \frac{\det(M)}{\det(M_{t-}^{s-})} \right| = \left| \frac{\prod_{i \in [D]^{s-}} (\alpha_i^\rho - \alpha_s^\rho) \prod_{j \in [D]^{t-}} (\beta_j^\rho - \beta_t^\rho)}{\prod_{i \in [D]} (1 + \alpha_i^\rho \beta_t^\rho) \prod_{j \in [D]^{t-}} (1 + \alpha_s^\rho \beta_j^\rho)} \right|.$$

We set $\alpha = \beta$ to be the scaled identity vector with $\alpha_i = \beta_i = i/D, \forall i \in [D]$. (By rescaling $\alpha_i = i/D^2, \beta_i = i$ one can make β the identity vector.) Then $\prod_{i \in [D]} (1 + \alpha_i^\rho \beta_t^\rho) \prod_{j \in [D]^{t-}} (1 + \alpha_s^\rho \beta_j^\rho) \leq 2^{2D-1}$,

$$\begin{aligned} \prod_{i \in [D]^{s-}} |\alpha_i^\rho - \alpha_s^\rho| &= \prod_{i=1}^{s-1} \frac{s^\rho - i^\rho}{D^\rho} \prod_{i=s+1}^D \frac{i^\rho - s^\rho}{D^\rho} \geq \prod_{i=1}^{s-1} \frac{(s-i)^\rho}{D^\rho} \prod_{i=s+1}^D \frac{(i-s)^\rho}{D^\rho} \\ &= \left(\frac{(s-1)!(D-s)!}{D^{D-1}} \right)^\rho \geq \left(\frac{(\lfloor \frac{D-1}{2} \rfloor)!}{D^{D-1}} \right)^\rho \geq \left(\frac{D-1}{2eD} \right)^{\rho(D-1)} \geq (3e)^{-\rho D}. \end{aligned}$$

Similarly, $\prod_{j \in [D]^{t-}} |\beta_j^\rho - \beta_t^\rho| \geq (3e)^{-\rho D}$. In conclusion, we have $\tau(M) \leq D \cdot \max_{s,t} \left| \frac{\det(M_{t-}^{s-})}{\det(M)} \right| \leq D \cdot 2^{2D} \cdot (3e)^{2\rho D} \leq (7e)^{2\rho D}$. ◀

Combining the bound on $\tau(M)$ above and the general KDE hardness Theorem 24, we obtain

► **Theorem 34** (Hardness results of t -Student KDE). Let $f(x) = 1/(1 + x^\rho)$ be a t -Student kernel parameterized by $\rho \geq 1$ an absolute constant. Then assuming SETH, for every $q > 0$, there exists $C_1, C_2, C_3, C'_3, C_4 \geq 0$ such that $\text{KDE}_f(n, m, \varepsilon)$ cannot be solved in time $O(n^{2-q})$ if either of the following holds:

(1) $m > C_1 \log n, 1/\varepsilon > n^{C_2}$, or

(2) $C_3 \log^* n < m < C'_3 (\log n) / (\log \log n)$, $\log \log(1/\varepsilon) > (\log n)/m \cdot (\log m) \cdot C_4^{\log^* n}$.

Here C_1, C_3, C'_3 are absolute constants while C_2, C_4 are dependent on ρ .

Proof. Analogous to the proof of Theorem 29. ◀

6 Cauchy-Binet Expansion

To bound $\tau(M)$ for counting matrices that are associated with more general kernels, we establish the connection between the determinant $\det(M)$ and the Taylor coefficients of f via Cauchy-Binet Expansion, based on a framework given in [2]. Our improvement mainly derives from an observation that the terms in the expansion are in fact generalized Vandermonde determinants. The additional structure in the determinants allows for simplifications in analysis through Schur polynomials.

Define infinite matrices $A : [D] \times \mathbb{N} \rightarrow \mathbb{R}$, $\tilde{A} : [D] \times \mathbb{N} \rightarrow \mathbb{R}$, and $B : \mathbb{N} \times [D] \rightarrow \mathbb{R}$ by

$$A[i, k] = \alpha_i^k, \quad \tilde{A}[i, k] = \frac{f^{(k)}(c)}{k!} \alpha_i^k, \quad B[k, j] = \beta_j^k.$$

Suppose the Taylor series of $f(c+x)$ at $c \in [0, 1]$ is *absolutely convergent* whenever $c+x \in [0, 1]$. Then for $i, j \in [D]$, assuming $c + \alpha_i \beta_j \in [0, 1]$, we have

$$M[i, j] = f(c + \alpha_i \beta_j) = \sum_{k=0}^{\infty} \frac{f^{(k)}(c)}{k!} \alpha_i^k \beta_j^k = \sum_{k=0}^{\infty} \tilde{A}[i, k] B[k, j].$$

By Cauchy-Binet formula, we expand $\det(M)$ as follows.

$$\begin{aligned} \det(M) &= \sum_{\bar{n} \in \mathbb{N}_{\leq}^D} \det \tilde{A}[[D]; \bar{n}] \cdot \det B[\bar{n}; [D]] \\ &= \sum_{\lambda \in \mathbb{N}_{\leq}^D} \left(\prod_{i=1}^D \frac{f^{(\lambda_i + \delta_i)}(c)}{(\lambda_i + \delta_i)!} \right) \det(\alpha^{\circ(\lambda + \delta)}) \det(\beta^{\circ(\lambda + \delta)}) \quad (\lambda = \bar{n} - \delta) \\ &= V(\alpha) V(\beta) \sum_{\lambda \in \mathbb{N}_{\leq}^D} \left(\prod_{i=1}^D \frac{f^{(\lambda_i + \delta_i)}(c)}{(\lambda_i + \delta_i)!} \right) s_{\lambda}(\alpha) s_{\lambda}(\beta). \end{aligned} \tag{3}$$

The last step follows from the algebraic definition of Schur polynomials. Similarly,

$$\det(M_{t-}^{s-}) = V(\alpha^{s-}) V(\beta^{t-}) \sum_{\lambda \in \mathbb{N}_{\leq}^{D-1}} \left(\prod_{i=1}^{D-1} \frac{f^{(\lambda_i + \delta_i)}(c)}{(\lambda_i + \delta_i)!} \right) s_{\lambda}(\alpha^{s-}) s_{\lambda}(\beta^{t-}). \tag{4}$$

6.1 Absolutely monotonic kernels

At this point one may try bounding (3) and (4) using analytic properties of Schur polynomials. However, for a general function f , the arbitrary signs of Taylor coefficients largely complicate the analysis, making tight bounds out of reach. Hence we first study the easy case where $\tau(M)$ is associated with an *absolutely monotonic* kernel f , i.e., $f : [0, 1] \rightarrow [0, 1]$ satisfies $f^{(n)}(c) \geq 0$ for all $n \in \mathbb{N}$ and $c > 0$. We will shortly see that such kernels are in fact not artificial as all the positive definite kernels naturally reduce to this case.

We first use the following identity of Schur polynomials to “align” (3) and (4).

► **Proposition 35.** *If $\lambda_1 \neq 0$, then $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1 = 0, \alpha_2, \dots, \alpha_n) = 0$. If $\lambda_1 = 0$, then $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1 = 0, \alpha_2, \dots, \alpha_n) = s_{(\lambda_2, \dots, \lambda_n)}(\alpha_2, \dots, \alpha_n)$.*

Proof. By the combinatorial definition of Schur polynomials, we have $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1, \alpha_2, \dots, \alpha_n) = \sum_{T \in \mathcal{T}} \alpha^{t(T)}$, where \mathcal{T} is the set of all SSYT of shape $\lambda = (\lambda_1, \dots, \lambda_n)$ on alphabet $[n]$, and $t(T)$ denotes the type of SSYT T . Now set $\alpha_1 = 0$. If the letter 1 appears in a SSYT

T , i.e. $t(T)[1] > 0$, then the corresponding monomial vanishes when evaluated. Therefore the equation simplifies to $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1, \alpha_2, \dots, \alpha_n) = \sum_{T \in \mathcal{T}_1} \alpha^{t(T)}$ where \mathcal{T}_1 is the set of all SSYT of shape λ on alphabet $\{2, 3, \dots, n\}$.

However, if $\lambda_1 > 0$, no sequence of length n on alphabet $\{2, 3, \dots, n\}$ satisfies the (strictly) decreasing constraint in the first column of the tableau. In this case $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1, \alpha_2, \dots, \alpha_n) = 0$. If $\lambda_1 = 0$, then \mathcal{T}_1 is essentially the set of all SSYT of shape $\lambda' = (\lambda_2, \dots, \lambda_n)$ over alphabet $\{2, 3, \dots, n\}$. By definition, $s_{(\lambda_1, \dots, \lambda_n)}(\alpha_1, \alpha_2, \dots, \alpha_n) = \sum_{T \in \mathcal{T}_1} \alpha^{t(T)} = s_{(\lambda_2, \dots, \lambda_n)}(\alpha_2, \dots, \alpha_n)$. \blacktriangleleft

In consequence, if we fix $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_D)$ with $\alpha_1 = 0$, then

$$\begin{aligned} \det(M) &= V(\alpha)V(\beta) \sum_{\lambda \in \mathbb{N}_{\leq}^D} \left(\prod_{i=1}^D \frac{f(\lambda_i + \delta_i)(c)}{(\lambda_i + \delta_i)!} \right) s_\lambda(\alpha) s_\lambda(\beta) \\ &= V(\alpha)V(\beta)f(c) \sum_{\lambda \in \mathbb{N}_{\leq}^{D-1}} \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_{i+1})(c)}{(\lambda_i + \delta_{i+1})!} \right) s_\lambda(\alpha^{1-}) s_\lambda(\beta^{1-}). \end{aligned}$$

Here $\lambda^{1-}, \alpha^{1-}, \beta^{1-}$ are obtained by removing the first entry in the corresponding partition/vector λ, α, β . (In the last step we abuse the notation by renaming λ^{1-} as λ .) Meanwhile,

$$\max_{s,t} \det(M_{t-}^{s-}) \leq \left(\max_{s,t} V(\alpha^{s-}) V(\beta^{t-}) \right) \sum_{\lambda \in \mathbb{N}_{\leq}^{D-1}} \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_i)(c)}{(\lambda_i + \delta_i)!} \right) s_\lambda(\alpha^{1-}) s_\lambda(\beta^{1-}).$$

In what follows, we let

$$E_\lambda = \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_{i+1})(c)}{(\lambda_i + \delta_{i+1})!} \right) s_\lambda(\alpha^{1-}) s_\lambda(\beta^{1-}), \quad F_\lambda = \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_i)(c)}{(\lambda_i + \delta_i)!} \right) s_\lambda(\alpha^{1-}) s_\lambda(\beta^{1-})$$

denote the corresponding terms in the two sums. For absolutely monotonic function f , we have $F_\lambda > 0$ and $E_\lambda > 0$ for all $\lambda \in \mathbb{N}_{\leq}^{D-1}$. Thus,

$$\max_{s,t} \frac{\det(M_{t-}^{s-})}{\det(M)} \leq \frac{1}{f(c)} \max_{s,t} \frac{V(\alpha^{s-}) V(\beta^{t-})}{V(\alpha)V(\beta)} \max_{\lambda \in \mathbb{N}_{\leq}^{D-1}} \frac{F_\lambda}{E_\lambda}. \quad (5)$$

6.2 KDE hardness for positive definite kernels

To accommodate positive definite kernels, we slightly modify the reduction in Section 3. Let $k(x, y) = f(\|x - y\|_2^2)$ be a positive definite kernel. By Schoenberg's characterization (Theorem 20), we have $(-1)^k \cdot f^{(k)}(t) \geq 0$ for all $k \in \mathbb{N}, t \geq 0$. Then for the function $g(x) = f(1 - x)$, it holds $g^{(k)}(t) = (-1)^k \cdot f^{(k)}(1 - t) \geq 0$ for all $k \in \mathbb{N}, t \in [0, 1]$. Namely g is an absolutely monotonic kernel.

Next we see how KDE_f can be related to $\tau(M_g)$. Let M be the $D \times D$ counting matrix associated with g , and let $W \in \mathbb{N}^{D \times n}$ be a re-indexed distance count matrix defined by $W[p, i] = \#\{j \in [n] : \|x^{(i)} - y^{(j)}\|_2^2 = D - p\}$. Then the matrix product $U = M \times W$ gives

$$\begin{aligned} U[\ell, i] &= \sum_{p=1}^D M[\ell, p] \cdot W[p, i] = \sum_{p=1}^D g(\alpha_\ell \beta_p) \sum_{j \in [n]} \mathbb{1} \left[\|x^{(i)} - y^{(j)}\|_2^2 = D - p \right] \\ &= \sum_{j \in [n]} g \left(c + \alpha_\ell \cdot \beta \left[D - \|x^{(i)} - y^{(j)}\|_2^2 \right] \right). \end{aligned} \quad (6)$$

If β is the identity vector, then

$$U[\ell, i] = \sum_{j \in [n]} g \left(c + \alpha_\ell \cdot (D - \|x^{(i)} - y^{(j)}\|_2^2) \right) = \sum_{j \in [n]} f \left(1 - c - \alpha_\ell D + \alpha_\ell \|x^{(i)} - y^{(j)}\|_2^2 \right).$$

Constructing vectors $\tilde{x}_\ell^{(i)}, \tilde{y}_\ell^{(j)} \in \mathbb{R}^{m+1}$ defined by $\tilde{x}_\ell^{(i)}[k] = \sqrt{\alpha(\ell)}x^{(i)}[k]$ if $k \in [m]$ and $\tilde{x}_\ell^{(i)}[m+1] = \sqrt{1-c-\alpha_\ell D}$; $\tilde{y}_\ell^{(j)}[k] = \sqrt{\alpha(\ell)}y^{(j)}[k]$ if $k \in [m]$ and $\tilde{y}_\ell^{(j)}[m+1] = 0$, we have $U[\ell, i] = \sum_{j \in [n]} f(\|\tilde{x}_\ell^{(i)} - \tilde{y}_\ell^{(j)}\|_2^2)$. In this way we show that the new product U can also be computed using KDE subroutines, and the reduction proceeds the same way as in Section 3 to recover the (re-indexed) distance count matrix. By similar analysis, we have the following hardness result relating the complexity of KDE_f to $\tau(M_g)$.

► **Theorem 36.** *Let $f : [0, 1] \rightarrow [0, 1]$ be a function and $g(x) = f(1-x)$. Then assuming SETH, for every $q > 0$, there exists $C \geq 0$ such that $\text{KDE}_f(n, m, (3n\tau(M))^{-1})$ cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$. if either of the following holds: (1) $m > C \log n, D = m$ or (2) $m > C^{\log^* n}, D > m^{C^{\log^* n} \cdot (\log n)/m}$. Here M is the $D \times D$ counting matrix associated with function g .*

6.3 Hardness of Rational Quadratic kernel

For f, α, β of certain forms, we give explicit bounds on the two terms in (5). Regarding the ratio of Vandermonde determinants, we focus on $V(x^{s^-})/V(x)$ for scaled identity x defined by $x_p = p/D$.

► **Proposition 37.** *Let $\rho \geq 1$ be a real number and $x \in [0, 1]^D$ be the vector defined by $x_i = i/D$. Then for $s \in [D]$ we have $\frac{V(x^{s^-})}{V(x)} = \prod_{i \in [D] \setminus \{s\}} |x_i - x_s|^{-1} \leq (3e)^D$.*

Proof. $\prod_{i \in [D] \setminus \{s\}} |x_i - x_s| = \prod_{i=1}^{s-1} \frac{s-i}{D} \cdot \prod_{i=s+1}^D \frac{i-s}{D} = \frac{(s-1)!(D-s)!}{D^{D-1}} \geq \left(\frac{D-1}{2eD}\right)^{(D-1)} \geq (3e)^{-D}$. ◀

Hence for vectors α defined by $\alpha_\ell = \ell/D, \ell \in [D]$ and β defined by $\beta_p = p/D, p \in [D]$, we have $\max_{s,t} \frac{V(\alpha^{s^-})V(\beta^{t^-})}{V(\alpha)V(\beta)} \leq (3e)^{2D}$.

For the term involving Taylor coefficients, we fix a real number $\sigma \geq 1$ and focus on the absolutely monotonic function $f(x) = (2-x)^{-\sigma}$. By calculation, for $k \in \mathbb{N}$, $f^{(k)}(x) = (2-x)^{-(\sigma+k)} \prod_{i=0}^{k-1} (\sigma+i)$. Then for $c=0$, we have

$$\frac{F_\lambda}{E_\lambda} = \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_i)(0)}{(\lambda_i + \delta_i)!} \right) / \left(\prod_{i=1}^{D-1} \frac{f(\lambda_i + \delta_{i+1})(0)}{(\lambda_i + \delta_{i+1})!} \right) = \prod_{i=1}^{D-1} \left(2 \cdot \frac{1 + \lambda_i + \delta_i}{\sigma + \lambda_i + \delta_i} \right) \leq 2^{D-1}.$$

Combining the calculations, we obtain the following bound on $\tau(M)$.

► **Theorem 38.** *Let $\sigma \geq 1$ be a real number. Let $f : [0, 1] \rightarrow [0, 1]$ be the function $f(x) = (2-x)^{-\sigma}$, and let vectors $\alpha, \beta \in \mathbb{R}^D$ be defined by $\alpha_\ell = \ell/D, \beta_p = p/D$. Then the $D \times D$ counting matrix M associated with f, α, β has $\tau(M) \leq D \cdot 2^\sigma \cdot (3e)^{2D} \cdot 2^{D-1} \leq 2^\sigma (7e)^{2D}$.*

Combining the bound on $\tau(M)$ associated with $f(x) = (1+(1-x))^{-\sigma}$ with the hardness of positive definite KDE Theorem 36, we obtain

► **Theorem 39 (Hardness of Rational Quadratic KDE).** *Let $f(x) = 1/(1+x)^\sigma$ be a Rational Quadratic kernel parameterized by $\sigma \geq 1$ an absolute constant. Then assuming SETH, for every $q > 0$, there exists $C_1, C_2, C_3, C'_3, C_4 \geq 0$ such that $\text{KDE}_f(n, m, \varepsilon)$ cannot be solved in time $O(n^{2-q})$ for any constant $q > 0$. if either of the following holds:*

- (1) $m > C_1 \log n, 1/\varepsilon > n^{C_2}$, or
 (2) $C_3 \log^* n < m < C'_3(\log n)/(\log \log n), \log \log(1/\varepsilon) > (\log n)/m \cdot (\log m) \cdot C_4^{\log^* n}$.
 Here C_1, C_3, C'_3 are absolute constants while C_2, C_4 are dependent on σ .

Proof. Analogous to the proof of Theorem 29. ◀

References

- 1 Amol Aggarwal and Josh Alman. Optimal-Degree Polynomial Approximations for Exponentials and Gaussian Kernel Density Estimation. In *37th Computational Complexity Conference (CCC 2022)*, 2022.
- 2 Josh Alman, Timothy Chu, Aaron Schild, and Zhao Song. Algorithms and hardness for linear algebra on geometric graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 2020. doi:10.1109/FOCS46700.2020.00057.
- 3 Josh Alman and Zhao Song. Fast attention requires bounded entries. In *NeurIPS*, 2023.
- 4 Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, October 2015. doi:10.1109/focs.2015.18.
- 5 Arturs Backurs, Moses Charikar, Piotr Indyk, and Paris Siminelakis. Efficient density evaluation for smooth kernels. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 615–626, 2018. doi:10.1109/FOCS.2018.00065.
- 6 Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/635440afdfc39fe37995fed127d7df4f-Paper.pdf.
- 7 Arturs Backurs, Piotr Indyk, and Tal Wagner. Space and time efficient kernel density estimation in high dimensions. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2019. Curran Associates Inc.
- 8 Moses Charikar, Michael Kapralov, Navid Nouri, and Paris Siminelakis. Kernel density estimation through density constrained near neighbor search. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 172–183, 2020. doi:10.1109/FOCS46700.2020.00025.
- 9 Moses Charikar, Michael Kapralov, and Erik Waingarten. A quasi-monte carlo data structure for smooth kernel evaluations. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5118–5144, 2024. doi:10.1137/1.9781611977912.184.
- 10 Moses Charikar and Paris Siminelakis. Hashing-based-estimators for kernel density in high dimensions. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1032–1043, 2017. doi:10.1109/FOCS.2017.99.
- 11 Moses Charikar and Paris Siminelakis. Multi-resolution hashing for fast pairwise summations. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 769–792, 2019. doi:10.1109/FOCS.2019.00051.
- 12 Lijie Chen. On the hardness of approximate and exact (bichromatic) maximum inner product. *Theory of Computing*, 16(4):1–50, 2020. doi:10.4086/toc.2020.v016a004.
- 13 Yen-Chi Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1(1):161–187, 2017.
- 14 L Greengard and V Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73(2):325–348, 1987. doi:10.1016/0021-9991(87)90140-9.
- 15 Leslie Greengard and John A. Strain. The fast gauss transform. *SIAM J. Sci. Comput.*, 12:79–94, 1991. URL: <https://api.semanticscholar.org/CorpusID:3145209>.

- 16 Sean Hallgren, Alexander Russell, and Amnon Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, STOC '00, pages 627–635, New York, NY, USA, 2000. Association for Computing Machinery. doi:10.1145/335305.335392.
- 17 Christian Ikenmeyer and Greta Panova. Rectangular kronecker coefficients and plethysms in geometric complexity theory. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 396–405, 2016. doi:10.1109/FOCS.2016.50.
- 18 Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.
- 19 Ryan O'Donnell and John Wright. Quantum spectrum testing. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 529–538, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2746539.2746582.
- 20 Jeff M. Phillips and Wai Ming Tai. Near-optimal coresets of kernel density estimates. *Discrete Comput. Geom.*, 63(4):867–887, June 2020. doi:10.1007/s00454-019-00134-6.
- 21 Aviad Rubinfeld. Hardness of approximate nearest neighbor search. *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, June 2018. doi:10.1145/3188745.3188916.
- 22 Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- 23 Richard Stanley. *Enumerative Combinatorics*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2 edition, 2023.
- 24 Paraskevas Syminelakis. *Fast kernel evaluation in high dimensions: Importance sampling and near neighbor search*. PhD thesis, Stanford University, 2019.
- 25 Ryan Williams. On the difference between closest, furthest, and orthogonal pairs: Nearly-linear vs barely-subquadratic complexity. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2018.

Gap MCSP Is Not (Levin) NP-Complete in Obfustopia

Noam Mazor ✉

Tel Aviv University, Israel

Rafael Pass ✉

Tel Aviv University, Israel

Cornell Tech, New York, NY, USA

Abstract

We demonstrate that under believable cryptographic hardness assumptions, Gap versions of standard meta-complexity problems, such as the Minimum Circuit Size Problem (MCSP) and the Minimum Time-Bounded Kolmogorov Complexity problem (MKTP) are not **NP**-complete w.r.t. Levin (i.e., witness-preserving many-to-one) reductions.

In more detail:

- Assuming the existence of indistinguishability obfuscation, and subexponentially-secure one-way functions, an appropriate Gap version of MCSP is not **NP**-complete under randomized Levin-reductions.
- Assuming the existence of subexponentially-secure indistinguishability obfuscation, subexponentially-secure one-way functions and injective PRGs, an appropriate Gap version of MKTP is not **NP**-complete under randomized Levin-reductions.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Kolmogorov complexity, MCSP, Levin Reduction

Digital Object Identifier 10.4230/LIPIcs.CCC.2024.36

Related Version *Full Version:* <https://eprint.iacr.org/2024/420> [52]

Funding *Noam Mazor:* Research partly supported by NSF CNS-2149305 and DARPA under Agreement No. HR00110C0086.

Rafael Pass: Supported in part by AFOSR Award FA9550-23-1-0387, AFOSR Award FA9550-23-1-0312, and an Algorand Foundation grant. This material is based upon work supported by DARPA under Agreement No. HR00110C0086. 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 United States Government, DARPA, AFOSR or the Algorand Foundation.

1 Introduction

As described by Trakhtenbrot [62], starting in the 1960s, there has been an on-going effort studying the computational complexity of so-called “meta-complexity” problems; notably (a) the *Minimum Circuit Size problem* (MCSP) [37, 62] – determining the size of the smallest Boolean circuit that computes a given function x , and (b) the *Time-Bounded Kolmogorov Complexity Problem* (MKTP) [41, 61, 16, 39, 26, 60] – determining the length, denoted $K^t(x)$ of the shortest program (evaluated on some particular Universal Turing machine U) that generates a given string x , within time t , where $t = \text{poly}(|x|)$ is a polynomial. In particular, a major problem since the 1960s is whether these problems, or the Gap versions of them (where the goal is to determine whether the size is above a threshold s_2 or below a threshold s_1) are **NP**-complete. Indeed, as recounted by [4, 30, 31], Levin is said to have delayed the publication of his theory of **NP**-completeness [44] in order to show **NP**-completeness of MCSP.



© Noam Mazor and Rafael Pass;
licensed under Creative Commons License CC-BY 4.0
39th Computational Complexity Conference (CCC 2024).

Editor: Rahul Santhanam; Article No. 36; pp. 36:1–36:21

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



In the following decades, there has been a lot of amazing progress – providing evidence pointing towards *both* a positive and a negative answer:

Towards NP-completeness: While it is still unknown whether the original problems are NP-complete, several generalizations of them have been proven to be NP-complete. Most notably, Ilango first demonstrated this for an oracle version of MCSP [30]; this was subsequently extended to a multi-bit version of MCSP referred to as Multi-MCSP [32], to a conditional version of the MKTP problem, McKTP [51], and to other variants [27]. [29] recently improved the parameters of the reduction to McKTP [51], assuming that witness encryption scheme exists. Additionally, Ilango [31] very recently demonstrates that NP-hardness of a variant of MCSP and MKTP where the programs are allowed to access a random oracle, yielding a *heuristic* NP-completeness Karp (i.e., many-one) reduction for these problems (if instantiating the random oracle with a concrete hash function). Finally, a recent work by Impagliazzo, Kabanets, and Volkovich [33] provides various different results that can be interpreted as giving evidence that MCSP is NP-complete with respect to randomized reductions.

Towards Non NP-completeness: There is also evidence pointing towards non NP completeness: Allender and Hirahara [3] showed that assuming one-way functions, the gap version of MCSP is not NP complete for super-polynomial gap. Ko [40] showed that a version of MKTP is not NP complete with respect to an oracle, and Ren and Santhanam [56] gave an oracle with respect to which MCSP is not NP complete. Other works prove limitations on the structure of reduction to meta-complexity problems. Murray and Williams [53] prove that MCSP is not NP complete under so-called *local reductions*. Kabanets and Cai [37] and Saks and Santhanam [58] show that the NP-completeness of MCSP under Turing reductions with certain properties implies circuit lower bounds. For example if MCSP is complete under so-called *parametric honest* Turing reductions, then $\mathbf{E} \not\subseteq \mathbf{SIZE}(\text{poly})$. More recently, Saks and Santhanam [59] gave evidence that the running time of any randomized non-adaptive reduction from SAT to K^t approximation must grow with the time parameter t . These results, however, only rule out quite limited types of reductions.

Despite this progress, the original question, however, remains wide open.

1.1 Our Results

The current paper provides strong evidence that the Gap versions of MCSP and MKTP are not NP-complete w.r.t. *Levin reductions* – that is *witness-preserving* many-to-one reductions. In particular, we demonstrate that under somewhat strong, but generally believed, cryptographic hardness assumptions, the Gap version of MCSP is not NP-complete w.r.t. Levin reductions.

Levin Reductions

The three original ways [17, 38, 45] of defining NP completeness differ in how reductions from a language L to a language L' are defined (see e.g., [24] for a discussion). Cook [17] considers the most permissive notion: a Turing machine deciding L having oracle access to a decider for L' . Karp's notion – called a *Karp reduction* (or *many-one reduction*) is more restrictive: it requires efficiently mapping an instance x into an instance x' such that $x \in L$ iff $x' \in L'$. Levin's notion, called a *Levin reduction* (or a *witness preserving many-one reduction*) is the most restrictive: it additionally requires *efficiently* mapping any witness w for x into a witness for x' , and furthermore any witness w' for x' into a witness w for x . While Karp

reductions are most commonly used, as far as we are aware, most natural **NP**-completeness reductions are actually of the Levin type as well. Furthermore, for *constructive applications* of **NP**-completeness, **NP**-hardness demonstrated using a Levin reduction is typically what is needed: In particular, for cryptographic application to interactive proofs (e.g., demonstrating that every language in **NP** has a zero-knowledge proof of knowledge [19], or that every language in **NP** has a succinct argument [11], the notion of a Levin reduction is crucial (see e.g., [11] that in particular notes that even the most sophisticated **NP** completeness reductions, as those provided by the PCP theorem [9, 10], are Levin reductions). Our focus here is on such Levin reductions; in particular, we will present the (conditional) impossibility of Levin reductions for demonstrating **NP**-completeness; in fact, our impossibility will apply not only to deterministic but also *randomized* Levin reductions (where the reduction is allowed to fail with some small constant probability).

We mention that e.g., the **NP**-completeness results of [31] and [51] rely on the **NP**-completeness of approximation for the *Set-Cover* problem [18, 63]. In both works, the reductions from Set-Cover to the GapMCSP and Gap_pMK^tP (or the conditional version in the case of [51]) are (randomized) Levin reductions (see the full version of this paper for a discussion of the result of [31]). The Set-cover **NP**-completeness itself relies on a long sequence of the reductions that we have not been able to verify whether they are all Levin (although, as mentioned above, the main technical core, the PCP theorem, is).

Our Cryptographic Hardness Assumptions: Indistinguishability Obfuscation

We will rely on the existence of *indistinguishability obfuscation* (*iO*) for circuits [12]. Roughly speaking, an indistinguishability obfuscator is an efficient algorithm *iO* that given a circuit *C* outputs an “obfuscated” version of *C* having the property that obfuscations of any two functionally equivalent circuits are indistinguishable. Following the ground-breaking work of [20], several heuristic candidates were proposed, as well as provably secure constructions based on various assumptions [55, 23, 46, 64, 49, 50, 47, 6, 35, 35, 5, 21, 2, 1]. Most notable, the recent breakthrough result presents a construction based on several well-founded (and generally believed) hardness assumption [36]. (Constructions based on less standard, but seemingly quantum-safe, “circular-security” assumptions also appear in [15, 22, 14]).

For our main results on MCSP, we will simply rely on indistinguishability obfuscation and subexponentially-secure one-way function. For our results on MKTP, we will rely on *iO* with subexponential security as well as other standard cryptographic assumptions such as injective pseudorandom generators (PRGs), that e.g., are implied by the existence of one-way permutations.

Main Theorem

We present the following main result:

- Assuming the existence of indistinguishability obfuscation and subexponentially-secure one-way function, an appropriate Gap version of MCSP is not **NP**-complete under randomized Levin-reductions.
- Assuming the existence of subexponentially-secure indistinguishability obfuscation, subexponentially-secure one-way function and injective PRGs, an appropriate Gap version of MKTP is not **NP**-complete under randomized Levin-reductions.

In more detail, let GapMCSP_[s₀, s₁] be the promise problem in which given a truth table *x* we need to distinguish between the following two cases:

- **Yes instances:** There exists a circuit *C* of size at most $s_0(|x|)$ that computes *x*.
- **No instances:** There is no circuit of size $s_1(|x|)$ that computes *x*.

Our first theorem states that when the gap between s_0 and s_1 is large enough, and under cryptographic assumptions, $\text{GapMCSP}[s_0, s_1]$ is not **NP**-complete with respect to Levin reductions.

► **Theorem 1.** *Assume that iO and subexponentially-secure one-way functions exist. Then there exists a polynomial p , such that for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n))$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not **NP** complete with respect to Levin reductions.*

We remark that if all of the assumed cryptographic primitives are secure against *sub-exponential adversaries* (in contrast to just polynomial adversaries), then our results rule out also randomized Levin reductions that run in sub-exponential time.

Additionally, the assumption of subexponentially-secure one-way functions in Theorem 1 is only to handle so-called non *honest* reductions: A Karp reduction f is to be *honest* if for every $x \in \{0, 1\}^*$, $|f(x)| \geq |x|^\delta$ for some constant $\delta > 0$ (i.e., the mapping from statements x to x' is polynomially preserving).

To exclude only honest reductions, it is enough to assume one-way function with polynomial security. Such one-way functions are known to exist assuming iO and the minimal assumption that $\text{NP} \notin \text{ioBPP}$ [42]. We get the following theorem.

► **Theorem 2.** *Assume that iO exists, and that $\text{NP} \not\subseteq \text{ioBPP}$. Then there exists a polynomial p , such that for every $\epsilon > 0$, for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n))$ and $s_0(n) > n^\epsilon$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not **NP** complete with respect to honest Levin reductions.*

Our second result is a similar result for the $\text{Gap}_p\text{MK}^t\text{P}$ problem. Recall that $K^t(x)$ is the minimal length of a program that outputs x within $t(|x|)$ steps. For polynomials t and p , let $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ be the promise problem in which given a string x we need to distinguish between the following two cases:

- **Yes instances:** $K^t(x) \leq s_0(|x|)$
- **No instances:** $K^{p(t)}(x) > s_1(|x|)$.

We prove the following theorem.

► **Theorem 3.** *Assume that subexponentially-secure iO , subexponentially-secure one-way functions and injective PRG exist. Then there exist a polynomial q such that for any $t \in \text{poly}$ and any efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > q(\log t(n), s_0(n))$, and for every large enough polynomial p , it holds that $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ is not **NP** complete with respect to Levin reductions.*

Achieving a smaller gap under stronger assumptions

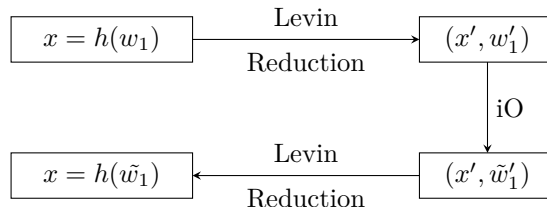
As discussed above, several generalizations of the GapMCSP and $\text{Gap}_p\text{MK}^t\text{P}$ problem have been proven **NP** complete. The work of [31] showed that the same problems we consider here are **NP** complete relative to a random oracle. There, the gap between the Yes and No instances is a multiplicative $(1 + \epsilon)$ gap, for a small constant $\epsilon > 0$ while in the theorems above we need the gap to be larger. Similarly, [51] showed that deciding a *conditional* version of MKTP is **NP**-hard, and their result can be generalized to a gap problem with a larger constant multiplicative factor. Hirahara [28] used a reduction from the Minimum Monotone Satisfying Assignment problem to McKTP , resulting with a **NP**-hardness of the GapMcKTP with a larger multiplicative gap, but still sub polynomial in the input length ($n^{o(1)}$).

The polynomial p in Theorems 1 and 2 is the *overhead* of the iO algorithm we use. By assuming a stronger assumption – that iO with a small overhead exists – we can improve the gap. For example, we say that iO has additive overhead if on input C and security parameter λ , the size of the obfuscated circuit is $|C| + \text{poly}(\lambda)$. If we assume iO with additive overhead, we would get the hardness of GapMCSP also for the additive gap case. Unfortunately, such iO constructions are currently not known (but as far as we know, there are also no results indicating that this should be impossible). However, if we consider slightly stronger assumptions, we can get iO for TM with a factor $2 + \epsilon$ overhead (for any constant $\epsilon > 0$) [8], yielding the following theorem.¹

► **Theorem 4.** *Assume subexponential-secure iO , and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every very constant $\epsilon > 0$, for every large enough polynomial p , and for every efficiently computable function s_0 it holds that $\text{Gap}_p\text{MK}^t\text{P}[s_0, (2 + \epsilon)s_0(n)]$ is not **NP** complete with respect to Levin reductions.*

Proof Overview

In this proof outline, we will for simplicity focus on ruling out deterministic Levin reductions for the GapMCSP problem. Additionally, on top of the existence of iO , we will here assume the existence of a collision-resistant hash function; that is the existence of a family of compressing functions such that for a randomly sampled h , it is infeasible to find two inputs x_1, x_2 that “collide” (i.e., $h(x_1) = h(x_2)$) although such collision exists. (In our actual proof, we instead rely on the weaker primitive of a target collision-resistant hash function (TCR; also known as, universal one-way hash function [54]) which can be constructed from one-way functions [57]. Finally, let us start by assuming that the reduction is “honest” (i.e., mapping statements x to statements x' of polynomially-related length.



■ **Figure 1** The proof overview. Given a witness w_1 such that $h(w_1) = x$, we use the Levin reduction to get MCSP witness. Then we use the iO to get a new MCSP witness, and use the Levin reduction again to get back \tilde{w}_1 such that $h(\tilde{w}_1) = x$.

The key idea will be to use the Levin reduction and the iO in order to find a collision for h . Roughly speaking, we start by sampling some w_1 and compute $x = h(w_1)$; we think of x as a statement for the language of images of h , and of w_1 as the witness for x . We next use the Levin reduction to get an MCSP statement x' and its corresponding witness w'_1 . Note that the witness w'_1 is a circuit computing x' . We then *obfuscate* w'_1 using the iO to get a *new* witness \tilde{w}'_1 for x' . Using the Levin reduction, we can finally turn \tilde{w}'_1 into a (hopefully new) witness \tilde{w}_1 for x . Indeed, the key point is that if we had started with a

¹ In a previous version of this paper, we claimed a similar result for GapMCSP using iO for circuits with a factor $2 + \epsilon$ overhead. iO with such small overhead w.r.t. circuits does not appear to be known; while [8] claim an iO where the size of an obfuscation of a circuit C is of length $2|C| + \text{poly}(\lambda)$, it appears that this “program” may need to be further interpreted, which may result in larger circuit size.

different preimage $w_2 \neq w_1$ for $x = h(w_1)$ and done the same process, then w'_2 would become a functionally equivalent circuit to w'_1 and thus by the security of the iO , the distributions of \tilde{w}'_2 and \tilde{w}'_1 are computationally indistinguishable, so we conclude that \tilde{w}_2 and \tilde{w}_1 also are. In particular, it follows that $\tilde{w}_1 \neq w_1$ with probability at least $1/2$, and we have thus found a collision.

Note that we here rely on the **NP**-completeness of the Gap version of the MCSP problem since when applying the iO we get a new witness for x' but this witness (i.e. the circuit) is *bigger* than the original one. In particular, the overhead of the iO translates into the gap of the problem – for instance, if the overhead of the iO is only linear, we can handle a linear gap, and if it has polynomial overhead then we can only rule out reductions for the polynomial gap version of the problem.

Dealing with Non-honest Reductions

If the reduction is not honest, the statement x' could be a lot shorter than x ; the problem then becomes if we run the iO on a security parameter that is polynomially related to $|x'|$ (which we require to ensure that we stay within the promise), we may no longer have security with respect to an attacker who runs in time polynomial in $|x| = n$ (which is required to ensure that we find a collision). However, if we start off with a collision-resistant hash function with sub-exponential security (i.e., 2^{n^ϵ} security), we can resolve this problem using a case-analysis. If $|x'| \leq n^\epsilon$, then we simply find a new witness \tilde{w}' using *brute-force search*, and otherwise use the iO . This ensures that we only run the iO in case the reduction behaves "honestly"; on the other hand, when the reduction chooses a short x' , we still contradict the subexponential security of the collision-resistant hash function.

Extensions for $\text{Gap}_p\text{MK}^t\text{P}$

We next generalize the above proof for the $\text{Gap}_p\text{MK}^t\text{P}$ problem. To be able to do so, we need a way to move from one $\text{Gap}_p\text{MK}^t\text{P}$ witness to another, when a $\text{Gap}_p\text{MK}^t\text{P}$ witness is a t -time TM P of size at most $s_0(|x|)$ that outputs x . A naive approach is to first convert the TM P into a circuit, then apply the iO for circuits, and lastly, convert the circuit back to a TM. The problem in this approach is that since the program P outputs x , the time bound t must be at least $|x|$. This means that the circuit we construct from P will have a trivial size, and we will not be able to get back a non-trivial program that outputs x .

Luckily, we can use iO for TMs directly on P , or even it suffices to rely on a weaker primitive of a *randomized encoding*. Randomized encoding for TMs is known to exist assuming subexponential-secure iO for circuits and injective PRGs [43, 48].

Discussion

The results presented yield give a strong evidence that the GapMCSP and $\text{Gap}_p\text{MK}^t\text{P}$ are not **NP**-complete w.r.t. Levin reductions, at least when the gap is at least a factor 2. Furthermore, although there are no known constructions of iO with only additive overhead based on well-founded hardness assumptions, one can come up with candidate constructions with only linear additive overhead and heuristically assume that they satisfy the notion of indistinguishability obfuscation.² Under these more heuristic assumptions (which in our eyes

² In particular, take the constructions from e.g. [13, 8] and instead of encrypting the program *twice* under an FHE with additive linear overhead, simply encrypt the program once. While the two encryptions are needed for the security proof, the construction without the two encryptions seems heuristically secure.

seem reasonable), our results thus give evidence that these problems are not **NP**-complete w.r.t. Levin reductions even when the gap is a small additive term. These results thus provide (in our eyes) convincing cryptographic evidence that the original task set out by Levin is impossible (since he indeed defined **NP**-completeness through the notion of what today is referred to as a Levin reduction.)

Of course, it could still be that a weaker notion of a reduction (e.g., a Karp) reduction can be used to prove **NP**-completeness of these problems. In particular, consider the results of [31], which shows **NP**-completeness of GapMCSP in the random oracle model. While, as discussed, his reduction from (approximate) Set-Cover to GapMCSP is a Levin reductions (see the full version of this paper), the witness preserving part of the reduction relies on the random oracle – in particular, the witness reconstruction step relies on observing the queries to the random oracle performed by the circuit \tilde{w}' (i.e., the witness for the transformed statement x').³ If instantiating the random oracle with a concrete hashfunction h , it is no longer clear how to perform this task – in particular if the circuit has been obfuscated so that it (intuitively) becomes hard to find the code of h in the description of the circuit. As such, when instantiating the random oracle with a hashfunction, the reduction most likely is no longer a Levin reduction, but conceivably it could still be a Karp reduction.

In contrast, as was shown in [34], if iO exists and $\text{MCSP} \in \mathbf{BPP}$ (and using similar ideas, even if GapMCSP or GapMKTP with polynomial gap are in **BPP**), then $\mathbf{NP} \subseteq \mathbf{BPP}$. Indeed, if $\text{GapMCSP}[n^\epsilon, n^{1-\epsilon}] \in \mathbf{BPP}$ then (infinitely-often) one-way functions do not exist, and thus by the result of [42], $\mathbf{NP} \subseteq \mathbf{BPP}$. This result gives, assuming obfuscation, a randomized reduction from **NP** to GapMCSP. This reduction however is not a Karp (or Levin) reduction.

2 Preliminaries

2.1 Notations

All logarithms are taken in base 2. We use calligraphic letters to denote sets and distributions, uppercase for random variables, and lowercase for values and functions. Given a set $\mathcal{S} \subseteq \{0, 1\}^*$, we let $\overline{\mathcal{S}} = \{0, 1\}^* \setminus \mathcal{S}$. Let poly stand for the set of all polynomials. Let PPT stand for probabilistic poly-time, and n.u.-poly-time stand for non-uniform poly-time. An n.u.-poly-time algorithm A is equipped with a (fixed) poly-size advice string set $\{z_n\}_{n \in \mathbb{N}}$. Let neg stand for a negligible function. For a SAT formula ϕ over n variables and an assignment $v \in \{0, 1\}^n$, we use $\phi[v] \in \{0, 1\}$ to denote the truth value of the evaluation of ϕ on v .

2.2 Distributions and Random Variables

When unambiguous, we will naturally view a random variable as its marginal distribution. The support of a finite distribution \mathcal{P} is defined by $\text{Supp}(\mathcal{P}) := \{x : \Pr_{\mathcal{P}}[x] > 0\}$. For a (discrete) distribution \mathcal{P} , let $x \leftarrow \mathcal{P}$ denote that x was sampled according to \mathcal{P} . Similarly, for a set \mathcal{S} , let $x \leftarrow \mathcal{S}$ denote that x is drawn uniformly from \mathcal{S} .

2.3 Kolmogorov Complexity

Roughly speaking, the t -time-bounded Kolmogorov complexity, $K^t(x)$, of a string $x \in \{0, 1\}^*$ is the length of the shortest program $\Pi = (M, y)$ such that, when simulated by a universal Turing machine, Π outputs x in $t(|x|)$ steps. Here, a program Π is simply a pair of a Turing

³ Interestingly, a similar method of observing the queries to the random oracle was used by [25] to show that there is no obfuscation for circuits with oracle access to a random oracle.

Machine M and an input y , where the output of Π is defined as the output of $M(y)$. When there is no running time bound (i.e., the program can run in an arbitrary number of steps), we obtain the notion of Kolmogorov complexity.

In the following, let $U(\Pi, 1^t)$ denote the output of Π when emulated on U for t steps. We now define the notion of Kolmogorov complexity with respect to the universal TM U .

► **Definition 5.** Let $t \in \mathbb{N}$ be a number. For all $x \in \{0, 1\}^*$, define

$$K_U^t(x) = \min_{\Pi \in \{0, 1\}^*} \{|\Pi| : U(\Pi, 1^t) = x\}$$

where $|\Pi|$ is referred to as the description length of Π .

It is well known that for every x , $K^t(x) \leq |x| + c$, for some constant c depending only on the choice of the universal TM U .

► **Fact 6.** For every universal TM U , there exists a constant c such that for every $x \in \{0, 1\}^*$, and for every t such that $t(n) > 0$, $K_U^t(x) \leq |x| + c$.

In the following we fix some universal TM U and omit it from the notation.

2.4 Levin Reductions

For a relation $\mathcal{R} \subseteq \{0, 1\}^* \times \{0, 1\}^*$, let $\mathcal{L}(\mathcal{R}) = \{x \in \{0, 1\}^* : \exists w \in \{0, 1\}^* \text{ s.t. } (x, w) \in \mathcal{R}\}$. We say that a relation \mathcal{R} is the witness relation of a language $\mathcal{L} \subseteq \{0, 1\}^*$ if $\mathcal{L}(\mathcal{R}) = \mathcal{L}$.

► **Definition 7 (Levin reduction).** Let \mathcal{R}_1 and \mathcal{R}_2 be relations. A triplet of efficiently computable functions (f, g, h) is a Levin reduction from \mathcal{R}_1 to \mathcal{R}_2 if

- For every $(x, w) \in \mathcal{R}_1$, $(f(x), g(x, w)) \in \mathcal{R}_2$.
- If $(f(x), w) \in \mathcal{R}_2$ then $(x, h(x, w)) \in \mathcal{R}_1$.

► **Remark 8.** Notice that if (f, g, h) a Levin reduction from \mathcal{R}_1 to \mathcal{R}_2 , then f is a Karp reduction from $\mathcal{L}(\mathcal{R}_1)$ to $\mathcal{L}(\mathcal{R}_2)$. Indeed, the first item above implies that if $x \in \mathcal{L}(\mathcal{R}_1)$ then $f(x) \in \mathcal{L}(\mathcal{R}_2)$, and the second item implies the other direction.

A Levin reduction (f, g, h) is *honest* if there exists a constant $\delta > 0$ such that for every large enough $n \in \mathbb{N}$ and every $x \in \{0, 1\}^n$, $f(x) \geq n^\delta$.

When for two languages \mathcal{L}_1 and \mathcal{L}_2 we fix canonical relations \mathcal{R}_1 and \mathcal{R}_2 , we say that there is a Levin reduction from \mathcal{L}_1 to \mathcal{L}_2 if there is a Levin reduction from \mathcal{R}_1 to \mathcal{R}_2 . We say that $\mathcal{L} \in \mathbf{NP}$ is **NP** complete under Levin reductions if there exists a Levin reduction from SAT to \mathcal{L} , where the canonical relation for SAT is

$$\mathcal{R}_{\text{SAT}} = \{(\phi, v) : \phi \text{ is a SAT formula and } \phi[v] = 1\}.$$

We also define Levin reductions for promise problems. In the following, we consider promise problem $(\mathcal{Y}, \mathcal{N})$ that is associated with two relations $(\mathcal{R}_{\mathcal{Y}}, \mathcal{R}_{\overline{\mathcal{N}}})$ such that $\mathcal{R}_{\mathcal{Y}} \subseteq \mathcal{R}_{\overline{\mathcal{N}}}$, where $\mathcal{R}_{\mathcal{Y}}$ is the witness relation for \mathcal{Y} , and $\mathcal{R}_{\overline{\mathcal{N}}}$ is the witness relation for $\overline{\mathcal{N}}$. That is, $(\mathcal{Y}, \mathcal{N}) = (\mathcal{L}(\mathcal{R}_{\mathcal{Y}}), \overline{\mathcal{L}(\mathcal{R}_{\overline{\mathcal{N}}})})$.

► **Definition 9 (Levin reduction, promise problems).** Let $(\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\overline{\mathcal{N}}}^1)$ and $(\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\overline{\mathcal{N}}}^2)$ be pairs of relations such that $\mathcal{R}_{\mathcal{Y}}^1 \subseteq \mathcal{R}_{\overline{\mathcal{N}}}^1$ and $\mathcal{R}_{\mathcal{Y}}^2 \subseteq \mathcal{R}_{\overline{\mathcal{N}}}^2$. A triplet of efficiently computable functions (f, g, h) is a Levin reduction from $(\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\overline{\mathcal{N}}}^1)$ to $(\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\overline{\mathcal{N}}}^2)$ if

- For every $(x, w) \in \mathcal{R}_{\mathcal{Y}}^1$, $(f(x), g(x, w)) \in \mathcal{R}_{\mathcal{Y}}^2$.
- If $(f(x), w) \in \mathcal{R}_{\overline{\mathcal{N}}}^2$ then $(x, h(x, w)) \in \mathcal{R}_{\overline{\mathcal{N}}}^1$.

Note that we can define reductions from language to promise problem by taking $\mathcal{R}_y = \mathcal{R}_{\overline{N}}$. Lastly, our results hold even when the reductions are allowed to be randomized. In this case, $f(x; r)$ can be a randomized function (that uses randomness r), and both g, h get access to r (and possibly use more randomness). We then only require that the above requirements hold with high probability over r .

► **Definition 10** (Randomized Levin reduction, promise problems). *Let $(\mathcal{R}_y^1, \mathcal{R}_{\overline{N}}^1)$ and $(\mathcal{R}_y^2, \mathcal{R}_{\overline{N}}^2)$ be pairs of relations such that $\mathcal{R}_y^1 \subseteq \mathcal{R}_{\overline{N}}^1$ and $\mathcal{R}_y^2 \subseteq \mathcal{R}_{\overline{N}}^2$. A triplet of efficiently computable functions (f, g, h) is a randomized Levin reduction with ϵ -error from $(\mathcal{R}_y^1, \mathcal{R}_{\overline{N}}^1)$ to $(\mathcal{R}_y^2, \mathcal{R}_{\overline{N}}^2)$ if*

■ *For every $x \in \mathcal{L}(\mathcal{R}_y^1)$, with probability at least $1 - \epsilon$ over the choice of r_1 the following holds:*

1. $(f(x; r_1), g(x, w; r_1)) \in \mathcal{R}_y^2$, and,
2. for every w' such that $(f(x; r_1), w') \in \mathcal{R}_{\overline{N}}^2$ it holds that

$$\Pr_{r_2 \leftarrow \{0,1\}^*} \left[(x, h(x, w'; r_1, r_2)) \in \mathcal{R}_{\overline{N}}^1 \right] \geq 1 - \epsilon.$$

■ *For every $x \notin \mathcal{L}(\mathcal{R}_{\overline{N}}^1)$ it holds that $\Pr_{r_1 \leftarrow \{0,1\}^*} \left[f(x; r_1) \in \mathcal{L}(\mathcal{R}_{\overline{N}}^2) \right] \leq \epsilon$.*

2.5 Cryptographic Primitives

In this part we define the cryptographic tools we will use. We start with the definition of one-way function.

► **Definition 11** (One-way function). *A polynomial-time computable function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is called a one-way function if for every PPT algorithm A , there is a negligible function $\mu : \mathbb{N} \rightarrow [0, 1]$ such that for every $n \in \mathbb{N}$*

$$\Pr_{x \leftarrow \{0,1\}^n} [A(f(x)) \in f^{-1}(f(x))] \leq \mu(n).$$

A one-way function is subexponentially-secure if there exists a constant $\delta > 0$ such that for every 2^{n^δ} time algorithm A , and for every large enough $n \in \mathbb{N}$

$$\Pr_{x \leftarrow \{0,1\}^n} [A(f(x)) \in f^{-1}(f(x))] \leq 2^{-n^\delta}.$$

Next, we define iO .

► **Definition 12** (indistinguishability obfuscation). *An efficiently randomized algorithm iO is an indistinguishability obfuscator if for every $\lambda, n \in \mathbb{N}$ and any circuit $C : \{0, 1\}^n \rightarrow \{0, 1\}$,*

$$\Pr_{\widehat{C} \leftarrow iO(1^\lambda, C), x \leftarrow \{0,1\}^n} [C(x) = \widehat{C}(x)] = 1,$$

and for every $s \in \text{poly}$ and every $n.u.$ -poly-time algorithm \mathcal{A} , there exists a negligible function μ , such that for every $\lambda \in \mathbb{N}$ and every two circuit $C, C' : \{0, 1\}^n \rightarrow \{0, 1\}$ with $|C| = |C'| \leq s(\lambda)$ and $n \leq \lambda$,

$$\left| \Pr[\mathcal{A}(1^\lambda, iO(1^\lambda, C)) = 1] - \Pr[\mathcal{A}(1^\lambda, iO(1^\lambda, C')) = 1] \right| \leq \mu(\lambda).$$

We say that iO has overhead p if for every C and λ , $|iO(1^\lambda, C)| \leq p(|C|, \lambda)$ with probability 1.

Next we define Target collision-resistant hash functions, also known as universal one-way hash functions.

36:10 Gap MCSP Is Not (Levin) NP-Complete in Obfustopia

► **Definition 13** (Target collision resistant hash). *An efficiently computable function*

$$T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-s(n)}$$

is a Target collision resistant hash function (TCR) if $s(n) \geq 1$ and for every PPT algorithm \mathcal{A} ,

$$\Pr_{x \leftarrow \{0, 1\}^n} [x' \leftarrow \mathcal{A}(x); T(x) = T(x') \text{ and } x \neq x'] = \text{neg}(n).$$

We say that a TCR is secure against subexponential adversaries if there exists a constant $\delta > 0$ such that for every 2^{n^δ} time algorithm \mathcal{A} ,

$$\Pr_{x \leftarrow \{0, 1\}^n} [x' \leftarrow \mathcal{A}(x); T(x) = T(x') \text{ and } x \neq x'] = \text{neg}(n).$$

Rompel [57] showed that TCR can be constructed from one-way functions.

► **Theorem 14** ([57]). *Assume that one-way functions exist. Then TCR $T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-s(n)}$ with $s(n) \in \omega(\log n)$ exists.*

Since the proof of the theorem above is black-box, the same holds for subexponential adversaries.

► **Theorem 15.** *Assume that subexponentially-secure one-way functions exist. Then there exists a TCR $T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-s(n)}$ secure against subexponential adversaries, with $s(n) \in \omega(\log n)$.*

We will also use the following theorem, by [42].

► **Theorem 16** ([42]). *Assume that iO exists and $\text{NP} \not\subseteq \text{ioBPP}$. Then one-way functions exist.*

Lastly, we will also use the fact that a TCR is a one-way function.

▷ **Claim 17.** Let $T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-s(n)}$ be a TCR with $s(n) \in \omega(\log n)$. Then T is a one-way function. That is, for every PPT algorithm \mathcal{A} ,

$$\Pr_{x \leftarrow \{0, 1\}^n} [\mathcal{A}(T(x)) \in T^{-1}(T(x))] = \text{neg}(n).$$

Moreover, if secure against subexponential adversaries, the above holds for any algorithm \mathcal{A} with running time at most 2^{n^δ} , for some constant δ .

We sketch the proof here.

Proof. Assume that algorithm \mathcal{A} can invert T with non-negligible probability. We claim that \mathcal{A} can be used to find a collision with non-negligible probability. Indeed, let $X \leftarrow \{0, 1\}^n$ be a uniformly distributed random variable. Let \mathcal{A}' be the algorithm that given random input X , executes $\mathcal{A}(T(X))$ and outputs its output.

Given that $\mathcal{A}(T(X))$ found a pre-image x' of $T(X)$, we get that the input of \mathcal{A}' , X , uniformly distributed over the set $T^{-1}(T(x'))$. Since the size of $T^{-1}(T(x'))$ is large (the probability that $|T^{-1}(T(x'))| \leq k$ is at most $k \cdot 2^{-s(n)}$), with high probability it holds that $x \neq X$, and thus \mathcal{A}' found a collision. ◁

3 GapMCSP is not NP-complete under Levin Reductions

In this section we prove our main result for GapMCSP. We first define $\text{GapMCSP}[s_0, s_1]$. In the following, a circuit C computes a string x if the truth table of C is x .

► **Definition 18.** For two functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$, let $\text{GapMCSP}[s_0, s_1]$ denote the following promise problem.

- $\mathcal{Y} = \{x \in \{0, 1\}^n: \text{There exists a circuit } C \text{ of size at most } s_0(n) \text{ that computes } x\}$
- $\mathcal{N} = \{x \in \{0, 1\}^n: \text{There is no circuit of size } s_1(n) \text{ that computes } x\}$

We define the relations $\mathcal{R}_{\mathcal{Y}}$ and $\mathcal{R}_{\mathcal{N}}$ for $\text{GapMCSP}[s_0, s_1]$ in the natural way:

$$\mathcal{R}_{\mathcal{Y}} = \{(x, C): C \text{ is a circuit of size at most } s_0(n) \text{ that computes } x\},$$

and,

$$\mathcal{R}_{\mathcal{N}} = \{(x, C): C \text{ is a circuit of size at most } s_1(n) \text{ that computes } x\}.$$

We start with the following theorem for deterministic reductions. In Section 3.2 we prove a similar theorem for randomized Levin reductions.

► **Theorem 19.** Let $p: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be a function. Assume that there exists iO with overhead p , and subexponentially-secure one-way function. Then for any constant $\alpha > 0$ and for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n), (s_0(n))^\alpha)$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not NP complete with respect to Levin reductions.

Since iO is an efficient algorithm, the overhead of any iO is polynomial. Combining this observation with Theorem 19 yields Theorem 1.

3.1 Proving Theorem 19

To prove Theorem 19, let iO be an indistinguishability obfuscator, and let $p \in \text{poly}$ be the overhead of iO . Let $T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-\omega(\log n)}$ be a TCR with security against subexponential algorithms.

Consider the following distribution ensemble $\mathcal{D} = \{\mathcal{D}_n\}_{n \in \mathbb{N}}$ over SAT formulas and assignments (ϕ, v) . For every $n \in \mathbb{N}$, to sample from \mathcal{D}_n : sample a random $x \in \{0, 1\}^n$. Let $\phi_{T(x)}$ be a formula such that $\phi_{T(x)}[x'] = 1$ if and only if $T(x') = T(x)$. Output $(\phi_{T(x)}, x)$. We remark that $\phi_{T(x)}$ only depends on the value of $T(x)$ and not on x itself.

We start with the following claim.

▷ **Claim 20.** The following hold for every $n \in \mathbb{N}$:

- $\Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\phi[v] = 1] = 1$
- $\Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\exists v' \text{ s.t. } v \neq v' \text{ and } \phi[v'] = 1] = 1 - \text{neg}(n)$, and,
- for every PPT algorithm \mathcal{A}

$$\Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\mathcal{A}(\phi, v) = v'; v \neq v' \text{ and } \phi[v'] = 1] = \text{neg}(n).$$

Proof. The first and last items follow directly from the definition of the distribution \mathcal{D} and the definition of TCR. The second item holds since T is shrinking. ◁

We also prove the following claim, which states that for any reduction f from SAT to GapMCSP, the output of f on inputs samples from \mathcal{D}_n must have length polynomial in n . Here we need the subexponential security of T .

36:12 Gap MCSP Is Not (Levin) NP-Complete in Obfustopia

▷ **Claim 21.** Let (f, g, h) be a Levin reduction from SAT to $\text{GapMCSP}[s_0, s_1]$. Then there exists a constant $\delta > 0$ such that

$$\Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [s_0(|f(\phi)|) \geq n^\delta] \geq 1 - \text{neg}(n)$$

► **Remark 22.** Claim 21 is the only place in which we use the subexponential security assumption. We need it to make sure that (with high probability over \mathcal{D}) $|s_0(f(\phi))|$ is not too small. While we can require that $s_0(n) \geq n^\epsilon$ for some $\epsilon > 0$, the reduction f itself can return short outputs.

When the reduction f is honest (that is, $|f(x)| \geq |x|^\alpha$ for all inputs x and for some $\alpha > 0$), we can replace the assumption on exponentially-secure one-way function with the above requirement that $s_0(n) \geq n^\epsilon$, and minimal assumption that $\mathbf{NP} \not\subseteq \mathbf{iOBPP}$. The latter assumption is known to imply (together with \mathbf{iO}) one-way function (see Theorem 16). Using the same proof as follows we get Theorem 2.

Proof. Assume toward a contradiction that this is not the case for all constant $\delta > 0$. We will show how to invert T . That is, we will show an algorithm \mathcal{A} that runs in time $2^{n^{c-\delta}}$ for some constant c such that

$$\Pr_{x \leftarrow \{0,1\}^n} [\mathcal{A}(T(x)) \in T^{-1}(T(x))] \geq \Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [s_0(|f(\phi)|) < n^\delta].$$

The claim will then follow by Claim 17, as by assumption $\Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [s_0(|f(\phi)|) < n^\delta]$ is non-negligible for all choices of $\delta > 0$ (and for infinitely many n 's).

Let \mathcal{A} be the algorithm that given $y = T(x)$, constructs the formula ϕ_y , and then uses brute force to find a minimal circuit C of size at most n^δ that computes $f(\phi_y)$. Lastly, if such C exists, \mathcal{A} outputs $h(\phi_y, C)$.

It is not hard to see that \mathcal{A} runs in time $2^{\text{poly}(n^\delta)}$. By the definition of Levin reductions, when $s_0(|f(\phi_{T(x)})|) < n^\delta$, \mathcal{A} always outputs x' such that $T(x') = T(x)$. Lastly, observe that the distribution of ϕ_y for $y = T(x)$ when $x \leftarrow \{0,1\}^n$, is exactly the distribution of ϕ when $(\phi, v) \leftarrow \mathcal{D}_n$. ◁

The next lemma shows it is possible to use \mathbf{iO} to find collisions in the TCR.

► **Lemma 23.** Let \mathbf{iO} be an indistinguishability obfuscator with overhead p , and let s_0 and s_1 as in Theorem 19. Assume that there exists a Levin reduction from SAT to $\text{GapMCSP}[s_0, s_1]$. Then there exists an efficient algorithm \mathcal{A} such that for every large enough $n \in \mathbb{N}$

$$\Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [\mathcal{A}(\phi, v) = v'; v \neq v' \text{ and } \phi[v'] = 1] > 1/4.$$

Proof. We start with the definition of \mathcal{A} . Let f, g, h be the Levin reduction between SAT to $\text{GapMCSP}[s_0, s_1]$. Define $\mathcal{A}(\phi, v) = h(\phi, \mathbf{iO}(1^{|g(\phi, v)|^\alpha}, g(\phi, v)))$. In the following we omit the security parameter $1^{|g(\phi, v)|^\alpha}$ from the notation.

Next, we show that $\mathcal{A}(\phi, v)$ returns $v' \neq v$ that satisfies ϕ with probability at least $1/4$. By Claim 20, such v' exists with all but negligible probability over a random sample $(\phi, v) \leftarrow \mathcal{D}_n$. For the constant $\delta > 0$ from Claim 21 let \mathcal{G} be the set of all (ϕ, v) such that $s_0(|f(\phi)|) \geq n^\delta$ and that exists $v' \neq v$ with $\phi[v'] = 1$. By Claim 21, $\Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [(\phi, v) \in \mathcal{G}] \geq 1 - \text{neg}(n)$. In the following, fix $n \in \mathbb{N}$, and fix $(\phi, v) \in \mathcal{G}$, and $v' \neq v$ with $\phi[v'] = 1$.

By the correctness of f and g , $g(\phi, v)$ and $g(\phi, v')$ are two circuits with size at most $s_0(|f(\phi)|)$ with the same truth table $f(\phi)$. We assume without loss of generality that $|g(\phi, v)| = |g(\phi, v')| = s_0(|f(\phi)|)$. By the assumption on the overhead time of the obfuscator \mathbf{iO} , we get that the size of the output of $\mathbf{iO}(g(\phi, v))$ and $\mathbf{iO}(g(\phi, v'))$ is at most

$$p(|g(\phi, v)|, |g(\phi, v')|^\alpha) = p(s_0(|f(\phi)|), (s_0(|f(\phi)|))^\alpha) < s_1(|f(\phi)|).$$

Thus, the output $iO(g(\phi, v))$ is a witness that $f(\phi)$ is not a No instance of $\text{GapMCSP}[s_0, s_1]$, and by the definition of h , $h(\phi, iO(g(\phi, v)))$ returns a witness that $\phi \in \text{SAT}$. Similarly, the same holds for v' : $h(\phi, iO(g(\phi, v')))$ returns a witness that $\phi \in \text{SAT}$.

Lastly, we use the security of iO to claim that $h(\phi, iO(g(\phi, v))) \neq v$ with a good probability. By the security of the obfuscator, and since $g(\phi, v)$ and $g(\phi, v')$ compute the same function $f(\phi)$ the output distributions of $iO(g(\phi, v))$ and $iO(g(\phi, v'))$ are indistinguishable. Moreover, since the iO is secure against non-uniform algorithms, the above distributions are indistinguishable also given (ϕ, v, v') (importantly, the size of (ϕ, v, v') is polynomial in the security parameter and in the size of the circuit $g(\phi, v)$ when $s_0(|f(x)|) \geq n^\delta$). In particular, by data processing, the distributions $h(\phi, iO(g(\phi, v)))$ and $h(\phi, iO(g(\phi, v')))$ must be indistinguishable.

By the definition of \mathcal{A} , we get that

$$\Pr[\mathcal{A}(\phi, v) = v] \leq \Pr[\mathcal{A}(\phi, v') = v] + \mu(s_0(|f(\phi)|))$$

for some negligible function μ . Since $(\phi, v) \in \mathcal{G}$, for every large enough n we get that

$$\Pr[\mathcal{A}(\phi, v) = v] \leq \Pr[\mathcal{A}(\phi, v') = v] + \mu(s_0(|f(\phi)|)) \leq \Pr[\mathcal{A}(\phi, v') \neq v'] + 1/3,$$

which implies that

$$1 - \Pr[\mathcal{A}(\phi, v) \neq v] \leq \Pr[\mathcal{A}(\phi, v') \neq v'] + 1/3,$$

or that

$$1/2 \cdot (\Pr[\mathcal{A}(\phi, v) \neq v] + \Pr[\mathcal{A}(\phi, v') \neq v']) \geq 1/3. \quad (1)$$

To finish the proof, consider the distribution \mathcal{D}'_n , in which we sample $(\phi, v) \leftarrow \mathcal{D}_n$, and then if $(\phi, v) \in \mathcal{G}$, we sample a random $v' \neq v$ such that $\phi[v'] = 1$ (or let $v' = v$ if $(\phi, v) \notin \mathcal{G}$). We then output (ϕ, v, v') .

We get that

$$\begin{aligned} & \Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\mathcal{A}(\phi, v) \neq v] \\ & \geq \Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\mathcal{A}(\phi, v) \neq v \mid (\phi, v) \in \mathcal{G}] \cdot \Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[(\phi, v) \in \mathcal{G}] \\ & = \Pr_{(\phi, v) \leftarrow \mathcal{D}_n}[\mathcal{A}(\phi, v) \neq v \mid (\phi, v) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\ & = \Pr_{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n}[\mathcal{A}(\phi, v_0) \neq v_0 \mid (\phi, v_0) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\ & = \Pr_{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n, b \leftarrow \{0,1\}}[\mathcal{A}(\phi, v_b) \neq v_b \mid (\phi, v_b) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\ & = 1/2 \cdot \sum_{b \in \{0,1\}} \Pr_{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n}[\mathcal{A}(\phi, v_b) \neq v_b \mid (\phi, v_b) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\ & \geq 1/3 - \text{neg}(n). \end{aligned}$$

where the third equality holds since the distribution of (ϕ, v_0) and (ϕ, v_1) are identical for $(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n$, and the last inequality by Equation (1). \blacktriangleleft

We are now ready to prove Theorem 19.

Proof of Theorem 19. Assume that iO and subexponential one-way functions exist. By Theorem 15, there exists a TCR with security against subexponential adversaries.

Assume there exists Levin reduction from SAT to $\text{GapMCSP}[s_0, s_1]$, and let \mathcal{D} be the distribution defined above. By Claim 20, there is no efficient algorithm that given a random sample (ϕ, v) from \mathcal{D}_n finds $v' \neq v$ such that $\phi[v'] = 1$ with non-negligible probability. But by Lemma 23, there exists such an algorithm that succeeds with probability $1/4$, which is a contradiction. \blacktriangleleft

3.2 Randomized Levin Reductions

In this part we generalize Theorem 19 to hold with respect to randomized reductions. We prove the following theorem.

► **Theorem 24.** *Let $0 \leq \epsilon \leq 1/30$ be a constant, and let $p: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be a function. Assume that there exist iO with overhead p , and subexponentially-secure one-way function. Then for any constant $\alpha > 0$ and for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n), (s_0(n))^\alpha)$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not **NP** complete with respect to randomized Levin reductions with ϵ -error.*

Theorem 1 (for randomized reductions) directly follows by Theorem 24 and the observation that the overhead p is always bounded by polynomial. The proof of Theorem 24 is similar to the proof of Theorem 19. Let iO be an indistinguishability obfuscator with overhead p , and $T: \{0, 1\}^n \rightarrow \{0, 1\}^{n-\omega(\log n)}$ be a TCR secure against subexponential adversaries. Let $\mathcal{D} = \{\mathcal{D}_n\}_{n \in \mathbb{N}}$ be the same distribution as defined in the proof of Theorem 19.

The following claim is the analog of Claim 21 for randomized reductions.

▷ **Claim 25.** Let (f, g, h) be a randomized Levin reduction with ϵ -error from SAT to $\text{GapMCSP}[s_0, s_1]$. Then there exists a constant $\delta > 0$ such that

$$\Pr_{(\phi, v) \leftarrow \mathcal{D}_n, r_1 \leftarrow \{0, 1\}^*} [s_0(|f(\phi; r_1)|) \geq n^\delta] \geq 1 - 2\epsilon - \text{neg}(n)$$

Proof. The proof follows the same lines as the proof of Claim 21. Specifically, let $\delta > 0$, \mathcal{A} be the algorithm described in the proof of Claim 21. We will show that

$$\Pr_{x \leftarrow \{0, 1\}^n} [\mathcal{A}(T(x)) \in T^{-1}(T(x))] \geq \Pr_{(\phi, v) \leftarrow \mathcal{D}_n, r_1 \leftarrow \{0, 1\}^*} [s_0(|f(\phi)|) < n^\delta] - 2\epsilon.$$

The claim will then follow by Claim 17.

By the definition of randomized Levin reductions, with probability at least $1 - \epsilon$ over the choice of r_1 , it holds that h succeed to convert a witness for $f(\phi; r_1)$ to a witness for ϕ with probability at least $1 - \epsilon$. By the union bound, with probability at least

$$1 - \Pr_{(\phi, v) \leftarrow \mathcal{D}_n, r_1 \leftarrow \{0, 1\}^*} [s_0(|f(\phi; r_1)|) < n^\delta] - \epsilon$$

over the choice of $(\phi, v) \leftarrow \mathcal{D}_n$ and r_1 , it holds that both $s_0(|f(\phi; r_1)|) < n^\delta$, and h converts witnesses for $f(\phi; r_1)$ to witnesses for ϕ with probability at least $1 - \epsilon$. In this case, \mathcal{A} finds a witness for $f(\phi; r_1)$ and outputs a pre-image of T with probability $1 - \epsilon$.

Using the union bound again, we get that \mathcal{A} finds such a pre-image with probability at least

$$1 - \Pr_{(\phi, v) \leftarrow \mathcal{D}_n, r_1 \leftarrow \{0, 1\}^*} [s_0(|f(\phi; r_1)|) \geq n^\delta] - 2\epsilon$$

as claimed. ◁

The next lemma generalized Lemma 23, to shows it is possible to use iO and randomized Levin reduction to find collisions in the TCR.

► **Lemma 26.** *Let iO be indistinguishability obfuscator with overhead p , and let ϵ, s_0 and s_1 as in Theorem 24. Assume that there exists a randomized Levin reduction with ϵ -error from SAT to $\text{GapMCSP}[s_0, s_1]$. Then there exists an efficient algorithm \mathcal{A} such that for every large enough $n \in \mathbb{N}$*

$$\Pr_{(\phi, v) \leftarrow \mathcal{D}_n} [\mathcal{A}(\phi, v) = v'; v \neq v' \text{ and } \phi[v'] = 1] > 1/4 - 7\epsilon.$$

Proof. We start with the definition of \mathcal{A} . Let f, g, h be the Levin reduction between SAT to GapMCSP $[s_0, s_1]$, and define \mathcal{A} to be the algorithm that on input ϕ, v , outputs

$$h(\phi, iO(1^{|g(\phi, v; r_1)|^\alpha}, g(\phi, v; r_1)); r_1, r_2),$$

for a random choice of randomness r_1, r_2 for g, h . In the following we omit the security parameter $1^{|g(\phi, v; r_1)|^\alpha}$ from the notation.

Next, we show that $\mathcal{A}(\phi, v)$ returns $v' \neq v$ that satisfies ϕ with probability at least $1/4$. Let \mathcal{G} be the set of all SAT formulas ϕ such that there are $v \neq v'$ such that $\phi[v] = \phi[v'] = 1$.

Let $\delta > 0$ be the constant from Claim 25. In the following, we say that a randomness r_1 is *good* for a formula ϕ and a satisfying assignments v , if it holds that (1) $s_0(|f(\phi; r_1)|) \geq n^\delta$, (2) $g(\phi, v; r_1)$ is a circuit of size at most $s_0(|f(\phi; r_1)|)$ that computes $f(\phi; r_1)$, and (3), for any circuit C of size less than $s_1(|f(\phi; r_1)|)$ which computes $f(\phi; r_1)$, it holds that $h(\phi, C; r_1, r_2)$ is a satisfying assignment for ϕ with probability at least $1 - \epsilon$ over the choice of r_2 . That is, r_1 is good if the output of $f(\phi; r_1)$ is not too short, and if the reduction succeed in converting witnesses from SAT to GapMCSP using the randomness r_1 .

By the definition of Levin reductions with ϵ -error a random r_1 fulfils the last two requirements with probability at least $1 - \epsilon$. Using Claim 25 and the union bound, we get that a random r_1 is good for (ϕ, v) with probability at least $1 - 3\epsilon - \text{neg}(n)$.

For $\phi \in \mathcal{G}$, and two satisfying assignments $v \neq v'$, let $\mathcal{R}_{\phi, v, v'}$ be the set of all random strings r_1 such that r_1 is good both for (ϕ, v) and for (ϕ, v') . Using the union bound again, we get that

$$\Pr_{r_1 \leftarrow \{0,1\}^*} [r_1 \in \mathcal{R}_{\phi, v, v'}] \geq 1 - 6\epsilon - \text{neg}(n). \quad (2)$$

We continue as in the proof of Lemma 23. In the following, fix $\phi \in \mathcal{G}$ and two satisfying assignments $v \neq v'$, and fix $r_1 \in \mathcal{R}_{\phi, v, v'}$.

By the definition of $\mathcal{R}_{\phi, v, v'}$, $g(\phi, v; r_1)$ and $g(\phi, v'; r_1)$ are two circuits with size at most $s_0(f(\phi))$ with the same truth table $f(\phi; r_1)$. We assume without loss of generality that $|g(\phi, v)| = |g(\phi, v')| = s_0(|f(\phi)|)$. As in the proof of Lemma 23, by the assumption on the overhead of the obfuscator iO , we get that the size of the output of $iO(g(\phi, v; r_1))$ and $iO(g(\phi, v'; r_1))$ is less than $s_1(|f(\phi; r_1)|)$. Thus, the output $iO(g(\phi, v; r_1))$ is a witness that $f(\phi; r_1)$ is not a No instance of GapMCSP $[s_0, s_1]$, and by the definition of h and $\mathcal{R}_{\phi, v, v'}$, $h(\phi, iO(g(\phi, v; r_1, r_2)))$ returns a witness that $\phi \in \text{SAT}$ with probability at least $1 - \epsilon$ over the choice of r_2 . Similarly, the same holds for v' : $h(\phi, iO(g(\phi, v'; r_1, r_2)))$ returns a witness that $\phi \in \text{SAT}$ with the same probability.

Lastly, we use the security of iO to claim that $h(\phi, iO(g(\phi, v; r_1); r_1, r_2))$ outputs an satisfying assignment to ϕ which is not equal to v with a good probability. By the security of the obfuscator, and since $g(\phi, v; r_1)$ and $g(\phi, v'; r_1)$ computes the same function $f(\phi; r_1)$ the output distributions of $iO(g(\phi, v; r_1))$ and $iO(g(\phi, v'; r_1))$ are indistinguishable. Moreover, by the non-uniform security, the above distributions are indistinguishable also given (x, v, v', r_1) . In particular, by data processing, the distributions $h(\phi, iO(g(x, v; r_1)); r_1, r_2)$ and $h(\phi, iO(g(x, v'; r_1)); r_1, r_2)$ must be indistinguishable. Let $\mathcal{A}(\phi, v; r_1)$ be the output of $\mathcal{A}(\phi, v)$ when we fix the randomness \mathcal{A} uses for f to be r_1 . In the following we assume without loss of generality that whenever \mathcal{A} do not output a satisfying assignment for ϕ , it outputs \perp . By the definition of \mathcal{A} , when $r_1 \in \mathcal{R}_{\phi, v, v'}$ we get that

$$\Pr[\mathcal{A}(\phi, v; r_1) = v] \leq \Pr[\mathcal{A}(\phi, v'; r_1) = v] + \mu(s_0(|f(\phi)|))$$

for some negligible function μ . As in the proof of Lemma 23, this implies that

$$1/2 \cdot (\Pr[\mathcal{A}(\phi, v; r_1) \neq v] + \Pr[\mathcal{A}(\phi, v'; r_1) \neq v']) \geq 1/3. \quad (3)$$

36:16 Gap MCSP Is Not (Levin) NP-Complete in Obfustopia

Since h fails with probability at most ϵ , we get that

$$1/2 \cdot (\Pr[\mathcal{A}(\phi, v; r_1) \notin \{v, \perp\}] + \Pr[\mathcal{A}(\phi, v'; r_1) \notin \{v', \perp\}]) \geq 1/3 - \epsilon. \quad (4)$$

To finish the proof, consider the distribution \mathcal{D}'_n , in which we sample $(\phi, v) \leftarrow \mathcal{D}_n$, and then if $\phi \in \mathcal{G}$, we sample a random $v' \neq v$ such that $\phi[v'] = 1$ (otherwise we let $v' = v$). We then output (ϕ, v, v') .

We get that

$$\begin{aligned} & \Pr_{(\phi, v) \leftarrow \mathcal{D}_n, r_1 \leftarrow \{0,1\}^*} [\mathcal{A}(\phi, v; r_1) \notin \{v, \perp\}] \\ &= \Pr_{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n, r_1 \leftarrow \{0,1\}^*} [\mathcal{A}(\phi, v_0; r_1) \notin \{v_0, \perp\}] \\ &\geq \Pr_{\substack{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n \\ r_1 \leftarrow \{0,1\}^*}} [\mathcal{A}(\phi, v_0; r_1) \notin \{v_0, \perp\} \mid \phi \in \mathcal{G}, r_1 \in \mathcal{R}_{\phi, v_0, v_1}] \\ &\quad \cdot \Pr[r_1 \in \mathcal{R}_{\phi, v_0, v_1} \mid \phi \in \mathcal{G}] \cdot \Pr[\phi \in \mathcal{G}] \\ &\geq \Pr_{\substack{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n \\ r_1 \leftarrow \{0,1\}^*}} [\mathcal{A}(\phi, v_0; r_1) \notin \{v_0, \perp\} \mid \phi \in \mathcal{G}, r_1 \in \mathcal{R}_{\phi, v_0, v_1}] \\ &\quad \cdot (1 - 6\epsilon - \text{neg}(n))(1 - \text{neg}(n)) \\ &\geq \Pr_{\substack{(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n \\ r_1 \leftarrow \{0,1\}^* \\ b \leftarrow \{0,1\}}} [\mathcal{A}(\phi, v_b; r_1) \notin \{v_b, \perp\} \mid \phi \in \mathcal{G}, r_1 \in \mathcal{R}_{\phi, v_0, v_1}] \\ &\quad \cdot (1 - 6\epsilon - \text{neg}(n))(1 - \text{neg}(n)) \\ &\geq (1/3 - \epsilon) \cdot (1 - 6\epsilon - \text{neg}(n))(1 - \text{neg}(n)) \\ &\geq 1/4 - 7\epsilon. \end{aligned}$$

where the second inequality holds by Equation (4) and by Claim 25, the third equality holds since the distribution of (ϕ, v_0) and (ϕ, v_1) are identical for $(\phi, v_0, v_1) \leftarrow \mathcal{D}'_n$, in by a similar argument as in the proof of Lemma 23, and the last inequality holds for large enough n and for a small enough constant ϵ . \blacktriangleleft

We are now ready to prove Theorem 19.

Proof of Theorem 19. Assume that iO and subexponentially-secure one-way function exist. By Theorem 15, there exists a TCR with security against subexponential adversaries.

Assume there exists Levin reduction from SAT to $\text{GapMCSP}[s_0, s_1]$, and let \mathcal{D} be the distribution defined above. By Claim 20, there is no efficient algorithm that given a random sample (ϕ, v) from \mathcal{D}_n finds $v' \neq v$ such that $\phi[v'] = 1$ with non-negligible probability. But by Lemma 26, there exists such an algorithm that succeeds with probability $1/4 - 7\epsilon$, which is a contradiction when $\epsilon < 1/28$. \blacktriangleleft

4 $\text{Gap}_p\text{MK}^t\text{P}$ is not NP-complete under Levin Reductions

In this section we prove our result for MK^tP . That is, we prove that (under cryptographic assumptions) there is no Levin reduction from SAT to the following promise problem. For $p, t \in \text{poly}$, let $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ be the following promise problem:

- $\mathcal{Y} = \{x \in \{0, 1\}^n : K^{t(n)}(x) \leq s_0(n)\}$
- $\mathcal{N} = \{x \in \{0, 1\}^n : K^{p(t(n))}(x) > s_1(n)\}$

We define the relations \mathcal{R}_Y and \mathcal{R}_N for $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ in the natural way:

$$\mathcal{R}_Y = \left\{ (x, P) : P \text{ is a program of length at most } s_0(n) \text{ such that } U(P, 1^{t(|x|)}) = x \right\},$$

and,

$$\mathcal{R}_N = \left\{ (x, P) : P \text{ is a program of length at most } s_1(n) \text{ such that } U(P, 1^{p(t(|x|))}) = x \right\}.$$

The proof follows the same line as the proof of Theorem 19, where we replace the iO with randomized encoding for Turing machines with indistinguishability-based security [7].

► **Definition 27** (Randomized encoding for TM). *A pair of efficient randomized algorithms (Enc, Dec) is randomized encoding for TMs if the following holds: Let M be a TM and $x \in \{0, 1\}^*$ be an input, $\lambda \in \mathbb{N}$ be a security parameter and let $T \in \mathbb{N}$ be a bound on the running time of $M(x)$. Then*

1. (Correctness:) $\Pr[\text{Dec}(\text{Enc}(1^\lambda, M, x, T)) = M(x)] = 1$
2. (Efficiency:) $\text{Enc}(1^\lambda, M, x, T)$ runs in time $\text{poly}(\lambda, |M|, |x|, \log T)$ and $\text{Dec}(\widehat{M(x)})$ runs in time $\text{poly}(\lambda, |M|, |x|, t)$ for $\widehat{M(x)} \leftarrow \text{Enc}(1^\lambda, M, x, T)$ and where $t \leq T$ is the running time of $M(x)$, and,
3. (Security:) For every ppt algorithm \mathcal{A} and every $s \in \text{poly}$ there exists a negligible function μ , such that for every TM M and two inputs x_0, x_1 such that $M(x_0) = M(x_1)$, $|M| \leq s(\lambda)$, $|x_0| \leq s(\lambda)$, $|x_1| \leq s(\lambda)$ and the running time of M on x_0 at most $s(\lambda)$ and is the same as the running time of M on x_1 , the following holds:

$$\left| \Pr[\mathcal{A}(\text{Enc}(1^\lambda, M, x_0, T)) = 1] - \Pr[\mathcal{A}(\text{Enc}(1^\lambda, M, x_1, T)) = 1] \right| = \mu(\lambda).$$

We say that (Enc, Dec) has overhead p if $|\text{Enc}(1^\lambda, M, x, T)| \leq p(|M|, |x|, T, \lambda)$ with probability 1.

Using randomized encoding, we get the following theorem.

► **Theorem 28.** *Let $0 \leq \epsilon \leq 1/30$ be a constant. Assume that randomized encoding for TMs with overhead q , and subexponentially-secure one-way function exists. Then there exists a constant $c \in \mathbb{N}$ such that for every constant $\alpha > 0$, for any $t \in \text{poly}$ and any efficiently computable functions $s_0, s_1 : \mathbb{N} \rightarrow \mathbb{N}$ for which*

$$s_1(n) > q(c, s_0(n) + c \log(t(n)) + c \log(s_0(n)), \log t(n), (s_0(n))^\alpha),$$

and for every large enough polynomial p , it holds that $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ is not **NP** complete with respect to randomized Levin reductions with ϵ -error.

By the results of [48, 43] such randomized encoding with polynomial overhead q for poly-time TMs can be constructed assuming one-way functions, subexponentially-secure iO for circuits and injective PRG (that can be constructed from one-way permutation). Together with Theorem 28 we get Theorem 3. As in Theorem 19, we can relax the requirement for subexponentially-secure one-way function if we only want to exclude honest reductions.

[8] constructed iO for TM with multiplicative overhead. By combining the construction of randomized encoding for TMs of [48] with the iO of [8], we get randomized encoding with multiplicative overhead.

► **Theorem 29.** *Assuming subexponentially-secure iO and subexponentially secure rerandomizable encryption schemes, there exists a randomized encoding for TMs scheme with overhead $q(|M|, |x|, T, \lambda) = 2(|M| + |x|) + \text{poly}(\lambda, \log T)$.*

We get the following corollary.

► **Corollary 30.** *Let $0 \leq \epsilon \leq 1/30$ be a constant. Assume subexponential-secure iO , and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every constant $\alpha > 0$, and for any efficiently computable function s_0 , it holds that $\text{Gap}_p\text{MK}^t\text{P}[s_0(n), (2 + \alpha)s_0(n)]$ is not **NP** complete with respect to randomized Levin reductions with ϵ -error.*

Proof of Theorem 28. For ease of notation, we explain how to modify the proof of Theorem 19 to get the proof of Theorem 28 for deterministic reductions. Similar changes to the proof of Theorem 24 yield the result for randomized reductions.

We only need to change the proof of Lemma 23. Let (f, g, h) be the Levin reduction from SAT to $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$, and assume that for every (ϕ, v) in the support of \mathcal{D} , $g(\phi, v)$ output a program of length exactly $s_0(|f(\phi)|)$ that runs in time exactly $t(|f(\phi)|)$ (this can be assume by adding $O(\log t(n) + \log s_0(n))$ bits to the description of $g(\phi, v)$). Let \mathbf{U} be a universal TM and (Enc, Dec) be randomized encoding for TMs. Consider the algorithm

$$\mathcal{A}(\phi, v) = h(\phi, \widehat{g(\phi, v)}),$$

where $\widehat{g(\phi, v)}$ is a program that runs Dec on \widehat{P} for $\widehat{P} \leftarrow Enc(1^{|g(\phi, v)|^\alpha}, \mathbf{U}, g(\phi, v), t(|f(\phi)|))$. That is, we replace the iO in the construction of \mathcal{A} from the proof of Lemma 23, with a randomized encoding of $\mathbf{U}(g(\phi, v))$. Since for every two witnesses v, v' of ϕ it holds that $\mathbf{U}(g(\phi, v)) = \mathbf{U}(g(\phi, v')) = f(\phi)$, we get that $\widehat{g(\phi, v)}$ and $\widehat{g(\phi, v')}$ are indistinguishable.

By the overhead of the randomized encoding scheme,

$$\left| \widehat{g(\phi, v')} \right| \leq q(|\mathbf{U}|, s_0(n) + O(\log(t(n)) + \log(s_0(n))), \log t(n), |g(\phi, v)|^\alpha).$$

By the efficiency of Dec , the running time of $\widehat{g(\phi, v')}$ is at most $\text{poly}(s_0(|f(\phi)|), t(|f(\phi)|)) = \text{poly}(t(|f(\phi)|))$, where the equality holds since $s_0(|f(\phi)|) \leq |f(\phi)| + O(1)$ or the $\text{Gap}_p\text{MK}^t\text{P}[s_0, s_1]$ problem is trivial. Thus, by taking p be a polynomial that bound the running time of $\widehat{g(\phi, v')}$, we get that $\widehat{g(\phi, v')}$ is a witness that $f(\phi)$ is not a No instance. The proof continues along the same lines as the proof of Lemma 23. ◀

References

- 1 Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: new methods for bootstrapping and instantiation. In *Advances in Cryptology—EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38*, pages 191–225. Springer, 2019.
- 2 Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear fe. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 110–140. Springer, 2020.
- 3 Eric Allender and Shuichi Hirahara. New insights on the (non-) hardness of circuit minimization and related problems. *ACM Transactions on Computation Theory (ToCT)*, 11(4):1–27, 2019.
- 4 Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77(1):14–40, 2011.
- 5 Prabhanjan Ananth, Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. Indistinguishability obfuscation without multilinear maps: new paradigms via low degree weak pseudorandomness and security amplification. In *Annual International Cryptology Conference*, pages 284–332. Springer, 2019.

- 6 Prabhanjan Ananth, Aayush Jain, and Amit Sahai. Indistinguishability obfuscation without multilinear maps: io from lwe, bilinear maps, and weak pseudorandomness. *Cryptology ePrint Archive*, 2018.
- 7 Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Annual Cryptology Conference*, pages 308–326. Springer, 2015.
- 8 Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Indistinguishability obfuscation for turing machines: constant overhead and amortization. In *Annual International Cryptology Conference*, pages 252–279. Springer, 2017.
- 9 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM (JACM)*, 45(3):501–555, 1998.
- 10 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. *Journal of the ACM (JACM)*, 45(1):70–122, 1998.
- 11 Boaz Barak and Oded Goldreich. Universal arguments and their applications. *SIAM Journal on Computing*, 38(5):1661–1694, 2009.
- 12 Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im) possibility of obfuscating programs. In *Annual international cryptology conference*, pages 1–18. Springer, 2001.
- 13 Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In *Theory of cryptography conference*, pages 52–73. Springer, 2014.
- 14 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and pairings are not necessary for io: Circular-secure lwe suffices. *Cryptology ePrint Archive*, 2020.
- 15 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from homomorphic encryption schemes. *Journal of Cryptology*, 36(3):27, 2023.
- 16 Gregory J. Chaitin. On the simplicity and speed of programs for computing infinite sets of natural numbers. *J. ACM*, 16(3):407–422, 1969.
- 17 Stephen A. Cook. The complexity of theorem-proving procedures. In *Annual ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 1971.
- 18 Irit Dinur, Venkatesan Guruswami, Subhash Khot, and Oded Regev. A new multilayered pcp and the hardness of hypergraph vertex cover. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 595–601, 2003.
- 19 Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 210–217, 1987.
- 20 Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.
- 21 Romain Gay, Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from simple-to-state hard problems: New assumptions, new techniques, and simplification. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 97–126. Springer, 2021.
- 22 Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 736–749, 2021.
- 23 Craig Gentry, Allison Bishop Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 151–170. IEEE, 2015.
- 24 Oded Goldreich. Computational complexity: A conceptual perspective, 2008.
- 25 Shafi Goldwasser and Guy N Rothblum. On best-possible obfuscation. *Journal of Cryptology*, 27(3):480–505, 2014.
- 26 J. Hartmanis. Generalized kolmogorov complexity and the structure of feasible computations. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, pages 439–445, 1983. doi:10.1109/SFCS.1983.21.

- 27 Shuichi Hirahara. NP-hardness of learning programs and partial mcsp. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 968–979. IEEE, 2022.
- 28 Shuichi Hirahara. Symmetry of information from meta-complexity. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
- 29 Yizhi Huang, Rahul Ilango, and Hanlin Ren. NP-hardness of approximating meta-complexity: A cryptographic approach. *Cryptology ePrint Archive*, 2023.
- 30 Rahul Ilango. Approaching MCSP from above and below: Hardness for a conditional variant and $AC^0[p]$. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
- 31 Rahul Ilango. SAT reduces to the minimum circuit size problem with a random oracle. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 733–742. IEEE, 2023.
- 32 Rahul Ilango, Bruno Loff, and Igor Carboni Oliveira. NP-hardness of circuit minimization for multi-output functions. In *CCC'20: Proceedings of the 35th Computational Complexity Conference*, pages 1–36, 2020.
- 33 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. *computational complexity*, 32(2):6, 2023.
- 34 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. Synergy between circuit obfuscation and circuit minimization. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.
- 35 Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. How to leverage hardness of constant-degree expanding polynomials over \mathbb{R} to build $i\mathcal{O}$. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38*, pages 251–281. Springer, 2019.
- 36 Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021.
- 37 Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 73–79, 2000.
- 38 Richard M. Karp. Reducibility among combinatorial problems. In J. W. Thatcher and R. E. Miller, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, Inc., 1972.
- 39 Ker-I Ko. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986. doi:10.1016/0304-3975(86)90081-2.
- 40 Ker-I Ko. On the complexity of learning minimum time-bounded turing machines. *SIAM Journal on Computing*, 20(5):962–986, 1991.
- 41 A. N. Kolmogorov. Three approaches to the quantitative definition of information. *International Journal of Computer Mathematics*, 2(1-4):157–168, 1968.
- 42 Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im) perfect obfuscation. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 374–383. IEEE, 2014.
- 43 Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*, pages 419–428, 2015.
- 44 Leonid A. Levin. Universal’nyĕ perebornyĕzadachi (Universal search problems : in Russian). *Problemy Peredachi Informatsii*, pages 265–266, 1973.
- 45 Leonid Anatolevich Levin. Universal sequential search problems. *Problemy peredachi informat-sii*, 9(3):115–116, 1973.

- 46 Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In *Advances in Cryptology – EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I 35*, pages 28–57. Springer, 2016.
- 47 Huijia Lin. Indistinguishability obfuscation from sxdh on 5-linear maps and locality-5 prgs. In *Annual International Cryptology Conference*, pages 599–629. Springer, 2017.
- 48 Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In *Theory of Cryptography Conference*, pages 96–124. Springer, 2015.
- 49 Huijia Lin and Stefano Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local prgs. In *Annual International Cryptology Conference*, pages 630–660. Springer, 2017.
- 50 Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 11–20. IEEE, 2016.
- 51 Yanyi Liu and Rafael Pass. On one-way functions from NP-complete problems. In *37th Computational Complexity Conference*, 2022.
- 52 Noam Mazor and Rafael Pass. Gap MCSP is not (levin) NP-complete in obfustopia. *Cryptology ePrint Archive*, 2024.
- 53 Cody D Murray and R Ryan Williams. On the (non) NP-hardness of computing circuit complexity. *Theory of Computing*, 13(1):1–22, 2017.
- 54 Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 33–43, 1989.
- 55 Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34*, pages 500–517. Springer, 2014.
- 56 Hanlin Ren and Rahul Santhanam. A relativization perspective on meta-complexity. In *39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
- 57 John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Annual ACM Symposium on Theory of Computing (STOC)*, pages 387–394, 1990.
- 58 Michael Saks and Rahul Santhanam. Circuit lower bounds from NP-hardness of MCSP under Turing reductions. *LIPICs*, 169, 2020.
- 59 Michael Saks and Rahul Santhanam. On randomized reductions to the random strings. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
- 60 Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 330–335, 1983.
- 61 R.J. Solomonoff. A formal theory of inductive inference. part i. *Information and Control*, 7(1):1–22, 1964. doi:10.1016/S0019-9958(64)90223-2.
- 62 Boris A Trakhtenbrot. A survey of russian approaches to perebor (brute-force searches) algorithms. *Annals of the History of Computing*, 6(4):384–400, 1984.
- 63 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.
- 64 Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious lwe sampling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 127–156. Springer, 2021.

