33rd EACSL Annual Conference on Computer Science Logic

CSL 2025, February 10–14, 2025, Amsterdam, Netherlands

Edited by

Jörg Endrullis

Sylvain Schmitz

LIPIcs – Vol. 326 – CSL 2025 www.dagstuhl.de/lipics

Editors

Jörg Endrullis
Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
j.endrullis@vu.nl

Sylvain Schmitz
Université Paris Cité, CNRS, IRIF, Paris, France
schmitz@irif.fr

ACM Classification 2012
Theory of computation → Logic

ISBN 978-3-95977-362-1

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/978-3-95977-362-1.

Publication date
February, 2025

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.

Digital Object Identifier: 10.4230/LIPIcs.CSL.2025.0

ISBN 978-3-95977-362-1 **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.

ISSN 1868-8969

https://www.dagstuhl.de/lipics

Contents

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).
Editors: Jörg Endrullis and Sylvain Schmitz
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Preface

This volume contains the papers presented at CSL 2025, the 33rd meeting in the conference series Computer Science Logic (CSL), the annual conference of the European Association for Computer Science Logic (EACSL). CSL 2025 was held from the 10th to 14th of February 2025 in Amsterdam, the Netherlands.

CSL started as a series of international workshops, and became an international conference in 1992. Previous instalments of CSL were held in Naples (2024), Warsaw (2023), Göttingen (2022, on-line), Ljubljana (2021, on-line), Barcelona (2020), Birmingham (2018), Stockholm (2017), Marseille (2016), Berlin (2015), Vienna (2014), Torino (2013), Fontainebleau (2012), Bergen(2011), Brno (2010), Coimbra (2009), Bologna (2008), Lausanne (2007), Szeged (2006), Oxford (2005), Karpacz (2004), Vienna (2003), Edinburgh (2002), Paris (2001), Munich (2000), Madrid (1999), Brno (1998), Aarhus (1997), Utrecht (1996), Paderborn (1995), Kazimierz (1994), Swansea (1993) and San Miniato (1992).

CSL is an interdisciplinary conference, spanning both basic and application-oriented research in mathematical logic and computer science. It is a forum for the presentation of research on all aspects of logic and its applications, including automated deduction and interactive theorem proving, constructive mathematics and type theory, equational logic and term rewriting, automata and games, game semantics, modal and temporal logic, logical aspects of computational complexity, finite model theory, computational proof theory, logic programming and constraints, lambda calculus and combinatory logic, domain theory, categorical logic and topological semantics, database theory, specification, extraction and transformation of programs, logical aspects of quantum computing, logical foundations of programming paradigms, verification and program analysis, linear logic, higher-order logic, and non-monotonic reasoning.

The conference received 130 abstracts, of which 113 were followed up by full-paper blind submissions, one of which was later retracted. The programme committee selected 44 papers for presentation at the conference. Each paper was overseen by at least three members of the programme committee, with the crucial help of 161 external reviewers who contributed 178 of the total 350 reviews. The submission and reviewing process, programme committee discussion, and author notifications were all handled through the Easychair conference management system.

In addition to the contributed papers, there were four invited talks, by: Patricia Bouyer-Decitre (CNRS, ENS Paris-Saclay, France), Yannick Forster (Inria Paris, France), Elaine Pimentel (University College London, UK), and Yde Venema (Universiteit van Amsterdam, the Netherlands). We thank the invited speakers for their stimulating talks and papers, which greatly contributed to the success of the conference.

One of the major regular events at CSL conferences is the presentation of the Ackermann Award: the annual EACSL award for an outstanding dissertation in the area of logic in computer science. The recipients of the award are selected by jury from a field of international nominees, and the recipients receive their award at a ceremony at which they give a prize lecture on their dissertation. This year, the jury elected to give the Ackermann Award 2024 jointly to Gaëtan Douéneau-Tabot for the PhD thesis entitled *Optimization of String Transducers*, completed at University Paris-Cité, France, in 2023, under the supervision of Olivier Carton and Emmanuel Filiot, and to Aliaume Lopez for the PhD thesis entitled *First Order Preservation Theorems in Finite Model Theory: Locality, Topology, and Limit*

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).
Editors: Jörg Endrullis and Sylvain Schmitz
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Constructions, completed at École Normale Supérieure Paris-Saclay, France, in 2023, under the supervision of Jean Goubault-Larrecq and Sylvain Schmitz. The award was presented during the conference. The citation for the award is included in the proceedings.

Another significant event at CSL 2025 was the presentation of the Helena Rasiowa Award, named after the eminent Polish mathematician and logician Helena Rasiowa (1917–1994) whose work had an essential impact on the emerging field of logic in computer science. The Helena Rasiowa Award, presented for the first time at CSL 2022, is given to the best paper, as decided by the programme committee, that is written solely by students or to which students were the main contributors. There was a strong field of candidates for this award edition, with 7 of the accepted papers reported as eligible by their authors. From these, the programme committee selected two recipients for the Helena Rasiowa Award: Ioannis Eleftheriadis for the paper entitled *Extension Preservation on Dense Graph Classes*, and Daumantas Kojelis for the paper entitled *On Homogeneous Models of Fluted Languages*. Ioannis Eleftheriadis is a PhD student at the Computer Laboratory of the University of Cambridge under the supervision of Anuj Dawar, and Daumantas Kojelis is a PhD student at the University of Manchester under the supervision of Ian Pratt-Hartmann.

CSL 2025 also had two affiliated workshops: the *Logic Mentoring Workshop* (LMW@CSL) and the *Workshop on Learning and Logic* (LeaLog@CSL).

We are very grateful to all the members of the CSL 2025 programme committee and external reviewers for their careful and efficient evaluation of the submitted papers. We would also like to thank the members of the organisation committee – Wan Fokkink and Emma Triesman – for taking care to ensure a smooth-running and enjoyable conference. It was, as always, a pleasure to work with Maribel Fernandez who, as the EACSL president, provided excellent guidance. The proceedings of CSL 2025 are published as a volume in the LIPIcs series. We thank Michael Wagner and all the Dagstuhl/LIPIcs team for their ongoing support and for the high quality preparation of these proceedings. Last, but not least, we are very grateful to the Theoretical Computer Science group at the Vrije Universiteit Amsterdam and ILLC at the University of Amsterdam for supporting the organisation of this conference.

Jörg Endrullis and Sylvain Schmitz November 29, 2024

Program Committee Members

Bahareh Afshari	(University of Gothenburg, Sweden)
Sandra Alves	(University of Porto, Portugal)
Camille Bourgaux	(CNRS, ENS Paris, France)
Laura Bozzelli	(Universita Napoli Federico II, Italy)
Paul Brunet	(Université Paris-Est Créteil, France)
Corina Cîrstea	(University of Southampton, UK)
Laure Daviaud	(City University London, UK)
Anuj Dawar	(University of Cambridge, UK)
Jörg Endrullis	(Vrije Universiteit Amsterdam, the Netherlands)
Natasha Fernandes	(Macquarie University, Sydney, Australia)
Dana Fisman	(Ben-Gurion University, Israel)
Moses Ganardi	(MPI-SWS Kaiserslautern, Germany)
Rob J. van Glabbeek	(UNSW, Sydney, Australia)
Julien Grange	(Université Paris-Est Créteil, France)
Robert Harper	(Carnegie Mellon University, USA)
Antti Kuusisto	(Tampere University, Finland)
Ugo Dal Lago	(University of Bologna, Italy)
Assia Mahboubi	(Inria Nantes, France)
Alessio Mansutti	(IMDEA Software Institute, Spain)
Dale Miller	(Inria Saclay, France)
Shankara Narayanan Krishna	(IIT Bombay, India)
Davide Sangiorgi	(University of Bologna, Italy)
Sylvain Schmitz	(Université Paris Cité, France)
Mahsa Shirmohammadi	(CNRS, IRIF, France)
Alwen Tiu	(Australian National University, Australia)
Takeshi Tsukada	(Chiba University, Japan)
Benoît Valiron	(CentraleSupélec, France)
Thomas Zeume	(Ruhr University Bochum, Germany)
Standa Živný	(University of Oxford, UK)

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).
Editors: Jörg Endrullis and Sylvain Schmitz

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

External Reviewers

Beniamino Accattoli
Matteo Acclavio
Antonis Achilleos
Robin Adams
Veeti Ahvonen
Shaull Almagor
Mario S. Alvim
Kazuyuki Asada
Steve Awodey
Arthur Azevedo de Amorim
Miriam Backens
Guillermo Badia
João Barbosa
Bartosz Bednarczyk
Massimo Benerecetti
Manuel Bodirsky
Alberto Bombardelli
Florian Bruse
Wojciech Buszkowski
Silvia Butti
Michaël Cadilhac
Silvio Capobianco
Marco Carmosino
Kostia Chardonnet
Vincent Cheval
Peter Cholak
Lorenzo Ciardo
Lorenzo Clemente
Gianluca Curzi
Tiziano Dalmonte
Luc Dartois
Maurizio De Nino
Dario Della Monica
Ivan Di Liberti
Sylvain Douteau
Andrej Dudenhefner
Thomas Ehrhard
Sebastian Enqvist
Marco Faella
Claudia Faggian
Chris Fermüller
Masood Feyzbakhsh Rankooh
Mário Florido
Jakub Gajarský
Zeinab Galal
Pierre Ganty
Han Gao
Francesco Gavazzo
Luca Geatti
Fatemeh Ghasemi
Nicola Gigante
Marianna Girlando
Silvio Gonnet
Rajeev Gore
Harrison Grodin
Giulio Guerrieri
Shibashis Guha
Ashutosh Gupta
Amar Hadzihasanovic
Masahito Hasegawa
Lauri Hella
Luisa Herrmann
Kengo Hirata
Nao Hirokawa
Piotr Hofman
Justin Hsu
Rosalie Iemhoff
Noa Izsak
Jean Christoph Jung
Benjamin Lucien Kaminski
Mamadou Moustapha Kanté
Shin-Ya Katsumata
Sayeh Khaniha
Emanuel Kieronski
Bartek Klin
Barbara König
Clemens Kupke
Dietrich Kuske
Stepan Kuznetsov
Ori Lahav
Alberto Larrauri
Karoliina Lehtinen
Ondrej Lengal
Marina Lenisa
Meven Lennon-Bertrand
Bert Lindenhovius
Kerkko Luosto
Kerkko Luosto
Marcin Łyczak
Ian Mackie
Florent Madelaine
Konstantinos Mamouras
Enrico Marchioni
Annabelle McIver
Brett McLean
Arne Meier
Samuel Mimram
Joshua Moerman
Fabio Mogavero
Sean Moss
Gopalan Nadathur
Tamio-Vesa Nakajima
Aleks Nanevski
Jakub Opršal
Magdalena Ortiz
Leonardo Pacheco
Anantha Padmanabha
Ludovic Patey
Adriano Peron
Frank Pfenning
Paolo Pistone
Andrew Pitts
Boldizsár Poór
Damien Pous
John Power
Cécilia Pradic
M. Praveen
Wojtek Przybyszewski
Loïc Pujet
Vineet Rajani
Miguel Ramos
Yann Ramusat
Colin Riba
Dino Rossegger
Jurriaan Rot
Sasha Rubin
Ken Sakayori
Pietro Sala
David Sanan
Alexis Saurin
Todd Schmid
Johannes Schmidt
Daniyar Shamkanov
Ian Shillito
Viorica Sofronie-Stokkermans

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).
Editors: Jörg Endrullis and Sylvain Schmitz
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Ana Sokolova
Dmitriy Traytel
Fabian Vehlken
Giannos Stamoulis
Felix Tschirbs
Miikka Vilander
Jonathan Sterling
Iris van der Giessen
Nils Vortmeier
Xin Sun
Vincent van Oostrom
James Worrell
Jacobo Torán
Daniele Varacca

The Ackermann Award 2024

Maribel Fernández
Department of Informatics, King's College London, UK

Prakash Panangaden
McGill University, Canada

Abstract

Report on the 2024 Ackermann Award.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Finite Model Theory; Theory of computation → Proof theory; Theory of computation → Transducers

Keywords and phrases finite automaton, string transducer, class membership problem, first-order logic, preservation theorem, finite model theory

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.1

Category Ackermann Award

Introduction

The Ackermann Award is the **EACSL Outstanding Dissertation Award for Logic in Computer Science**. It is presented at CSL, the annual conference of the EACSL (European Association for Computer Science Logic). This year the 20th Ackermann Award is presented at CSL 2025 in Amsterdam, The Netherlands.

A call for nominations was issued in February 2024, open to any PhD dissertation (on any topic represented at the annual CSL and LICS conferences) formally accepted by a degree-granting institution in fulfilment of the PhD degree between 1 January 2023 and 31 December 2023.

The Jury received ten nominations, which came from a number of different countries around the world: the nominees obtained their doctorates at institutions in Belgium, Canada, Czech Republic, France, Poland and the United Kingdom. The topics covered a wide range of areas in Logic and Computer Science.

This year we received a particularly strong set of nominations. All the nominated PhD theses contained significant contributions to their particular fields. On behalf of the Ackermann Jury, we extend our warmest congratulations to all the nominated candidates for their outstanding work.

All the submissions were evaluated by the Jury, and after two phases of reviewing and extensive discussion, the jury decided to grant the **2024 Ackermann Award** jointly to (in alphabetic order):

- **Gaëtan Douéneau-Tabot** for the PhD thesis entitled *Optimization of string transducers*, completed at *University Paris-Cité*, France, in 2023;
- **Aliaume Lopez** for the PhD thesis entitled *First Order Preservation Theorems in Finite Model Theory: Locality, Topology, and Limit Construction*, completed at *University Paris-Saclay*, France, in 2023.

Citation for Gaëtan Douéneau-Tabot

Gaëtan Douéneau-Tabot shares the *2024 Ackermann Award* of the European Association of Computer Science Logic for the PhD thesis

Optimization of string transducers,

which solves open class membership problems for various transducer models, providing effective membership procedures that give rise to program optimization techniques. To achieve this goal, the thesis develops an extensive toolbox for solving class membership problems for transducers, including new characterisations of transducer behaviours.

Background to the thesis

This thesis deals with transducer models, that is, finite automata extended with string outputs, which define functions from strings to strings. String transducers are finite-state devices that represent programs with bounded memory. They are used in language processing tools, compilers, streaming algorithms and model-checkers, amongst others.

Various transducer models have been defined as extensions of the basic model by adding features to increase their expressive power. A natural question then arises: what is the expressive power of each class of transducers? This can be rephrased as a class membership problem: given a transducer with complex features, is there a simpler transducer that has the same behaviour? Such a question can be interpreted as a program optimisation problem: given a program, can we build an equivalent program that requires less resources?

Class membership problems in this context are known to be challenging. In his PhD thesis, Douéneau-Tabot solves multiple instances of the problem, obtaining new results that characterise various classes of transducer models and help understand their relationships.

Contributions of the thesis

The thesis solves open class membership problems for various transducer models. In each case the membership procedure is effective, in the sense that it builds a more efficient transducer whenever one exists, thus providing program optimisation techniques in the setting of transducers. In particular, the techniques can be applied to automatically remove recursion or nested loops for specific classes of programs.

In addition, the thesis provides new computation models and new characterisations that capture pre-existing classes of transductions. These results provide new insights on the expressive power of different kinds of transducers.

The proof techniques introduced are also valuable as a generic toolbox for solving other class membership problems in transducer models, and more generally in automata theory.

Biographical sketch

Gaëtan Douéneau-Tabot carried out his PhD studies at University Paris-Cité, under the supervision of Olivier Carton and Emmanuel Filiot, from 2020 to 2023. During his PhD he (co)-authored papers published in the proceedings of conferences such as ICALP 2023, LICS 2023, FoSSaCS 2023, MFCS 2022, ICALP 2022, MFCS 2021, MFCS 2020. His MFCS 2022 and ICALP 2022 papers won “best student paper” awards.

Citation for Aliaume Lopez

Aliaume Lopez shares the *2024 Ackermann Award* of the European Association of Computer Science Logic for the PhD thesis

> *First Order Preservation Theorems in Finite Model Theory: Locality, Topology, and Limit Constructions*,

which presents a systematic approach to investigating preservation theorems in Finite Model Theory, providing tools to explain when and why some classes of structures are well-behaved with respect to preservation theorems. The approach provides a compositional theory for preservation theorems that was previously lacking.

Background to the thesis

Preservation theorems in first-order logic are a collection of results derived from classical model theory, which establish a direct correspondence between the semantic properties of formulas and the constraints imposed on their syntax.

These theorems have a practical impact in computer science, where they can be used for example to characterise syntactic classes of database queries for which the termination and correctness of database algorithms are guaranteed. Unfortunately preservation theorems are notably challenging when focusing on finite models – the models considered in computer science applications. Identifying well-behaved classes has been an active domain of research for the last sixty years, with a series of negative results as well as some positive ones, which motivated the quest for a systematic approach to the problem.

Contributions of the thesis

This thesis presents a systematic approach to investigating preservation theorems within the realm of Finite Model Theory. The traditional ad-hoc proofs are replaced with a theoretical framework that generalises techniques based on locality, and introduces a topological presentation of preservation theorems called logically presented pre-spectral spaces.

Introducing these topological spaces enables the development of a compositional theory for preservation theorems. Additionally, this thesis develops a methodology to systematically examining preservation theorems across inductively defined classes of finite structures, by proving a generic fixed point theorem for a topological restriction of logically presented pre-spectral spaces: Noetherian spaces, which are topological spaces in which every open set is compact.

Biographical sketch

Aliaume Lopez carried out his PhD studies under the supervision of Jean Goubault-Larrecq (École Normale Supérieure Paris-Saclay) and Sylvain Schmitz (University Paris-Cité). The thesis is built around three articles published at CSL 2021, LICS 2022 and FoSSaCS 2023 (he published other papers during his PhD unrelated with the thesis). He is the winner of the E.W. Beth Dissertation Prize 2024. Currently he is a postdoctoral researcher at the University of Warsaw.

Jury

The jury for the **Ackermann Award 2024** consisted of nine members, two of them *ex officio*, namely, the president and the vice-president of EACSL. In addition, the jury also included a representative of SIGLOG (the ACM Special Interest Group on Logic and Computation).

The members of the jury were:

- Albert Atserias (Technical University of Catalonia);
- Christel Baier (Technical University Dresden);
- Andrej Bauer (University of Ljubljana);
- Maribel Fernández (King's College London), president of EACSL;
- Joost-Pieter Katoen (RWTH Aachen University), ACM SIGLOG representative;
- Delia Kesner (IRIF, University Paris Cité);
- Slawomir Lasota (University of Warsaw);
- Florin Manea (University of Göttingen), vice-president of EACSL;
- Prakash Panangaden (McGill University);

Previous winners

Previous winners of the Ackermann Award were

2005, Oxford:
Mikołaj Bojańczyk from Poland,
Konstantin Korovin from Russia, and
Nathan Segerlind from the USA.

2006, Szeged:
Balder ten Cate from the Netherlands, and
Stefan Milius from Germany.

2007, Lausanne:
Dietmar Berwanger from Germany and Romania,
Stéphane Lengrand from France, and
Ting Zhang from the People's Republic of China.

2008, Bertinoro:
Krishnendu Chatterjee from India.

2009, Coimbra:
Jakob Nordström from Sweden.

2010, Brno:
no award given.

2011, Bergen:
Benjamin Rossman from USA.

2012, Fontainebleau:
Andrew Polonsky from Ukraine, and
Szymon Toruńczyk from Poland.

2013, Turin:
Matteo Mio from Italy.

2014, Vienna:
Michael Elberfeld from Germany.

2015, Berlin:
Hugo Férée from France, and
Mickael Randour from Belgium.

2016, Marseille:
Nicolai Kraus from Germany.
2017, Stockholm:
Amaury Pouly from France.
2018, Birmingham:
Amina Doumane from France.
2019, Barcelona (conference in 2020):
Antoine Mottet from France.
2020, Ljubljana (conference online in 2021)
Benjamin Kaminski from Germany.
2021, Göttingen (conference online in 2022)
Marie Fortin from France, and
Sandra Kiefer from Germany.
2022, Warsaw (conference in 2023)
Alexander Bentkamp from The Netherlands.
2023, Naples (conference in 2024)
Gabriele Vanoni from Italy.
Detailed reports on their work appeared in the CSL proceedings and are also available on the EACSL homepage.

On the Probabilistic and Statistical Verification of Infinite Markov Chains

Patricia Bouyer
Université Paris-Saclay, CNRS, ENS Paris-Saclay, Laboratoire Méthodes Formelles,
91190 Gif-sur-Yvette, France

Abstract

The verification of infinite-state Markov chains is a challenging problem, even when those chains are described by structured high-level models. In 2007, Abdulla *et al* introduced the concept of decisiveness [1], and showed that a natural approximation scheme could be applied to infinite Markov chains that are decisive. This was, up to our knowledge, the unique generic scheme that could be widely applied to (decisive) infinite Markov chains providing guarantees on the computed values (under some mild assumptions for effectiveness). On the other hand, statistical model-checking is a very efficient method that can be used for estimating probabilities in stochastic systems [8, 7]. We explain in this talk that decisiveness is also a key concept that allows to apply such statistical methods to infinite Markov chains.

While decisiveness is a crucial property, not all Markov chains are decisive, and it is therefore desirable to propose methods to analyze non-decisive Markov chains. Importance sampling [6] is a method which has been proposed to improve efficiency of statistical model-checking, in particular for estimating probabilities of rare events in stochastic systems. The idea is to biase the original chain, and to estimate the probabilities in the biased chain; guarantees can sometimes be given, as studied for instance in [5].

In this talk, we will explain how we use the importance sampling idea to turn a non-decisive Markov chain into a biased decisive Markov chain, in which we can estimate probabilities (with guarantees). We apply the general approach to a class of probabilistic pushdown automata. Our algorithms have been implemented in the tool Cosmos [2], and we discuss the methodology for experiments as well as our (partial) conclusions.

2012 ACM Subject Classification Mathematics of computing → Markov processes; Theory of computation → Concurrency

Keywords and phrases Markov Chains, Infinite state systems, Numerical and statistical Vverification

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.2

Category Invited Talk

Related Version A preliminary work on this topic appeared in [3], whose extended version is [4].
Full Version: `https://arxiv.org/abs/2409.18670` [4]

Funding This work has been partly supported by ANR projects MAVeriQ (ANR-20-CE25-0012) and BisoUS (ANR-22-CE48-0012).

Acknowledgements I am thankful to Benoît Barbot and Serge Haddad for introducing me to the world of statistical model-checking.

References

1 P. A. Abdulla, N. B. Henda, and R. Mayr. Decisive Markov chains. *Logical Methods in Computer Science*, 3(4), 2007. doi:10.2168/LMCS-3(4:7)2007.

2 P. Ballarini, B. Barbot, M. Duflot, S. Haddad, and N. Pekergin. HASL: A new approach for performance evaluation and model checking from concepts to experimentation. *Performance Evaluation*, 90:53–77, 2015. doi:10.1016/j.peva.2015.04.003.

3 B. Barbot, P. Bouyer, and S. Haddad. Beyond decisiveness of infinite Markov chains. In *Proc. 44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'24)*, volume 323 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:21, 2024.

4 B. Barbot, P. Bouyer, and S. Haddad. Beyond decisiveness of infinite Markov chains. *CoRR arXiV*, 2024. doi:10.48550/arXiv.2409.18670.

5 B. Barbot, S. Haddad, and C. Picaronny. Coupling and importance sampling for statistical model checking. In *18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'12)*, volume 7214 of *LNCS*, pages 331–346. Springer, 2012. doi:10.1007/978-3-642-28756-5_23.

6 H. Kahn and T. E. Harris. Estimation of particle transmission by random sampling. *National Bureau of Standards applied mathematics series*, 12:27–30, 1951.

7 H. L. S. Younes, E. M. Clarke, and P. Zuliani. Statistical verification of probabilistic properties with unbounded until. In *13th Brazilian Symposium on Formal Methods (SBMF'10)*, volume 6527 of *LNCS*, pages 144–160. Springer, 2010. doi:10.1007/978-3-642-19829-8_10.

8 H. L. S. Younes and R. G. Simmons. Statistical probabilistic model checking with a focus on time-bounded properties. *Information and Computation*, 204(9):1368–1409, 2006. doi:10.1016/J.IC.2006.05.002.

Synthetic Mathematics for the Mechanisation of Computability Theory and Logic

Yannick Forster
Inria Paris, France

Abstract

Mathematical practice in most areas of mathematics is based on the assumption that proofs could be made fully formal in a chosen foundation in principle. This assumption is backed by partial attempts at formalisation and by full mechanisation of various areas of mathematics in various proof assistants and various foundations. Areas that have been largely neglected for computer-assisted and machine-checked proofs are computability theory and logic: Fundamental results like Gödel's second incompleteness theorem in its stronger forms due to Kleene and Rosser, Löb's theorem, Post's theorem connecting the arithmetical hierarchy and Turing jumps, or the Friedberg-Muĉnik theorem solving Post's problem have not or only very recently been re-produced in proof assistants. This is due to the fact that making these arguments formal is several orders of magnitude more involved than formalising other areas of mathematics, due to the amount of invisible mathematics (a term coined by Andrej Bauer) involved.

In computability theory, invisible arguments occur mainly behind proofs that a certain intuitively sketched procedure is computable in – citing Emil Post – "forbidding, diverse and alien formalisms in which this [...] work of Gödel, Church, Turing, Kleene, Rosser [...] is embodied.". For instance, there have been various approaches of formalising Turing machines, all to the ultimate dissatisfaction of the respective authors, and none going further than constructing a universal machine and proving the halting problem undecidable. Professional computability theorist and teachers of computability theory thus rely on the informal Church Turing thesis to carry out their work and only argue the computability of described algorithms informally.

For computability theory, a way out was proposed in the 1980s by Fred Richman and developed during the last decade by Andrej Bauer: Synthetic computability theory, where one assumes axioms in a constructive foundation which essentially identify all (constructively definable) functions with computable functions. A drawback of the approach is that assuming such an axiom on top of the axiom of countable choice - which is routinely assumed in this branch of constructive mathematics and computable analysis - is that the law of excluded middle, i.e. classical logic, becomes invalid. Computability theory is however, as all mainstream branches of mathematics, making routine use of the axiom of excluded middle.

In the case of logic, the invisible mathematics usually is either centered around encoding formulas and proofs as numbers using Gödel or similar encodings or about provability arguments that certain results can be proved in restricted proof systems such as Peano arithmetic. In several settings, synthetic computability arguments can be employed to mechanise these proofs.

We observe that a slight foundational shift rectifies the situation: By basing synthetic computability theory in the Calculus of Inductive Constructions, the type theory underlying amongst others the Coq and Lean proof assistants, where countable choice is independent and thus not provable, axioms for synthetic computability are compatible with the law of excluded middle. This insight can be used to finally mechanise computability theory and logic, in an elegant, concise way where invisible arguments stay invisible: with Felix Jahn I have mechanised arguments related to many-one and truth-table reduction theory (published at CSL '23), Dominik Kirst and Benjamin Peters have presented Gödel's first incompleteness theorem in this style (at CSL '23), and in collaboration with Dominik Kirst and Niklas Mück I have given a proof of Post's hierarchy theorem (at CSL '24).

In this invited talk, I will give a broader overview of this line of research investigating a mechanised synthetic approach to logic and computability theory. In particular, I will discuss a Coq library of undecidability proofs, results in the theory of reducibility degrees, constructive reverse analysis of theorems, as well as generalised incompleteness results such as Löb's theorem.

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).
Editors: Jörg Endrullis and Sylvain Schmitz; Article No. 3; pp. 3:1–3:2

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

2012 ACM Subject Classification Theory of computation → Constructive mathematics

Keywords and phrases Synthetic mathematics, computability theory, logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.3

Category Invited Talk

Supplementary Material
Software: `https://github.com/uds-psl/coq-synthetic-computability/`
Software: `https://github.com/uds-psl/coq-library-fol`

Acknowledgements Joint work with Dominik Kirst, Gert Smolka, Felix Jahn, Niklas Mück, Janis Bailitis, Haoyi Zeng, and the contributors of the Coq Library of Undecidability Proofs.

Playing with Modalities

Elaine Pimentel[1]
Computer Science Department, University College London, UK

Carlos Olarte
LIPN, CNRS UMR 7030, Université Sorbonne Paris Nord, France

Timo Lang
Computer Science Department, University College London, UK

Robert Freiman
TU Wien, Austria

Christian G. Fermüller
TU Wien, Austria

Abstract

In this work, we will explore modalities through dialogical game lenses. Games provide a powerful tool for bridging the gap between intended and formal semantics, often offering a more conceptually natural approach to logic than traditional model-theoretic semantics.

We begin by exploring substructural calculi from a game semantic perspective, driven by intuitions about resource-consciousness and, more specifically, cost-sensitive reasoning. The game comes into full swing as we introduce cost labels to assumptions and a corresponding budget. Different proofs of the same end-sequent are interpreted as strategies for a player to defend a claim, which vary in cost. This leads to a labelled calculus, which can be viewed as a fragment of subexponential linear logic. We conclude this first part with a discussion of cut-admissibility for the proposed system.

In the second part, we show that our games offer an interesting insight also into modal logics. More precisely, we will focus on the modal logic **PNL**, characterised by Kripke frames with two types of disjoint and symmetric reachability relations. This framework is motivated by the study of group polarisation, where the opinions or beliefs of individuals within a group become more extreme or polarised after interaction. Our approach to reasoning about group polarisation is based on **PNL** and highlights a different aspect of formal reasoning about the corresponding models – using games and proof systems. We conclude by outlining potential directions for future research.

2012 ACM Subject Classification Theory of computation → Linear logic; Theory of computation → Modal and temporal logics; Theory of computation → Proof theory

Keywords and phrases Linear logic, modal logic, proof theory, game semantics

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.4

Category Invited Talk

Funding *Elaine Pimentel*: Pimentel has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement Number 101007627 and by the Leverhulme Project ECUMENICAL (RPG-2024-196).
Carlos Olarte: The work of Olarte has been partially supported by the SGR project PROMUEVA (BPIN 2021000100160) under the supervision of Minciencias (Ministerio de Ciencia Tecnología e Innovación, Colombia). Olarte acknowledges also support from the NATO Science for Peace and Security Programme through grant number G6133 (project SymSafe).

[1] Corresponding author.

1 Introduction

Modalities, both as formal constructs and as tools for reasoning, have been central to the development of logic and proof theory. In this work, we explore modalities through the lens of dialogical games, emphasising their potential to bridge the gap between formal semantics and conceptual intuition. Games not only offer a dynamic perspective on logical systems but also serve as a unifying framework for analysing the structure of proofs and resource management in a variety of logical settings.

We begin by examining substructural calculi, inspired by resource-sensitive reasoning. We introduce the concept of *prices* for resources (represented by formulas) into the game using the unary operator $!^a$, $a \in \mathbb{R}^+$, which shares some characteristic features with *subexponentials* in linear logic LL (SELL [14, 32]). Intuitively, a formula $!^a A$ represents a *permanent resource*: from $!^a A$, we can derive A as many times as needed, paying the price a each time.

We extend our game to this enriched language by incorporating a *budget* into the game states, which decreases whenever a price is paid. Different strategies for proving the same end-sequent can then be evaluated based on the budget required to execute them safely, *i.e.*, without incurring debt. This approach to resource-consciousness not only enhances the game but also translates naturally into a sequent system, where cost bounds for proofs are expressed as labels attached to sequents. By associating costs with proof steps, we provide a fine-grained analysis of proof strategies and their computational bounds.

We note that, up to this point, the content summarises the work presented in [28], where resources were considered only in *assumptions*. In this setting, sequents are restricted by limiting the occurrences of the modality $!^a$ *negatively*, thereby eliminating the need for a promotion rule.

In Section 2.2, we introduce new perspectives by allowing modalities in positive contexts. This includes the addition of "worse costs," linearisation of the cut formula, and tracking the use of contraction during the cut-elimination process.

In the second part of this paper, we present an overview of our work in [22], going beyond resource-awareness, and showing how games can illuminate modal logics. Specifically, we focus on the positive-negative modal logic (**PNL** [47]), characterised by Kripke frames with two disjoint and symmetric reachability relations. In **PNL**, individuals in a social network are identified with worlds of the frame, and the associated relations represent either "friends" (positive) or as "enemies" (negative). These relationships can be understood in different ways: Instead of genuine friendship or enduring enmity, they may simply mean agreement or disagreement on a particular issue. Our interest in **PNL** stems from its application in modelling phenomena such as group polarisation, where interactions amplify the extremity of opinions within a network. We show how the dialogical game lenses lead to both a semantic game and a provability game for (hybrid) extensions of **PNL**.

In semantic games [25], each instance is played over a formula F and a model $\mathbb{M}$ by two players, traditionally called I (or *Me*) and *You*. At every point in the game, one player acts as the proponent (**P**), while the other acts as the opponent (**O**) of the current formula. The set of actions at each stage is determined by the main connective of the current formula.

In contrast, provability games [29] do not concern truth in a specific model but rather *logical validity*. These games are also played by two participants, *Me* and *You*, and involve attacking assertions of formulas made by the other player and defending against these attacks.

We conclude this summary by showing how to transform the semantic game over single models into a provability game that characterises logical validity. This transformation led to *the first* Gentzen-style systems for variants of **PNL**, which modularly adapt to different frame properties by faithfully capturing the rules for *elementary* games.

Each part concludes with a discussion of future research directions and methodologies for combining and adapting the frameworks presented here to other logics and systems.

2 A game model for costs

Our starting point is a calculus for *affine intuitionistic linear logic* (aILL) [24]. Formulas in aILL are built from the grammar

$$A ::= p \mid \mathbf{0} \mid \mathbf{1} \mid A_1 \,\&\, A_2 \mid A_1 \oplus A_2 \mid A_1 \otimes A_2 \mid A_1 \multimap A_2 \mid !\,A.$$

with a denumerable infinite set of propositional variables $\{p, q, r, \ldots\}$, the units $\{\mathbf{0}, \mathbf{1}\}$, the binary connectives for additive conjunction and disjunction $\{\&, \oplus\}$, the multiplicative conjunction $\otimes$, the linear implication $\multimap$, and the exponential $!$.

Similar to modal connectives, the exponential $!$ in linear logic is not *canonical*, in the sense that, even having the same scheme for introduction rules, marking the exponentials with different labels does not preserve equivalence. That is, if $i \neq j$ then $!^i A \not\equiv !^j A$. Intuitively, this means that we can mark the exponential with *labels* taken from a set $\mathcal{I}$ organized in a pre-order $\preceq$ (*i.e.*, a reflexive and transitive relation), obtaining (possibly infinitely-many) exponentials $!^i$ for $i \in \mathcal{I}$. These are called *subexponentials* [14], and the respective proof system for linear logic with subexponentials is called SELL [33]. As in multi-modal systems, the pre-order determines the provability relation: for a general formula A, $!^b A$ *implies* $!^a A$ iff $a \preceq b$. Pre-ordering the labels (together with an upward closeness requirement) guarantees cut-elimination in SELL [14].

The algebraic structure of subexponentials, combined with their intrinsic structural properties (weakening and contraction) allow for the proposal of rich linear logic based frameworks. This opened a venue for proposing different multi-modal substructural logical systems [46], that encountered a number of different applications (see [37] for a survey).

In this paper, we will use subexponentials to model the notion of *costs*. We will start by considering the particular case where labels will be elements of $\mathbb{R}^+$, the set of non-negative real numbers, with the usual pre-order $\leq$. Formally, we substitute in aILL the exponential $!$ by the unary modal operators $!^a$ for each $a \in \mathbb{R}^+$.

We shall use A, B, C (resp. Γ, Δ) to range over formulas (resp. multisets of formulas). Sequents have the form $\Gamma \Rightarrow C$ where subformulas $!^a A$ will have a restriction to occur only *negatively* in the sequent.[2] We denote by $!\Gamma$ a set of formulas prefixed with $!^a$ for some (not necessarily the same) $a \in \mathbb{R}^+$.

The rules for the system $\mathcal{C}(\mathbb{R}^+)$ are depicted in Figure 1. Note that the cut rule is not included in our presentation of $\mathcal{C}$ and that weakening is present only implicitly, via the context Γ in the initial sequents. Furthermore, in rule *init*, p is a propositional variable and there is no right rule for $!$ in $\mathcal{C}(\mathbb{R}^+)$ since this connective only appears in negative polarity. We shall write $\vdash_{\mathcal{C}(\mathbb{R}^+)} S$ if the sequent S is provable in $\mathcal{C}(\mathbb{R}^+)$.

[2] The notion of polarity is the standard one: A subformula occurrence in the antecedent of a sequent is *negative* if it occurs in the scope of an even number (including 0) of contexts $([\cdot] \multimap B)$, and otherwise it is *positive*. For occurrences of a subformula in the consequent, one replaces "even" by "odd". The reason for this restriction will be made clear in Section 2.2.

$$\frac{\Gamma, A, B \Rightarrow C}{\Gamma, A \otimes B \Rightarrow C}\otimes_L \quad \frac{!\Gamma, \Delta_1 \Rightarrow A \quad !\Gamma, \Delta_2 \Rightarrow B}{!\Gamma, \Delta_1, \Delta_2 \Rightarrow A \otimes B}\otimes_R$$

$$\frac{!\Gamma, \Delta_1 \Rightarrow A \quad !\Gamma, \Delta_2, B \Rightarrow C}{!\Gamma, \Delta_1, \Delta_2, A \multimap B \Rightarrow C}\multimap_L \quad \frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \multimap B}\multimap_R \quad \frac{\Gamma, !^a A, A \Rightarrow C}{\Gamma, !^a A \Rightarrow C}\,!_L$$

$$\frac{\Gamma, A_i \Rightarrow B}{\Gamma, A_1 \,\&\, A_2 \Rightarrow B}\,\&_{L_i} \quad \frac{\Gamma \Rightarrow A \quad \Gamma \Rightarrow B}{\Gamma \Rightarrow A \,\&\, B}\,\&_R \quad \frac{\Gamma, A \Rightarrow C \quad \Gamma, B \Rightarrow C}{\Gamma, A \oplus B \Rightarrow C}\oplus_L \quad \frac{\Gamma \Rightarrow A_i}{\Gamma \Rightarrow A_1 \oplus A_2}\oplus_{R_i}$$

$$\frac{}{\Gamma, p \Rightarrow p}\,init \qquad \frac{}{\Gamma \Rightarrow \mathbf{1}}\,\mathbf{1}_R \qquad \frac{}{\Gamma, \mathbf{0} \Rightarrow C}\,\mathbf{0}_L$$

Figure 1 The sequent system $\mathcal{C}(\mathbb{R}^+)$.

2.1 Playing with subexponentials

We shall characterize $\mathcal{C}(\mathbb{R}^+)$ proofs as winning strategies (w.s.) in a two-player game, the players denoted **P** and **O**. As usual, we will interpret bottom-up proof search in sequent systems as a game where, at any given state, player **P** first chooses a formula of a sequent and, in the next step:

- if the rule has only one premise: **P** moves to the premise sequent of the corresponding introduction rule;
- if the rule has two premises either
  - **(i)** player **O** chooses a premise sequent in which the game continues; or
  - **(ii)** the game splits into independent subgames, where **P** has to win all of them if she wants to win the game.

The choice between (i) and (ii) depends on the nature of the rule: branching in *additive rules* is modelled as choices made by **O**, while branching in *multiplicative rules* involves **P** splitting the context into two disjoint parts, which then serve as the corresponding contexts for two subgames played in parallel. Consequently, the state of the game is represented by a *multiset of sequents*, with each sequent belonging to a distinct subgame.

Now, to capture the notion of *costs*, game states include a *budget* (modelled as a real number) that decreases whenever the rule $!_L$ is applied. This implies a cost a is incurred during dereliction, *i.e.*, when unpacking a formula stored within the modality $!^a$. Formally we have the following.

▶ **Definition 1** (The game $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$)**.** *$\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$ is a game of two players,* **P** *and* **O**. *Game states are tuples (H, b), where H is a finite multiset of sequents and $b \in \mathbb{R}$ is a "budget". $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$ proceeds in rounds, initiated by* **P***'s selection of a sequent S from the current game state. The successor state is determined according to rules that fit one of the two following schemes:*

(1) $(G \cup \{S\}, b) \quad \rightsquigarrow \quad (G \cup \{S'\}, b')$

(2) $(G \cup \{S\}, b) \quad \rightsquigarrow \quad (G \cup \{S^1\} \cup \{S^2\}, b)$

A round proceeds as follows: After **P** *has chosen a sequent $S \in H$ among the current game state, she chooses a rule instance r of $\mathcal{C}(\mathbb{R}^+)$ such that S is the conclusion of that rule. Depending on r, the round proceeds as follows:*

1. *If r is a unary rule different from $!_L$ with premise S', then the game proceeds in the game state $(G \cup \{S'\}, b)$.*
2. Budget decrease: *If $r = !_L$ with premise S' and principal formula $!^a A$, then the game proceeds in the game state $(G \cup \{S'\}, b - a)$.*

3. Parallelism: *If r is a binary rule with premises S_1, S_2 pertaining to a* multiplicative *connective, then the game proceeds as $(G \cup \{S_1\} \cup \{S_2\}, b)$.*
4. **O**-choice: *If r is a binary rule with premises S_1, S_2 pertaining to an* additive *connective, then* **O** *chooses $S' \in \{S_1, S_2\}$ and the game proceeds in the game state $(G \cup \{S'\}, b)$.*

A winning state *(for* **P***) is a game state (H, b) such that all $S \in H$ are initial sequents of $\mathcal{C}(\mathbb{R}^+)$ and $b \geq 0$.*

▶ **Definition 2** (Plays and strategies). *A* play *of $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$ on a game state (H, b) is a sequence $(H_1, b_1), (H_2, b_2), \ldots, (H_n, b_n)$ of game states, where $(H_1, b_1) = (H, b)$ and each (H_{i+1}, b_{i+1}) arises by playing one round on (H_i, b_i). A* strategy *(for* **P***) on a game state (H, b) is defined as a function telling* **P** *how to move in any given state. A strategy on (H, b) is a* winning strategy (w.s.) *if all plays following it eventually reach a winning state. We write $\models_{\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)} (H, b)$ if* **P** *has a w.s. in the $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$-game starting on (H, b).*

The intuitive reading of $\models_{\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)} (H, b)$ is: The budget b suffices to win the game H.

▶ **Example 3.** Consider the following well-known riddle:

> You have white and black socks in a drawer in a completely dark room. How many socks do you have to take out blindly to be sure of having a matching pair?

We can model the matching pair by the disjunction $(w \otimes w) \oplus (b \otimes b)$, and the act of drawing a random sock by the labelled formula $!^1(w \oplus b)$. The above question then becomes:

> What is the least budget n such that $\models_{\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)} (!^1(w \oplus b) \Rightarrow (w \otimes w) \oplus (b \otimes b), n)$?

The following play illustrates that $n = 3$ suffices, where $F = (w \otimes w) \oplus (b \otimes b)$ and $G = !^1(w \oplus b)$:

1. $(\{G \Rightarrow F\}, 3)$
2. $(\{G, w \oplus b, w \oplus b, w \oplus b \Rightarrow F\}, 0)$ (**P** plays $!^1{}_L$ $3\times$, budget decrease)
3. $(\{G, w, w \oplus b, w \oplus b \Rightarrow F\}, 0)$ (**O** chooses w)
4. $(\{G, w, b, w \oplus b \Rightarrow F\}, 0)$ (**O** chooses b)
5. $(\{G, w, b, b \Rightarrow F\}, 0)$ (**O** chooses b)
6. $(\{G, w, b, b \Rightarrow b \otimes b\}, 0)$ (**P** plays $\oplus_{R_2}$)
7. $(\{G, w, b \Rightarrow b\} \cup \{G, b \Rightarrow b\}, 0)$ (**P** plays $\otimes_R$, parallelism)

The other possible choices for **O** are similar or simpler, and show that $n = 2$ is not enough for winning the game.

We note that it is not necessary to consider all possible strategies in $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$: For example, **P** never needs to take the budget into account when deciding the next move. Also, it is easy to see that a $\mathcal{C}(\mathbb{R}^+)$-proof Ξ of a sequent S translates to a w.s. in $(\{S\}, b)$ for some *sufficiently large* budget b. Taking these observations together, one can prove the following.

▶ **Theorem 4** (Weak adequacy for $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$ [28]). *Let S be a sequent. Then*

$$\exists b \left(\models_{\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)} (\{S\}, b) \right) \quad \textit{iff} \quad \vdash_{\mathcal{C}(\mathbb{R}^+)} S$$

This is a *weak* adequacy since information about the budget b is lost in the proof theoretic representation. In other words, the game $\mathcal{G}_{\mathcal{C}}(\mathbb{R}^+)$ is more expressive than the calculus $\mathcal{C}(\mathbb{R}^+)$.

To overcome this discrepancy, we introduce a labelled extension of $\mathcal{C}(\mathbb{R}^+)$ that we call $\mathcal{C}^{\ell}(\mathbb{R}^+)$. A $\mathcal{C}^{\ell}(\mathbb{R}^+)$-proof is build from labelled sequents $b : \Gamma \Rightarrow A$ where $\Gamma \Rightarrow A$ is a sequent and $b \in \mathbb{R}^+$. The complete system is given in Figure 2. Now we can prove the desired correspondence.

labelled sequent system for $\mathcal{C}^\ell(\mathbb{R}^+)$

$$\frac{b : \Gamma, A, B \Rightarrow C}{b : \Gamma, A \otimes B \Rightarrow C}\ \otimes_L \qquad \frac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2 \Rightarrow B}{a + b : !\Gamma, \Delta_1, \Delta_2 \Rightarrow A \otimes B}\ \otimes_R$$

$$\frac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, B \Rightarrow C}{a + b : !\Gamma, \Delta_1, \Delta_2, A \multimap B \Rightarrow C}\ \multimap_L \qquad \frac{b : \Gamma, A \Rightarrow B}{b : \Gamma \Rightarrow A \multimap B}\ \multimap_R$$

$$\frac{b : \Gamma, A_i \Rightarrow B}{b : \Gamma, A_1 \,\&\, A_2 \Rightarrow B}\ \&_{L_i} \qquad \frac{a : \Gamma \Rightarrow A \quad b : \Gamma \Rightarrow B}{\max\{a, b\} : \Gamma \Rightarrow A \,\&\, B}\ \&_R$$

$$\frac{a : \Gamma, A \Rightarrow C \quad b : \Gamma, B \Rightarrow C}{\max\{a, b\} : \Gamma, A \oplus B \Rightarrow C}\ \oplus_L \qquad \frac{b : \Gamma \Rightarrow A_i}{b : \Gamma \Rightarrow A_1 \oplus A_2}\ \oplus_{R_i}$$

$$\frac{c : \Gamma, !^a A, A \Rightarrow C}{c + a : \Gamma, !^a A \Rightarrow C}\ !^a{}_L$$

$$\frac{}{0 : \Gamma, p \Rightarrow p}\ init \qquad \frac{}{0 : \Gamma \Rightarrow \mathbf{1}}\ \mathbf{1}_R \qquad \frac{}{0 : \Gamma, \mathbf{0} \Rightarrow A}\ \mathbf{0}_L \qquad \frac{a : \Gamma \Rightarrow A}{b : \Gamma \Rightarrow A}\ w_\ell(b \geq a)$$

Figure 2 The labelled sequent system $\mathcal{C}^\ell(\mathbb{R}^+)$.

▶ **Theorem 5** (Strong adequacy for $\mathcal{G}_\mathcal{C}(\mathbb{R}^+)$ [28]). $\models_{\mathcal{G}_\mathcal{C}(\mathbb{R}^+)} (\{\Gamma \Rightarrow A\}, b)$ *iff* $\vdash_{\mathcal{C}^\ell(\mathbb{R}^+)} b : \Gamma \Rightarrow A$.

This result can be further strengthened. In fact, proofs (and games) can be assigned a minimal budget, referred to as *the cost*: given a proof Ξ of a sequent, one can assign the label 0 to all initial sequents of Ξ and propagate the labels downward according to the rules of $\mathcal{C}^\ell(\mathbb{R}^+)$. However, the broader implications are even more interesting, as illustrated in the following example.

▶ **Example 6.** Suppose that a printer costs \$500 and it produces copies for \$0.1. Which is the budget needed for making 2 copies?

Since buying a printer and making a copy can be modelled as $!^{500}(!^{0.1}C)$, the goal is to find possible budgets for

$$b : \ !^{500}(!^{0.1}C) \Rightarrow C \otimes C$$

Now, there are many ways of proving this sequent in $\mathcal{C}^\ell(\mathbb{R}^+)$. For example, the proof below has a cost \$500.20:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{0 : \ C, C \Rightarrow C \otimes C}\ \otimes, init}{0.20 : \ !^{0.1}C \Rightarrow C \otimes C}\ !^{0.10} \times 2}{500.20 : \ !^{500}(!^{0.1}C) \Rightarrow C \otimes C}\ !^{500}$$

This proof corresponds to purchasing one printer and producing two copies from it.

Alternatively, one could overprice the scenario by purchasing two printers and making one copy with each, incurring a cost of \$1,000.20.

$$\dfrac{\dfrac{\dfrac{}{0:\ C, C \Rightarrow C \otimes C}\ \otimes, init}{0.20:\ !^{0.1}C, !^{0.1}C \Rightarrow C \otimes C}\ !^{0.10}}{1,000.20:\ !^{500}(!^{0.1}C) \Rightarrow C \otimes C}\ !^{500} \times 2$$

Hence, different proofs of the same sequent can lead to different costs. Nevertheless, cost-optimal strategies exist for all provable sequents, as the following result shows.[3]

▶ **Theorem 7** (Cost-optimal proofs [28]). *If* $\vdash_{\mathcal{C}(\mathbb{R}^+)} \Gamma \Rightarrow A$*, then there exists a smallest* b *such that* $\vdash_{\mathcal{C}^\ell(\mathbb{R}^+)} b : \Gamma \Rightarrow A$*.*

2.2 About cut-admissibility

We begin by noting that establishing cut-admissibility in $\mathcal{C}^\ell(\mathbb{R}^+)$ critically relies on the ability to define a computable function f that relates the cost of the end-sequent to the labels of the premises in the cut rule. Given that exponentials only occur negatively in $\mathcal{C}^\ell(\mathbb{R}^+)$, no cut steps involve banged formulas. This allows us to demonstrate that $f(a, b) = a + b$ is the *minimal* such function.

▶ **Theorem 8** (Negative-cut [28]). *For* $f(a, b) = a + b$*, the following cut rule is admissible in* $\mathcal{C}^\ell(\mathbb{R}^+)$*:*

$$\dfrac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, A \Rightarrow C}{f(a, b) : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}\ cut_\ell$$

Moreover, whenever cut_ℓ *is admissible w.r.t. a given* f'*, then* $a + b \leq f'(a, b)$*.*

It turns out that extending cost-conscious reasoning to modalities occurring *positively* in sequents is far from trivial. While an intuitive game-theoretic interpretation of promotion could be provided in the style of [16], this *does not* align with a proof-theoretic notion of cut-admissibility. This is due to the inherent difficulty in defining a functional notion of the cut-label, as demonstrated below.

Let $\mathcal{CP}^\ell(\mathbb{R}^+)$ be the system resulting from $\mathcal{C}^\ell(\mathbb{R}^+)$ by adding the following *labelled promotion rule*

$$\dfrac{b : \Gamma^{\leq !^a} \Rightarrow A}{b : \Gamma \Rightarrow\ !^a A}\ !^a{}_R$$

where $\Gamma^{\leq !^a}$ denotes all formulas in Γ which are of the form $!^c B$ and $a \geq c$.

The question that arises is whether the cut-admissibility result can be extended to $\mathcal{CP}^\ell(\mathbb{R}^+)$. To address this, consider the following derivation:

$$\dfrac{\dfrac{b_1 :\Rightarrow A}{b_1 :\Rightarrow\ !^a A}\ !^a{}_R \quad \dfrac{b_2 : \Delta, !^a A, A \Rightarrow C}{b_2 + a : \Delta, !^a A \Rightarrow C}\ !^a{}_L}{b_1 + b_2 + a :\ \Delta \Rightarrow C}\ cut$$

[3] We note that the proof of this result is non-constructive!

This is usually reduced to

$$\dfrac{b_1 :\Rightarrow A \qquad \dfrac{b_1 :\Rightarrow !^a A \quad b_2 : \Delta, !^a A, A \Rightarrow C}{b_1 + b_2 : \Delta, A \Rightarrow C}\ cut}{2b_1 + b_2 : \Delta \Rightarrow C}\ cut$$

where the upper cut has a smaller rank, and the lower cut has a smaller degree than the original cut. However, this approach fails in the labelled setting because, whenever $a < b_1$, the label increases.

Although alternative reduction methods could be explored, the following result shows that it is impossible to define a labelled cut rule for $\mathcal{CP}^\ell(\mathbb{R}^+)$ where the label of the conclusion depends solely on the labels of the premises. We include the proof, as it is highly insightful.

▶ **Theorem 9** (Impossible-cut [28]). *There is no function $f : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ such that the rule*

$$\dfrac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, A \Rightarrow C}{f(a,b) : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}\ cut$$

is admissible in $\mathcal{CP}^\ell(\mathbb{R}^+)$.

Proof. Let p, q be different propositional variables, and let $A^{\otimes n}$ denote the n-fold multiplicative conjunction of a formula A. The sequents

$$a : !^{1/k} p \Rightarrow !^{1/k} p^{\otimes(k\cdot a)} \qquad \text{and} \qquad b : !^{1/k} p^{\otimes(k\cdot a)} \Rightarrow p^{\otimes(k\cdot k\cdot a\cdot b)}$$

are provable in $\mathcal{CP}^\ell(\mathbb{R}^+)$ for all natural numbers a, b, k. The smallest label f which makes their cut conclusion $f : !^{1/k} p \Rightarrow p^{\otimes(k\cdot k\cdot a\cdot b)}$ provable in $\mathcal{CP}^\ell(\mathbb{R}^+)$ is $k \cdot a \cdot b$, which is not a function on the premise labels a, b. ◀

The theorem above indicates that, to find an admissible labelled cut rule, we must either:

1. restrict the form of the cut formula;
2. allow the labelling function f to incorporate more information from the premises than just their labels;
3. keep track of the use of contraction in the cut-elimination process.

We shall explore next different fragments and (admissible) cut-like rules that can be proposed for $\mathcal{CP}^\ell(\mathbb{R}^+)$.

2.2.1 Infinite costs

We start by observing that the inclusion of "worse costs" entails a trivial labelling that makes cut admissible. Let $\mathbb{R}^+_\infty$ be the completion of $\mathbb{R}^+$ with ∞ and $\mathcal{CP}^\ell(\mathbb{R}^+_\infty)$ the corresponding labeled proof system with *decreasing* for $b \leq a$ being defined as follows:

- If $a, b \neq \infty$, $a - b$ is defined as usual;
- If $a = \infty$, then $a - b = \infty$.

In the following theorem, the cut formula A is an arbitrary formula (containing, possibly, positive and/or negative occurrences of the modality $!^a$).

▶ **Theorem 10** (Infinite-cut). *The following rule is admissible in $\mathcal{CP}^\ell(\mathbb{R}^+_\infty)$*

$$\dfrac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, A \Rightarrow C}{\infty : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}\ cut_\infty$$

The proof follows the same steps of the cut-elimination proof for SELL [14, 33], using natural extensions of invertibility and permutability of rules to the labelled case.

But this still does not define a computable function relating the labels of the premises and the conclusion of the cut rule.

2.2.2 Linearity

Now we show cases where the cut formula is restricted, starting with the case where the cut formula is !-free.

▶ **Theorem 11** (Linear-cut). *Let A be a formula with no occurrences of $!^a$. Then, the following rule is admissible in $\mathcal{CP}^\ell(\mathbb{R}^+)$*

$$\frac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, A \Rightarrow C}{a + b : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}\ cut_L$$

Moreover, if $a : \Gamma \Rightarrow C$ is provable using cut_L, then there is a cut-free proof of $a' : \Gamma \Rightarrow C$ with $a \geq a'$.

The proof uses a standard cut-reduction strategy for SELL, observing in each case that the reduction of the label is possible.

Still, forcing cut formulas to be linear seems to be a very severe restriction to impose. We will now consider another, and less limiting, syntactic restriction on the cut formula.

▶ **Definition 12.** *A formula of the form $!^a A$ is* simply exp-labelled *if $a \neq 0$ and A is bang-free.*

Since the formulas used in the proof of Theorem 9 can be simply exp-labelled, it is clear that we cannot expect to find an admissible cut rule for all simply exp-labelled cut formulas where the labelling depends solely on the labels of the premises. However, we can also incorporate the information from the label a in the simply exp-labelled formula $!^a A$, as follows.

▶ **Theorem 13** (Exp-labelled-cut [27]). *For any simply exp-labelled formula $!^a A$, the following cut rule is admissible in $\mathcal{CP}^\ell(\mathbb{R}^+)$:*

$$\frac{b_1 : !\Gamma, \Delta_1 \Rightarrow !^a A \quad b_2 : !\Gamma, \Delta_2, !^a A \Rightarrow C}{f(b_1, b_2, a) : !\Gamma, \Delta_1, \Delta_1 \Rightarrow C}\ cut_{el}$$

where $f(b_1, b_2, a) = b_2 + \lfloor b_2/a \rfloor \cdot b_1$.

The intuition behind this labelling is as follows: if the right subproof R of the cut_{el} ends with the label b_2, then the formula $!^a A$ can be unpacked at most $\lfloor b_2/a \rfloor$ times within a multiplicative subtree of R. Therefore, we can assume that the rule $!^a{}_L$ is applied only $\lfloor b_2/a \rfloor$ times on such a subtree.

2.2.3 Accumulated costs

We will end the part of substructural modalities with a new approach towards cut-admissibility, where we keep track of the use of contraction in the cut-elimination process. The idea is that, if proving A costs b, then any use of A must pay this "extra cost". For that, we introduce the following notation.

▶ **Definition 14.** *Let* $\mathcal{E} = \{a_b \mid a, b \in \mathbb{R}^+\}$ *be such that*

1. $a_b +_{\mathcal{E}} c_d = a + b + c + d$.
2. $a_b \geq_{\mathcal{E}} a_c$ *(i.e., the ordering* $\geq_{\mathcal{E}}$ *ignores the subindices).*
3. $a_b >_{\mathcal{E}} c_d$ *iff* $a > c$.

For any formula $A \in \mathcal{CP}^{\ell}(\mathbb{R}^+)$*, we define* $[A]_c$ *as the formula that substitutes any modality* $!^{a_b}$ *with* $!^{a_{b+c}}$.

Hence $\mathcal{CP}^{\ell}(\mathbb{R}^+)$ can be slightly modified so that sequent labels belong to $\mathbb{R}^+$, while modal labels belong to $\mathcal{E}$. Due to the ordering above, the promotion of $!^{a_0}$ has the same effect/constraints that the promotion of $!^{a_b}$. However, the dereliction of the latter requires a greater budget ($a + b$ instead of a). Moreover, the equivalence $!^{a_b}A \equiv !^{a_c}A$ can be proven, each direction requiring a different budget. Finally, note that $\mathcal{E}_0 = \{a_0 \mid a \in \mathbb{R}^+\} \simeq \mathbb{R}^+$, that is, each element $a \in \mathbb{R}^+$ can be seen as the equivalence class of a_0 in $\mathbb{R}^+ \times \mathbb{R}^+$ modulo $\mathbb{R}^+$. We will abuse of the notation and continue representing the resulting system by $\mathcal{CP}^{\ell}(\mathbb{R}^+)$, also unchanging the representation of sequents.

The following lemma has a straightforward proof.

▶ **Lemma 15.** *If* $b : \Gamma, [A]_c \Rightarrow C$ *then* $b' : \Gamma, A \Rightarrow C$ *with* $b \geq b'$*. More generally, if* $b : \Gamma, [A]_c \Rightarrow C$ *and* $c \geq c'$ *then* $b' : \Gamma, [A]_{c'} \Rightarrow C$ *with* $b \geq b'$.

The next definition restricts the occurrence of unbounded modalities only under linear implication.

▶ **Definition 16.** *We say that* A *is* $\multimap$*-linear if for all subformulas of the form* $B \multimap C$ *in* A*,* B *is bang-free.*

The following result presents the admissibility of an extended form of the cut rule, where the budget information from the left premise is passed to the cut-formula in the right premise. Observe that the label of the conclusion is now a function of the labels of the premises. Moreover, the cut-reduction is *label preserving*, meaning that the budget monotonically decreases in the cut-elimination process.

▶ **Theorem 17** ($\multimap$-linear-cut)**.** *The following rule is admissible*

$$\frac{a : !\Gamma, \Delta_1 \Rightarrow A \quad b : !\Gamma, \Delta_2, [A]_a \Rightarrow C}{a + b : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}\ cut_{LL} \quad A \text{ is a } \multimap\text{-linear formula}$$

Moreover, if $b : \Gamma \Rightarrow C$ *is provable using* cut_{LL}*, then there is a cut-free proof of* $b' : \Gamma \Rightarrow C$ *with* $b \geq b'$.

Proof. We will illustrate some cases.

- Note that: $[!^{a_b}A]_c = !^{a_{b+c}}[A]_c$; the promotion of $!^{a_b}A$, bottom-up, results in a context of ! formulas (that can be contracted at will); and the dereliction of $!^{a_b}[A]_c$ decreases the budget in $a + b$. Hence,

$$\frac{\dfrac{c : (!\Gamma)^{\leq !^{a_b}} \Rightarrow A}{c : !\Gamma, \Delta_1 \Rightarrow !^{a_b}A} \quad \dfrac{d : !\Gamma, \Delta_2, [A]_c, !^{a_{b+c}}[A]_c \Rightarrow C}{a + b + c + d : !\Gamma, \Delta_2, !^{a_{b+c}}[A]_c \Rightarrow C}}{a + b + 2c + d : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  reduces to

$$\frac{c : (!\Gamma)^{\leq !^{a_b}} \Rightarrow A \qquad \dfrac{c : !\Gamma \Rightarrow !^{a_b}A \quad d : !\Gamma, !^{a_{b+c}}[A]_c, \Delta_2, [A]_c \Rightarrow C}{c + d : !\Gamma, \Delta_2, [A]_c \Rightarrow C}}{2c + d : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  where the "extra cost" a_b disappears after the reduction.

- Note that $[A \otimes B]_c = [A]_c \otimes [B]_c$. Here, let $c = c_1 + c_2$:

$$\dfrac{\dfrac{c_1 : !\Gamma, \Delta_1' \Rightarrow A \quad c_2 : !\Gamma, \Delta_1'' \Rightarrow B}{c : !\Gamma, \Delta_1 \Rightarrow A \otimes B} \quad \dfrac{b : !\Gamma, \Delta_2, [A]_c, [B]_c \Rightarrow C}{b : !\Gamma, \Delta_2, [A \otimes B]_c \Rightarrow C}}{b + c : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  reduces to

$$\dfrac{c_1 : !\Gamma, \Delta_1' \Rightarrow A \quad \dfrac{c_2 : !\Gamma, \Delta_1'' \Rightarrow B \quad b : !\Gamma, \Delta_2, [A]_{c_1}, [B]_{c_2} \Rightarrow C}{b + c_2 : !\Gamma, \Delta_1'', \Delta_2, [A]_{c_1} \Rightarrow C}}{b + c : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  It is worth noticing that in the first derivation, the cost $c = c_1 + c_2$ is "charged" to $A \otimes B$ (in the formula $[A \otimes B]_c$) while in the second one, in a finer way, the cost c_1 is charged to A and c_2 to B.

- The case of implication explains the restriction we impose. Here $b = b_1 + b_2$:

$$\dfrac{\dfrac{c : !\Gamma, \Delta_1, A \Rightarrow B}{c : !\Gamma, \Delta_1 \Rightarrow A \multimap B} \quad \dfrac{b_1 : !\Gamma, \Delta_2' \Rightarrow [A]_c \quad b_2 : !\Gamma, \Delta_2'', [B]_c \Rightarrow C}{b : !\Gamma, \Delta_2, [A \multimap B]_c \Rightarrow C}}{c + b : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  reduces to

$$\dfrac{b_1 : !\Gamma, \Delta_2' \Rightarrow A \quad \dfrac{c : !\Gamma, \Delta_1, [A]_{b_1} \Rightarrow B \quad b_2 : !\Gamma, \Delta_2'', [B]_c \Rightarrow C}{c + b_2 : !\Gamma, \Delta_1, \Delta_2'', [A]_{b_1} \Rightarrow C}}{c + b : !\Gamma, \Delta_1, \Delta_2 \Rightarrow C}$$

  Note that the reduction above is correct since A does not have occurrences of $!^a$ and then $[A]_c = [A]_{b_1} = A$. ◀

2.3 Discussion – part I

This research line offers at least three promising directions for future exploration.

First, the work initiated in [28] highlights that our games and systems provide more precise control over resources appearing negatively in sequents, unlocking new opportunities for analysing the problem of comparing proofs. For instance, studying proof costs in labelled calculi could reveal deeper links between labels and computational bounds [2]. Similarly, examining the interplay between resource budgets and the complexity of the cut-elimination process, particularly within the multiplicative-(sub)exponential fragment, presents considerable opportunities [40, 41].

Second, there is substantial value in investigating how the dialogue games we have developed align with the framework of concurrent games [1, 15, 13]. Understanding these connections could enrich our framework and provide new perspectives on resource management in proof theory.

Lastly, an essential direction involves addressing compositionality in dialogue games governed by the cut rule. Regardless of the specific approach taken to achieve cut-admissibility, ensuring compositionality remains a critical and promising challenge [34].

3 A game model for polarisation

We now turn to the study of modalities in the classical setting, focusing on the positive-negative modal logic **PNL** with nominals [47, 35]. This logic is based on Kripke frames with two disjoint and symmetric reachability relations. Here we will outline the construction of an adequate semantic game for **PNL**, its transformation into a provability game, and the derivation of a corresponding sequent system. This opens a discussion on how to generalise this method to other modal systems.

We begin with a brief discussion of games for modal logics and the motivation for hybrid extensions. As studied in [9] and further developed in [19], extending Hintikka games [25] dialogue game to modal logic is conceptually straightforward: in addition to the current roles of the players and the current formula F, one only has to keep track of the current world w in the model. However, this extension introduces an unfortunate drawback: the *game trees*, *i.e.*, labelled trees whose nodes are game states, are no longer determined solely by the syntax of the formula, but instead depend on the relational structure of the model. This is in stark contrast to semantic games for propositional logic, where semantic information is required only at the final stage to determine the winner. The loss of uniformity in game trees across all models is a significant limitation of this approach.

As in [9, 18], we address this problem by turning to hybrid logic [10, 12, 11], allowing explicit references to worlds and the accessibility relation within the object language.

Let $\mathsf{A} = \{\mathsf{a}, \mathsf{b}, \ldots\}$ be a non-empty set of agents, $\mathsf{At} = \{p, q, \ldots\}$ be a countable set of propositional variables, and $N = \{i, j, \ldots\}$ be a countable set of *nominals*. The language of **PNL** is generated by the following grammar

$$F ::= p \mid \neg F \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid R^+(i,j) \mid R^-(i,j) \mid \Diamond\!\!\!\!+ F \mid \Diamond\!\!\!\!- F \mid [A]F$$

where $p \in \mathsf{At}$, and $i, j \in N$. Formulas of the form p, $R^+(i,j)$, or $R^-(i,j)$ are called *elementary*. We shall use F, G, H to range over formulas. The propositional connectives $\top$, $\bot$, $\to$, and the (dual) modalities $\boxplus$ and $\boxminus$ can be obtained in the usual way.

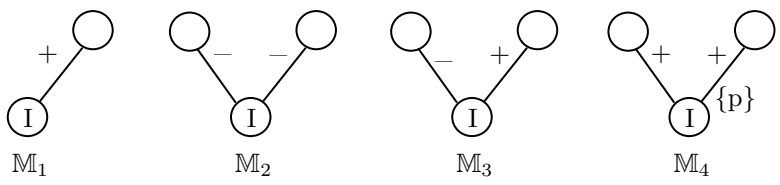
Intuitively, nominals are used as names for worlds of the model, while the propositions $R^{\pm}(i,j)$ state that agent i is a *friend*/*enemy* (or, more generally, *agrees*/*disagrees*) with j. The formula $\Diamond\!\!\!\!+ F$ (resp. $\Diamond\!\!\!\!- F$) states that F holds for a friend (resp. an enemy). The global modality $[A]F$ states that F holds for all the agents. We use $R^{\pm}$ to denote either R^+ or R^-, and $\Diamond^{\pm}$ to denote either $\Diamond\!\!\!\!+$ or $\Diamond\!\!\!\!-$.

A model $\mathbb{M}$ is a tuple $\langle \mathsf{A}, \mathsf{R}^+, \mathsf{R}^-, \mathsf{V}, \mathsf{g} \rangle$ where A is a set (of agents), $\mathsf{g} : N \to \mathsf{A}$ is called *denotation function*, $\mathsf{R}^+, \mathsf{R}^- \subseteq \mathsf{A} \times \mathsf{A}$, and $\mathsf{V} : \mathsf{At} \to \mathcal{P}(\mathsf{A})$. A model is a **PNL**-model if:

- g is surjective, *i.e.*, every agent has a name;
- R^+ is reflexive; and
- R^+ and R^- are both symmetric and non-overlapping, *i.e.*, for all $\mathsf{a}, \mathsf{b} \in \mathsf{A}$, $(\mathsf{a}, \mathsf{b}) \notin \mathsf{R}^+$ or $(\mathsf{a}, \mathsf{b}) \notin \mathsf{R}^-$.

The Kripke semantics of **PNL** is in Figure 3. A formula F is true over $\mathbb{M}$, written $\mathbb{M} \Vdash F$ iff $\mathbb{M}, \mathsf{a} \Vdash F$, for all agent $\mathsf{a} \in \mathsf{A}$. For a set of formulas Δ, we write $\mathbb{M} \models \Delta$ iff $\mathbb{M} \Vdash \Delta$ for all $F \in \Delta$. A formula F is valid iff $\mathbb{M} \Vdash F$ for every **PNL**-model $\mathbb{M}$. For a class of models $\mathfrak{M}$, we write $\Delta \models_{\mathfrak{M}} F$ iff $\mathbb{M} \Vdash F$ for every model $\mathbb{M} \in \mathfrak{M}$ with $\mathbb{M} \models \Delta$.

▶ **Example 18.** Consider the following models (omitting self loops for R^+):



$\mathbb{M}, \mathsf{a} \Vdash p$	iff $\mathsf{a} \in \mathsf{V}(p)$	$\mathbb{M}, \mathsf{a} \Vdash \neg F$	iff $\mathbb{M}, \mathsf{a} \nVdash F$
$\mathbb{M}, \mathsf{a} \Vdash F \wedge G$	iff $\mathbb{M}, \mathsf{a} \Vdash F$ and $\mathbb{M}, \mathsf{a} \Vdash G$	$\mathbb{M}, \mathsf{a} \Vdash F \vee G$	iff $\mathbb{M}, \mathsf{a} \Vdash F$ or $\mathbb{M}, \mathsf{a} \Vdash G$
$\mathbb{M}, \mathsf{a} \Vdash R^{\pm}(i,j)$	iff $(\mathsf{g}(i), \mathsf{g}(j)) \in \mathsf{R}^{\pm}$		
$\mathbb{M}, \mathsf{a} \Vdash \Diamond^{\pm} F$	iff there is $j \in N$ such that $\mathbb{M}, \mathsf{g}(j) \Vdash R^{\pm}(i,j)$ and $\mathbb{M}, \mathsf{g}(j) \Vdash F$		
$\mathbb{M}, \mathsf{a} \Vdash [A]F$	iff $\mathbb{M}, \mathsf{g}(j) \Vdash F$, for all $j \in N$.		

Figure 3 Kripke semantics for **PNL**.

The following holds:

- $\mathbb{M}_1$: I have a friend where $\neg p$: $\mathbb{M}_1, I \Vdash \Diamond\!\!\!\!-\neg p$;
- $\mathbb{M}_2$: All *my* enemies do not believe in p: $\mathbb{M}_2, I \Vdash \boxminus \neg p$;
- $\mathbb{M}_3$: I have an enemy: $\mathbb{M}_3, I \Vdash \Diamond\!\!\!\!-\top$;
- $\mathbb{M}_4$: Everybody has a friend where p: $\mathbb{M}_4, \mathsf{a} \Vdash [A]\Diamond\!\!\!\!-p$ for any agent a.

3.1 Playing with models

Before starting playing, remember that in a **PNL**-model $\mathbb{M}$, every agent a has a name i, *i.e.*, there exists $i \in N$ s.t. $\mathsf{g}(i) = \mathsf{a}$. Hence, from now on, we will internalise the nominals, identifying an agent a with its respective nominal i.

The *semantic game* is played over a **PNL**-model $\mathbb{M} = (\mathsf{A}, \mathsf{R}^{+}, \mathsf{R}^{-}, \mathsf{V}, \mathsf{g})$ by two players, *Me* (or I) and *You*, who argue about the truth of a formula F at an agent i. At each stage of the game, one player acts as *proponent*, while the other acts as *opponent* of the claim that F is true at i.

We represent the situation where I am the proponent (and *You* are the opponent) by the *game state* $\mathbf{P}, i : F$, and the situation where I am the opponent (and *You* are the proponent) by $\mathbf{O}, i : F$.

We call a game state *elementary* if its involved formula is elementary. For a game state g, we denote the game starting at g over the model $\mathbb{M}$ by $\mathbf{G}_{\mathbb{M}}(g)$.

The game over a **PNL**-model $\mathbb{M}$ proceeds by reducing the involved formula F to an elementary formula by following the rules described in Figure 4.[4]

In general, every two-person, zero-sum, win-lose game is usually represented by a game tree. In our case, the root of the game tree representing the game $\mathbf{G}_{\mathbb{M}}(g)$ is g. The children of each node in the game tree are exactly the possible choices of the corresponding player. For instance, if $h = \mathbf{P}, i : F_1 \wedge F_2$ appears in the game tree, then its children are $\mathbf{P}, i : F_1$ and $\mathbf{P}, i : F_2$. Each node in the tree is labelled either "I", or "Y", depending on which player is to move in the corresponding game state, and we label the nodes $\mathbf{P}, i : \neg F$ and $\mathbf{O}, i : \neg F$ with "I" (even though there is no choice involved in these game states). For instance, the node corresponding to the game state h above is "Y", since it is *Your* choice in $\mathbf{P} : F_1 \wedge F_2$. The leaves of the tree receive the label of the winning player. A *run* of the game is a maximal path through the game tree.

Now we are ready to define winning strategies and state the main result of this section: the adequacy of the proposed game semantics with respect to the Kripke semantics for **PNL**.

▶ **Definition 19.** *A* strategy *for* Me *in the game* $\mathbf{G}_{\mathbb{M}}(g)$ *is a subtree* σ *of the associated game tree such that:* ***(1)*** $g \in \sigma$, ***(2)*** *if* $h \in \sigma$ *is a node labelled "Y", then all children of* h *are in* σ, ***(3)*** *if* $h \in \sigma$ *is a node labelled "I", then exactly one child of* h *is in* σ. *The strategy* σ *is called* winning *if all leaves in the tree* σ *are labelled "I". (Winning) strategies for* You *are defined dually.*

[4] The outcome of the game state $\mathbf{Q}, k : R^{\pm}(i,j)$ is independent of k (it only depends on the underlying model $\mathbb{M}$). Hence, we write $\mathbf{Q}, \_ : R^{\pm}(i,j)$ instead of $\mathbf{Q}, k : R^{\pm}(i,j)$.

($\mathbf{P}_\wedge$) At $\mathbf{P}, i : F_1 \wedge F_2$, *You* choose between $\mathbf{P}, i : F_1$ and $\mathbf{P}, i : F_2$ to continue the game.

($\mathbf{O}_\wedge$) At $\mathbf{O}, i : F_1 \wedge F_2$, *I* choose between $\mathbf{O}, i : F_1$ and $\mathbf{O}, i : F_2$ to continue the game.

($\mathbf{P}_\vee$) At $\mathbf{P}, i : F_1 \vee F_2$, *I* choose between $\mathbf{P}, i : F_1$ and $\mathbf{P}, i : F_2$ to continue the game.

($\mathbf{O}_\vee$) At $\mathbf{O}, i : F_1 \vee F_2$, *You* choose between $\mathbf{O}, i : F_1$ and $\mathbf{O}, i : F_2$ to continue the game.

($\mathbf{P}_\neg$) At $\mathbf{P}, i : \neg F$, the game continues with $\mathbf{O}, i : F$.

($\mathbf{O}_\neg$) At $\mathbf{O}, i : \neg F$, the game continues with $\mathbf{P}, i : F$.

($\mathbf{P}_{\Diamond^\pm}$) At $\mathbf{P}, i : \Diamond^\pm F$, *I* choose a nominal j, and *You* decide whether the game ends in the state $\mathbf{P}, \_ : R^\pm(i, j)$ or continues with $\mathbf{P}, j : F$.

($\mathbf{O}_{\Diamond^\pm}$) At $\mathbf{O}, i : \Diamond^\pm F$, *You* choose j, and *I* choose between $\mathbf{O}, \_ : R^\pm(i, j)$ and $\mathbf{O}, j : F$.

($\mathbf{P}_{[A]}$) At $\mathbf{P}, i : [A]F$, *You* choose a nominal j and the game continues with $\mathbf{P}, j : F$.

($\mathbf{O}_{[A]}$) At $\mathbf{O}, i : [A]F$, *I* choose a nominal j, and the game continues with $\mathbf{O}, j : F$.

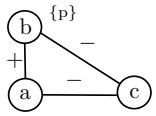
($\mathbf{P}_{el}$) Let F_e be an elementary formula. *I* win and *You* lose at $\mathbf{P}, i : F_e$ iff $\mathbb{M}, i \models F_e$. Otherwise, *You* win and *I* lose.

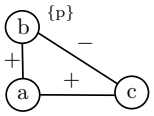
($\mathbf{O}_{el}$) At $\mathbf{O}, i : F_e$, *I* win and *You* lose iff $\mathbb{M}, i \not\models F_e$. Otherwise, *You* win and *I* lose.

Figure 4 Semantic game given a **PNL**-model $\mathbb{M}$.

▶ **Theorem 20** (Adequacy - semantic games [22]). *Let $\mathbb{M}$ be a **PNL**-model, a an agent with nominal i, and F a formula.*
(1) I *have a winning strategy for* $\mathbf{G}_\mathbb{M}(\mathbf{P}, i : F)$ *iff* $\mathbb{M}, \mathsf{a} \models F$.
(2) You *have a winning strategy for* $\mathbf{G}_\mathbb{M}(\mathbf{P}, i : F)$ *iff* $\mathbb{M}, \mathsf{a} \not\models F$.

▶ **Example 21** ([22]). Let $(\mathbf{4B}) = ((\Diamond\Box p \vee \Box\Diamond p) \to \Box p) \wedge ((\Diamond\Box p \vee \Box\Diamond p) \to \Diamond p)$. This formulas specifies *local balance* [35] and captures the idea that "the enemy of my enemy is my friend", "the friend of my enemy is my enemy", and "the friend of my friend is my friend". *I* have a winning strategy for the game $\mathbf{P}, \mathsf{a} : 4B$ on $\mathbb{M}_1$ while *You* have a winning strategy for the same game on $\mathbb{M}_2$ where (omitting self-loops for R^+):

$\mathbb{M}_1 =$ 

$\mathbb{M}_2 =$ 

For $\mathbb{M}_1$, in the first conjunct, *I* pick ($\mathbf{P}_\vee$) $\Box p$ and then b in ($\mathbf{P}_\Box$); for the second conjunct, *I* pick the first disjunction in $F = (\Box\Diamond p \vee \Diamond\Box p) \to \Diamond p)$ where, in any of *Your* choices ($\mathbf{P}_\neg$ followed by $\mathbf{O}_\vee$ and $\mathbf{O}_{\Diamond^\pm}$), *I* win all the elementary states. For $\mathbb{M}_2$, *I* do not have a winning strategy for the second conjunct: *I* can neither win $\Diamond p$ (no R^- successor), nor the first disjunct in F above since, after $\mathbf{P}_\neg$, *You* choose ($\mathbf{O}_\vee$) $\Box\Diamond p$ and select c and then b ($\mathbf{O}_{\Diamond^\pm}$) where p holds and *You* win. See the complete game in our tool [23].

3.2 Playing all models

We now leverage semantic games to **PNL**-provability games. The key observation is that the rules of the semantic game remain independent of the underlying model, except at the level of elementary game states.

(Dupl) If no state in D is underlined, I can choose a non-elementary $g \in D$ and the game continues with $D \bigvee g$.

(Sched) If no state in $D = D' \bigvee g$ is underlined, and g is non-elementary, I can choose to continue the game with $D' \bigvee \underline{g}$.

(Move) If $D = D' \bigvee \underline{g}$ then the player who is to move in the semantic game $\mathbf{G}(g)$ at g makes a legal move to the game state g' and the game continues with $D' \bigvee g'$.

(End) The game ends if there are no non-elementary game states left in D, or if no game state is underlined and I win according to Definition 22. Otherwise, I must move according to **(Dupl)** or **(Sched)**.

Figure 5 Rules for the provability game.

The *provability game* $\mathbf{DG}(\mathbf{P}, i : F)$ can be thought of as *Me* and *You* playing all semantic games $\mathbf{G}(\mathbf{P}, i : F)$ over all **PNL**-models $\mathbb{M}$ simultaneously. We point out that the rules of the semantic game do not depend on the structure of $\mathbb{M}$ but merely on F. Truth degrees are only needed at the atomic level to determine who wins the particular run of the game. This allows us to require players to play "blindly", *i.e.*, without explicitly referencing a model $\mathbb{M}$. Clearly, if I have a winning strategy in such a game, then I can win in $\mathbf{G}_{\mathbb{M}}(\mathbf{P}, i : F)$, for every $\mathbb{M}$, making this strategy an adequate witness of logical validity.

Provability game states are finite multisets of the game states defined in Section 3.1. We denote by $g_1 \bigvee ... \bigvee g_n$ the provability game state $\{g_1, ..., g_n\}$. We write $D_1 \bigvee D_2$ for the multiset sum $D_1 + D_2$ and $D \bigvee g$ for $D + \{g\}$. A provability state is called *elementary* if all its game states are elementary. We use $\mathbf{DG}(D)$ to denote the provability game starting at D.

▶ **Definition 22.** *Let D^{el} denote the provability state consisting of the elementary game states of D.* I *win and* You *lose at D if for every **PNL**-model there is a game state in D^{el} where* I *win the corresponding semantic game.*

In the provability game, I additionally take the role of a *scheduler*, deciding which game is to be played next. We signal the chosen game state by underlining it as in $\underline{g}$.

▶ **Definition 23.** *The rules of the provability game are in Figure 5. Infinite runs, and runs that end in elementary provability states where* I *do not win according to Definition 22, are winning for* You *and losing for* Me. ***(Dupl)*** *is referred to as the* duplication rule *and* ***(Sched)*** *as the* scheduling*, or* underlining rule*.*

▶ **Theorem 24** (Adequacy - provability games [22])**.** I *have a winning strategy in $\mathbf{DG}(D)$ iff for every **PNL**-model $\mathbb{M}$, there is some $g \in D$ such that* I *have a winning strategy in $\mathbf{G}_{\mathbb{M}}(g)$.*

▶ **Corollary 25.** *The formula F is **PNL**-valid iff* I *have a winning strategy in $\mathbf{DG}(\mathbf{P}, i : [A]F)$.*

▶ **Example 26.** Consider the game $\mathbf{P}, i : p \vee \neg p$. I duplicate the game state in the first round and the game continues with the provability state $\mathbf{P}, i : p \vee \neg p \bigvee \mathbf{P}, i : p \vee \neg p$.

Now I move to $\mathbf{P}, i : p$ in the first subgame and to $\mathbf{P}, i : \neg p$ in the second. After a role switch in the second subgame, the final state is $\mathbf{P}, i : p \bigvee \mathbf{O}, i : p$, where I win regardless of the underlying model.

3.3 From games to proofs

Theorems 20 and 24 establish that winning strategies for *Me* in the provability game correspond to the validity of formulas. In this section, we extend this result to proof systems by introducing a sequent calculus, **DS**, where proofs correspond to *My*'s winning strategies in the provability game.

Labelled nominal formulas are either *labelled formulas* of the form $i : F$ or *relational atoms* of the form $R(i, j)$, where i and j are nominals and F is a **PNL** formula.[5] *Labelled sequents* have the form $\Gamma \Rightarrow \Delta$, where Γ, Δ are multisets containing labelled nominal formulas.

Starting with sequents, every provability state of the form

$$\mathbf{O}, i_1 : F_1 \bigvee \ldots \bigvee \mathbf{O}, i_n : F_n \bigvee \mathbf{P}, j_1 : G_1 \bigvee \ldots \bigvee \mathbf{P}, j_m : G_m$$

can be rewritten as the labelled sequent $\Gamma \Rightarrow \Delta$ where $\Gamma = \{i_1 : F_1, \ldots, i_n : F_n\}$ and $\Delta = \{j_1 : G_i, \ldots, j_m : G_m\}$. In what follows, we will not distinguish between provability states and their corresponding labelled sequent. For example, the provability game state $\mathbf{O}, i : (\Diamond\Diamond p \vee \Diamond\Diamond p) \bigvee \mathbf{P}, i : \Diamond p$ will be identified with the sequent $i : (\Diamond\Diamond p \vee \Diamond\Diamond p) \Rightarrow i : \Diamond p$.

The inference rules must be tailored in such a way that *proofs* in the sequent system match exactly *My winning strategies* in the provability game. This means that the user of the proof system takes the role of *Me*, scheduling game states and choosing moves in **P**-states. Moreover, *provability* in the proof system should correspond to *validity* in the game. For that, it is crucial to establish the formal relationship between elementary game states and logical axioms.

▶ **Lemma 27** ([22]). *Let $\Gamma \Rightarrow \Delta$ be composed of elementary game states only.* I *win the provability game in $\Gamma \Rightarrow \Delta$ iff one of the following holds*[6]

- **i.** $R^-(i, i) \in \Gamma$ *or* $R^+(i, i) \in \Delta$ *for some* i*;*
- **ii.** $\{R^+(i, j), R^-(i, j)\} \subseteq \Gamma$ *for some* $i \neq j$*;*
- **iii.** $\Gamma \cap \Delta \neq \varnothing$*.*

Figure 6 presents the labelled sequent systems **DS** with the standard initial axiom and structural/propositional rules. The modal rules and the relational rules *sym* and $ref\pm$ coincides with the modal rules originally presented by Viganò in [45], adapted to multi-relational modal logics. It is routine to show that the rule *no* in Figure 6 correspond to the non-overlapping axiom $\forall i, j.\neg(R^+(i, j) \wedge R^-(i, j))$.

The following result immediately implies that the provability game **DG** is adequate with respect to the calculus **DS**.

▶ **Theorem 28** (Adequacy - sequent system [22]). I *have a winning strategy in the provability game* $\mathbf{DG}(\Gamma \Rightarrow \Delta)$ *iff* $\Gamma \Rightarrow \Delta$ *is provable in* **DS***.*

Let us write $\models_{\mathbf{PNL}} \Gamma \Rightarrow \Delta$ iff for every **PNL**-model there is some $i : F \in \Gamma$ such that $\mathbb{M}, \mathsf{g}(i) \not\models F$, or there is some $i : G \in \Delta$ such that $\mathbb{M}, \mathsf{g}(i) \models G$. We have the following consequence of Theorems 20, 24, and 28:

▶ **Corollary 29.** *Let Γ, Δ be multisets of labelled formulas. Then $\models_{\boldsymbol{PNL}} \Gamma \Rightarrow \Delta$ iff there is a proof of $\Gamma \Rightarrow \Delta$ in* **DS***. In particular, F is **PNL**-valid iff there is a proof of $\Rightarrow F$ in* **DS***.*

---

[5] Observe that here we are abusing the notation, identifying $k : R(i, j)$ with $R(i, j)$ – see Footnote 4.
[6] Since relations are symmetric, we will identify $R^{\pm}(i, j)$ with $R^{\pm}(j, i)$.

Axiom and Structural Rules

$$\frac{}{\Gamma, i : F_{el} \Rightarrow \Delta, i : F_{el}}\ init \qquad \frac{\Gamma, i : F, i : F \Rightarrow \Delta}{\Gamma, i : F \Rightarrow \Delta}\ (L_c) \qquad \frac{\Gamma \Rightarrow i : F, i : F, \Delta}{\Gamma \Rightarrow i : F, \Delta}\ (R_c)$$

Propositional Rules

$$\frac{\Gamma \Rightarrow i : F, \Delta}{\Gamma, i : \neg F \Rightarrow \Delta}\ (L_\neg) \qquad \frac{\Gamma, i : F \Rightarrow \Delta}{\Gamma \Rightarrow i : \neg F, \Delta}\ (R_\neg)$$

$$\frac{\Gamma, i : F \Rightarrow \Delta \quad \Gamma, i : G \Rightarrow \Delta}{\Gamma, i : F \vee G \Rightarrow \Delta}\ (L_\vee) \qquad \frac{\Gamma \Rightarrow i : F, \Delta}{\Gamma \Rightarrow i : F \vee G, \Delta}\ (R^1_\vee) \qquad \frac{\Gamma \Rightarrow i : G, \Delta}{\Gamma \Rightarrow i : F \vee G, \Delta}\ (R^2_\vee)$$

$$\frac{\Gamma, i : F \Rightarrow \Delta}{\Gamma, i : F \wedge G \Rightarrow \Delta}\ (L^1_\wedge) \qquad \frac{\Gamma, i : G \Rightarrow \Delta}{\Gamma, i : F \wedge G \Rightarrow \Delta}\ (L^2_\wedge) \qquad \frac{\Gamma \Rightarrow i : F, \Delta \quad \Gamma \Rightarrow i : G, \Delta}{\Gamma \Rightarrow i : F \wedge G, \Delta}\ (R_\wedge)$$

Modal Rules

$$\frac{\Gamma, R^{\pm}(i, j) \Rightarrow \Delta}{\Gamma, i : \Diamond^{\pm} F \Rightarrow \Delta}\ (L_{\Diamond^{\pm}})_1 \qquad \frac{\Gamma, j : F \Rightarrow \Delta}{\Gamma, i : \Diamond^{\pm} F \Rightarrow \Delta}\ (L_{\Diamond^{\pm}})_2$$

$$\frac{\Gamma \Rightarrow R^{\pm}(i, j), \Delta \quad \Gamma \Rightarrow j : F, \Delta}{\Gamma \Rightarrow i : \Diamond^{\pm} F, \Delta}\ (R_{\Diamond^{\pm}}) \qquad \frac{\Gamma, j : F \Rightarrow \Delta}{\Gamma, i : [A] F \Rightarrow \Delta}\ (L_{[A]}) \qquad \frac{\Gamma \Rightarrow j : F, \Delta}{\Gamma \Rightarrow i : [A] F, \Delta}\ (R_{[A]})$$

Relational Rules

$$\frac{\Gamma \Rightarrow \Delta, R^{\pm}(j, i)}{\Gamma \Rightarrow \Delta, R^{\pm}(i, j)}\ sym \qquad \frac{}{\Gamma \Rightarrow \Delta, R^{+}(i, i)}\ ref+$$

$$\frac{}{\Gamma, R^{-}(i, i) \Rightarrow \Delta}\ ref- \qquad \frac{\Gamma \Rightarrow \Delta, R^{+}(i, j) \quad \Gamma \Rightarrow \Delta, R^{-}(i, j)}{\Gamma \Rightarrow \Delta}\ no$$

Figure 6 The proof system **DS**. In the rule init, F_{el} denotes an elementary formula. In the rules $(L_{\Diamond^{\pm}})_1$, $(L_{\Diamond^{\pm}})_2$, and $(R_{[A]})$, the nominal j is fresh. The rule $R_{\Diamond}$ has the proviso that $i \neq j$.

Proving cut-admissibility of labelled systems can be cumbersome due to the presence of relational rules. In [30], a systematic procedure for transforming axioms into rules was presented, based on *focusing* and *polarities* [5]. This procedure not only allows for generalizing different approaches for transforming axioms into sequent rules present in the literature [39, 45, 31], but it also provides a uniform way of proving cut-admissibility for the resulting systems.

The cut-admissibility result for **DS** is a particular instance of the general result in [30].

▶ **Theorem 30** (**PNL**-cut). *The following cut rule is admissible in* **DS**

$$\frac{\Gamma \Rightarrow \Delta, i : F \quad i : F, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\ cut$$

As a consequence, **DS** is consistent, since the only rule that can be applied in an empty sequent is *no*, and it is routine to show that it does not trivialise derivations.

3.4 Discussion – part II

This work opens up several promising directions for future exploration.

It would be interesting to explore extensions of **PNL** that relax symmetry assumptions, enabling the representation of scenarios where an agent a can influence the opinion of agent b, but not vice versa. Another potential direction involves incorporating the concept of a "budget," as introduced in the game discussed in the first part of this paper, to model

situations where proponents and opponents operate under a limited amount of *political capital*. In such scenarios, adding or modifying relations (*i.e.*, making new friends, making enemies to reconcile, etc) could reduce this capital. Preferences on how to "expend" the political capital could be expressed through a combination of **PNL** with a suitable choice logic – a framework where preferences are explicitly definable at the object level. Semantic games for choice logics have been explored in [20], and the extension of game-induced choice logic (**GCL**) to a provability game and proof system was proposed in [21]. Exploring these dynamics within our framework offers a compelling direction for future research.

Another particularly interesting avenue is extending the semantic-provability-proof system approach to other logics characterised by Kripke semantics. For instance, it would be worthwhile to investigate games for logics that involve model-change modalities [44, 36] or dynamic modalities [42]. Initial progress in this direction was made in [22], where we showed how the global link-adding and local link-changing modalities from [35] (inspired by sabotage modal logic [6, 7, 43]) can be incorporated into our framework.

We are also interested in exploring the application of this framework to develop games for constructive and intuitionistic modal logics [17, 38, 39, 8]. The constructive logic **CK** stands out as a promising candidate due to its intuitive semantics and straightforward sequent system. The main challenge lies in adapting the classical approach presented here to an intuitionistic setting.

Finally, building on ideas from [4, 3], we aim to establish a correspondence between winning innocent strategies in games played on Hyland-Ong arenas [26] and proofs in these constructive logics. This correspondence would deepen the connection between game semantics and constructive modal reasoning, opening new avenues for further study.

References

1 Samson Abramsky and Paul-André Melliès. Concurrent games and full completeness. In *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999*, pages 431–442. IEEE Computer Society, 1999. doi:10.1109/LICS.1999.782638.

2 Beniamino Accattoli, Stéphane Graham-Lengrand, and Delia Kesner. Tight typings and split bounds, fully developed. *J. Funct. Program.*, 30:e14, 2020. doi:10.1017/S095679682000012X.

3 Matteo Acclavio and Davide Catta. Lorenzen-style strategies as proof-search strategies. In Vadim Malvone and Aniello Murano, editors, *EUMAS 2023*, volume 14282 of *LNCS*, pages 150–166. Springer, 2023. doi:10.1007/978-3-031-43264-4_10.

4 Matteo Acclavio, Davide Catta, and Lutz Straßburger. Game semantics for constructive modal logic. In Anupam Das and Sara Negri, editors, *TABLEAUX*, volume 12842 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2021. doi:10.1007/978-3-030-86059-2_25.

5 Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. Log. Comput.*, 2(3):297–347, 1992. doi:10.1093/LOGCOM/2.3.297.

6 Carlos Areces, Raul Fervari, and Guillaume Hoffmann. Relation-changing modal operators. *Log. J. IGPL*, 23(4):601–627, 2015. doi:10.1093/JIGPAL/JZV020.

7 Guillaume Aucher, Johan van Benthem, and Davide Grossi. Modal logics of sabotage revisited. *J. Log. Comput.*, 28(2):269–303, 2018. doi:10.1093/LOGCOM/EXX034.

8 Gavin M. Bierman and Valeria de Paiva. On an intuitionistic modal logic. *Stud Logica*, 65(3):383–416, 2000. doi:10.1023/A:1005291931660.

9 Patrick Blackburn. Modal logic as dialogical logic. *Synthese*, 127:57–93, 2001. doi:10.1023/A:1010358017657.

10 Patrick Blackburn and Jerry Seligman. Hybrid languages. *J. Log. Lang. Inf.*, 4(3):251–272, 1995. doi:10.1007/BF01049415.

11 Torben Braüner. *Hybrid logic and its proof-theory*, volume 37. Springer Science & Business Media, 2010. doi:10.1007/978-94-007-0002-4.

12 Torben Braüner and Valeria de Paiva. Intuitionistic hybrid logic. *J. Appl. Log.*, 4(3):231–255, 2006. doi:10.1016/J.JAL.2005.06.009.

13 Simon Castellan, Pierre Clairambault, Silvain Rideau, and Glynn Winskel. Games and strategies as event structures. *Log. Methods Comput. Sci.*, 13(3), 2017. doi:10.23638/LMCS-13(3:35)2017.

14 Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. The structure of exponentials: Uncovering the dynamics of linear logic proofs. In Georg Gottlob, Alexander Leitsch, and Daniele Mundici, editors, *KGC*, volume 713 of *Lecture Notes in Computer Science*, pages 159–171. Springer, 1993. doi:10.1007/BFB0022564.

15 Claudia Faggian and François Maurel. Ludics nets, a game model of concurrent interaction. In *LICS*, pages 376–385. IEEE Computer Society, 2005. doi:10.1109/LICS.2005.25.

16 Christian G. Fermüller and Timo Lang. Interpreting sequent calculi as client-server games. In Renate A. Schmidt and Cláudia Nalon, editors, *TABLEAUX*, volume 10501 of *LNCS*, pages 98–113. Springer, 2017. doi:10.1007/978-3-319-66902-1_6.

17 F. B. Fitch. Intuitionistic modal logic with quantifiers. *Portugaliae Mathematicae*, 7:113–118, 1948. doi:doi:10.2307/2269275.

18 Robert Freiman. Games for hybrid logic - from semantic games to analytic calculi. In Alexandra Silva, Renata Wassermann, and Ruy J. G. B. de Queiroz, editors, *WoLLIC 2021*, volume 13038 of *LNCS*, pages 133–149. Springer, 2021. doi:10.1007/978-3-030-88853-4_9.

19 Robert Freiman. *From Semantic Games to Analytic Calculi*. PhD thesis, Technische Universität Wien, 2024.

20 Robert Freiman and Michael Bernreiter. Truth and preferences - A game approach for qualitative choice logic. In Sarah A. Gaggl, Maria V. Martinez, and Magdalena Ortiz, editors, *JELIA*, volume 14281 of *LNCS*, pages 547–560. Springer, 2023. doi:10.1007/978-3-031-43619-2_37.

21 Robert Freiman and Michael Bernreiter. Validity in choice logics - A game-theoretic investigation. In Helle Hvid Hansen, Andre Scedrov, and Ruy J. G. B. de Queiroz, editors, *WoLLIC 2023*, volume 13923 of *LNCS*, pages 211–226. Springer, 2023. doi:10.1007/978-3-031-39784-4_13.

22 Robert Freiman, Carlos Olarte, Elaine Pimentel, and Christian Fermüller. Reasoning about group polarization: From semantic games to sequent systems. In Nikolaj Bjorner, Marijn Heule, and Andrei Voronkov, editors, *LPAR*, volume 100 of *EPiC Series in Computing*, pages 70–87. EasyChair, 2024. doi:10.29007/wptz.

23 Robert Freiman, Carlos Olarte, Elaine Pimentel, and Christian G. Fermüller. Reasoning about group polarization: From semantic games to sequent systems. Technical report and tool available at https://github.com/promueva/PNL-game.git, 2024.

24 Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987. doi:10.1016/0304-3975(87)90045-4.

25 Jaakko Hintikka. Logic, language-games and information, Kantian themes in the philosophy of logic. *Revue Philosophique de la France Et de l'Etranger*, 163:477–478, 1973.

26 J. M. E. Hyland and C.-H. Luke Ong. On full abstraction for PCF: i, ii, and III. *Inf. Comput.*, 163(2):285–408, 2000. doi:10.1006/INCO.2000.2917.

27 Timo Lang. *Games, modalities and analytic proofs in nonclassical logics*. PhD thesis, Technische Universität Wien, 2021. doi:10.34726/hss.2021.92047.

28 Timo Lang, Carlos Olarte, Elaine Pimentel, and Christian G. Fermüller. A game model for proofs with costs. In Serenella Cerrito and Andrei Popescu, editors, *TABLEAUX*, volume 11714 of *LNCS*, pages 241–258. Springer, 2019. doi:10.1007/978-3-030-29026-9_14.

29 Paul Lorenzen and Kuno Lorenz, editors. *Dialogische Logik*. Wissenschaftliche Buchgesellschaft, [Abt. Verl.], Darmstadt, 1978.

30 Sonia Marin, Dale Miller, Elaine Pimentel, and Marco Volpe. From axioms to synthetic inference rules via focusing. *Ann. Pure Appl. Log.*, 173(5):103091, 2022. doi:10.1016/j.apal.2022.103091.

31 Sara Negri. Proof analysis in modal logic. *J. Philosophical Logic*, 34(5-6):507–544, 2005. doi:10.1007/s10992-005-2267-3.

32 Vivek Nigam and Dale Miller. Algorithmic specifications in linear logic with subexponentials. In António Porto and Francisco Javier López-Fraguas, editors, *PPDP*, pages 129–140. ACM, 2009. doi:10.1145/1599410.1599427.

33 Vivek Nigam and Dale Miller. A framework for proof systems. *J. Autom. Reason.*, 45(2):157–188, 2010. doi:10.1007/S10817-010-9182-1.

34 Catarina Dutilh Novaes and Rohan French. Paradoxes and structural rules from a dialogue perspective. *Philosophical Issues*, 28:129–158, 2018. doi:10.1111/phis.12119.

35 Mina Young Pedersen, Sonja Smets, and Thomas Ågotnes. Modal logics and group polarization. *J. Log. Comput.*, 31(8):2240–2269, 2021. doi:10.1093/logcom/exab062.

36 Elise Perrotin and Fernando R. Velázquez-Quesada. A semantic approach to non-prioritized belief revision. *Log. J. IGPL*, 29(4):644–671, 2021. doi:10.1093/JIGPAL/JZZ045.

37 Elaine Pimentel, Carlos Olarte, and Vivek Nigam. Process-as-formula interpretation: A substructural multimodal view (invited talk). In Naoki Kobayashi, editor, *FSCD*, volume 195 of *LIPIcs*, pages 3:1–3:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.FSCD.2021.3.

38 Gordon D. Plotkin and Colin P. Stirling. A framework for intuitionistic modal logic. In J. Y. Halpern, editor, *1st Conference on Theoretical Aspects of Reasoning About Knowledge*. Morgan Kaufmann, 1986. doi:10.2307/2274559.

39 Alex K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, College of Science and Engineering, School of Informatics, University of Edinburgh, 1994.

40 Lutz Straßburger. MELL in the calculus of structures. *Theor. Comput. Sci.*, 309(1-3):213–285, 2003. doi:10.1016/S0304-3975(03)00240-8.

41 Lutz Straßburger and Alessio Guglielmi. A system of interaction and structure IV: the exponentials and decomposition. *ACM Trans. Comput. Log.*, 12(4):23:1–23:39, 2011. doi:10.1145/1970398.1970399.

42 Johan van Benthem, Sujata Ghosh, and Fenrong Liu. Modelling simultaneous games in dynamic logic. *Synth.*, 165(2):247–268, 2008. doi:10.1007/S11229-008-9390-Y.

43 Johan van Benthem, Lei Li, Chenwei Shi, and Haoxuan Yin. Hybrid sabotage modal logic. *J. Log. Comput.*, 33(6):1216–1242, 2023. doi:10.1093/LOGCOM/EXAC006.

44 Fernando R. Velázquez-Quesada. Reliability-based preference dynamics: lexicographic upgrade. *J. Log. Comput.*, 27(8):2341–2381, 2017. doi:10.1093/LOGCOM/EXX019.

45 Luca Viganò. *Labelled Non-Classical Logics*. Kluwer Academic Publishers, 2000.

46 Bruno Xavier, Carlos Olarte, and Elaine Pimentel. A linear logic framework for multimodal logics. *Math. Struct. Comput. Sci.*, 32(9):1176–1204, 2022. doi:10.1017/S0960129522000366.

47 Zuojun Xiong and Thomas Ågotnes. On the logic of balance in social networks. *J. Log. Lang. Inf.*, 29(1):53–75, 2020. doi:10.1007/S10849-019-09297-0.

Modal Automata: Analysing Modal Fixpoint Logics, One Step at a Time

Yde Venema
Institute for Logic, Language and Computation, Universiteit van Amsterdam, The Netherlands

Abstract

We present and investigate a general framework for studying modal fixpoint logics and some related versions of monadic second-order logic, by means of certain finite automata that operate on Kripke structures. Characteristic of these modal automata is that the co-domain of their transition function is a set of formulas of a so-called one-step logic. The motivation for taking this perspective is that if a logic is characterised by a class of modal automata, many of its properties are already determined at the level of the much simpler one-step logic.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases modal logic, parity automata, fixpoint logic, one-step logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.5

Category Invited Talk

1 Modal automata

There is a long tradition in theoretical computer science connecting the research fields of automata theory and logic. This link becomes particularly strong when automata are used to classify (possibly) infinite objects like streams, trees or graphs. Interestingly, this research area has provided not only fundamental theoretical results, such as Rabin's decidability theorem, but also quite concrete applications in computer science, such as tools for the automatic verification of reactive systems.

Building on this tradition, and in particular on the work by Janin & Walukiewicz [8, 9] and D'Agostino & Hollenberg [3] on the modal μ-calculus, we present and investigate a general framework for studying modal fixpoint logics, and some versions of monadic second-order logic, by means of certain finite automata that operate on Kripke structures.

▶ **Definition 1.** *Let $\mathtt{Q}$ be a set of proposition letters. A (Kripke) structure over $\mathtt{Q}$ is a triple $\mathbb{S} = (S, R, \kappa)$ where S is a set of objects called points or worlds, $R \subseteq S \times S$ is a binary relation and κ is a $\mathtt{Q}$-marking or colouring on S, that is, a map $\kappa : S \to \wp(\mathtt{Q})$. Given a state s, the set $\{t \in S \mid (s,t) \in R\}$ of its successors is denoted as $R[s]$. A pointed structure is a pair $(\mathbb{S}, s)$ where s is a point in $\mathbb{S}$, and a query is a class of pointed structures.*

Characteristic of the modal automata that we are about to introduce is that the co-domain of their transition function is a set of formulas in a simple formalism that we call a one-step logic. The concept of a one-step logic was developed by several authors, including Cîrstea, Pattinson and Schröder, in the setting of coalgebraic modal logic.

▶ **Definition 2.** *A one-step model over a set A of monadic predicates is a pair (D, m) where $m : D \to \wp(A)$ is a A-marking on D.*

A one-step logic is a pair $(\mathtt{L}, \Vdash_1)$ where $\mathtt{L}$ is a one-step language, that is, a map $\mathtt{L}$ assigning to any set A a collection $\mathtt{L}(A)$ of objects called one-step formulas; and $\Vdash_1$ is a truth relation between one-step formulas and one-step models. If we have $(D, m) \Vdash_1 \phi$ we will say that ϕ is true of (D, m) or that (D, m) satisfies ϕ.

We will require every one-step formula ϕ to be monotone; that is: $(D, m) \Vdash_1 \phi$ implies $(D, m') \Vdash_1 \phi$ whenever m' is an extension of m (that is, $m(d) \subseteq m'(d)$ for every $d \in D$).

In addition to the monotonicity requirement, we impose some natural coherence conditions on one-step logics; for instance, we require that $\mathtt{L}(A) \subseteq \mathtt{L}(B)$ if $A \subseteq B$, that the truth relation is invariant under isomorphism, etc. We will usually blur the distinction between one-step logics and one-step languages, since the semantics of one-step formulas is generally fixed.

▶ **Example 3.** The one-step language $\mathtt{FOE}_1(A)$ of first-order logic with equality on a set of predicates A is given by the sentences (formulas without free variables) generated by the following grammar, where $a \in A$ and x, y are individual variables:

$$\phi ::= a(x) \mid x = y \mid x \neq y \mid \exists x.\phi \mid \forall x.\phi \mid \phi \vee \phi \mid \phi \wedge \phi \tag{1}$$

We use $\mathtt{FO}_1$ for the equality-free fragment of $\mathtt{FOE}_1$, where we omit the clauses $x = y$ and $x \neq y$, and we write $\mathtt{FOE}_1^\infty$ for the extension of $\mathtt{FOE}_1$ with the infinity quantifiers $\exists^\infty$ and $\forall^\infty$. The semantics of these languages is the obvious one.

▶ **Definition 4.** *Let* $\mathtt{L}$ *and* $\mathtt{Q}$ *be, respectively, a one-step language and a set of proposition letters. An* $\mathtt{L}$*-automaton over* $\mathtt{Q}$ *is a quadruple* $\mathbb{A} = (A, \Theta, \Omega, a_I)$ *where* A *is a finite set of objects called states;* $\Theta : A \times \wp(\mathtt{Q}) \to \mathtt{L}(A)$ *is its transition map;* $\Omega : A \to \omega$ *is its priority function; and* a_I *is its initial state.*

We will use the term "modal automaton" as a generic name for $\mathtt{L}$-automata for some one-step logic $\mathtt{L}$. The operational semantics of these automata is given in terms of parity games.

▶ **Definition 5.** *Let* $\mathtt{L}$ *and* $\mathtt{Q}$ *be, respectively, a one-step language and a set of proposition letters. Furthermore, let* $\mathbb{S} = (S, R, \kappa)$ *and* $\mathbb{A} = (A, \Theta, \Omega, a_I)$ *be, respectively, a Kripke structure and an* $\mathtt{L}$*-automaton over* $\mathtt{Q}$*.*

The acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ *is given as the two-player game of which the set of positions, as well as their owners, admissible moves and priorities are given in the table below:*

Position	*Player*	*Admissible moves*	*Priority*
$(a, s) \in A \times S$	$\exists$	$\{m : S \to \wp(A) \mid R[s], m \Vdash_1 \Theta(a, \kappa(s))\}$	$\Omega(a)$
$m : S \to \wp(A)$	$\forall$	$\{(b, t) \mid b \in m(t)\}$	0

Explained in words, the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ proceeds in rounds, each round moving from one basic position $(a, s) \in A \times S$ to the next. At such a basic position, it is $\exists$'s task to turn the set $R(s)$ of successors of s into the domain of a one-step model for the formula $\Theta(a, \kappa(s)) \in \mathtt{L}(A)$. That is, she needs to come up with a marking $m : R[s] \to \wp(A)$ such that $(R[s], m) \Vdash_1 \Theta(a, \kappa(s))$ (and if she cannot find such a valuation, she looses immediately). One may think of the set $\{(b, t) \mid b \in m(t)\}$ as a collection of witnesses to her claim that, indeed, the one-step formula $\Theta(a, \kappa(s))$ is true of $(R[s], m)$. The round ends with $\forall$ picking one of these witnesses, which then becomes the basic position at the start of the next round. (Unless, of course, $\exists$ managed to satisfy the formula $\Theta(a, \kappa(s))$ with the empty set of witnesses, in which case $\forall$ gets stuck and looses immediately.)

If $\exists$ and $\forall$ play the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ and neither of them gets stuck, the resulting match π will take infinitely long. In this case we use the parity condition to assign a winner to π: we consider the set $\mathsf{Inf}(\pi)$ of states that occur infinitely often in π, and declare $\exists$ to be the winner of π if the maximal priority of the states in $\mathsf{Inf}(\pi)$ is even, and $\forall$ if this maximum priority is odd.

▶ **Definition 6.** *Let* $\mathbb{A} = (A, \Theta, \Omega, a_I)$ *be some modal automaton. In case the pair* (a_I, s) *is a winning position for* $\exists$ *in the game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$*, we say that* $\mathbb{A}$ *accepts the pointed structure* $(\mathbb{S}, s)$ *and we write* $\mathbb{S}, s \Vdash \mathbb{A}$*. A query is recognized by an automaton* $\mathbb{A}$ *if it contains exactly those pointed structures that are accepted by* $\mathbb{A}$*.*

We say that a logic is characterised by a class of modal automata if for every formula of the logic we can find an equivalent automaton n the class, and vice versa. Some well-known logics are of this kind.

▶ **Example 7.** Any one-step logic $\mathtt{L}$ naturally induces a fixpoint logic $\mu\mathtt{L}$ which is characterised by the class $\mathsf{Aut}(\mathtt{L})$. The modal μ-calculus $\mu\mathtt{ML}$ is characterised by the class $\mathsf{Aut}(\mathtt{FO}_1)$, and various fragments of $\mu\mathtt{ML}$, including the alternation-free fragment and computational tree logic (CTL), correspond to natural subclasses of $\mathsf{Aut}(\mathtt{FO}_1)$. The same applies to propositional dynamic logic (PDL), if we consider a multi-sorted version of $\mathtt{FO}_1$.

Furthermore, if we restrict attention to tree-based structures, monadic second-order logic is characterised by the class $\mathsf{Aut}(\mathtt{FO}_1)$, and weak monadic second-order logic is characterised by a natural fragment of $\mathsf{Aut}(\mathtt{FOE}_1^\infty)$.

2 Some results

The motivation for studying logics from the perspective of modal automata is that much of their sometimes complex behaviour is already determined at the far simpler one-step level. This means that, in order to establish some property of a class of modal automata – or of the logic it characterises – it may suffice to prove a similar result for the one-step logic(s) underlying the automata. Here are some examples, where $\mathtt{L}$ and $\mathtt{L}'$ represent arbitrary one-step logics.

Closure properties of recognisable queries If $\mathtt{L}$ is closed under taking, respectively, disjunctions, conjunctions and boolean duals, then the class of $\mathsf{Aut}(\mathtt{L})$-recognizable queries is closed under taking union, intersection, and complementation.

Bisimulation invariance We say that a one-step formula ϕ is invariant under quotients if we have $(D, m) \Vdash_1 \phi$ iff $(D', m') \Vdash_1 \phi$, whenever (D', m') is a quotient of (D, m).

If $\mathtt{L}$ is the quotient-invariant fragment of $\mathtt{L}'$ (in some strong sense), then $\mathsf{Aut}(\mathtt{L})$ is the bisimulation-invariant fragment of $\mathsf{Aut}(\mathtt{L}')$. This result lies at the heart of the proof of the Janin-Walukiewicz Theorem, which identifies the modal μ-calculus as the bisimulation-invariant fragment of monadic second-order logic.

Nondeterminism Modal automata are generally alternating in nature, but there is a natural notion of nondeterminism as well: we call a modal automaton $\mathbb{A}$ nondeterministic if in any of its acceptance games, the role of $\forall$ is essentially reducible to that of a pathfinder. At the level of one-step logic, we introduce the notion of a disjunctive formula; roughly the idea is that if a disjunctive formula holds of some model then we can also satisfy it in a related model where the marking assigns to each element of the domain either the empty set or some singleton. It is easy to see that a modal automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$ is nondeterministic if the codomain of Θ consists of disjunctive formulas.

Disjunctive bases and simulation Assume that $\mathtt{L}' \subseteq \mathtt{L}$ and that $\mathtt{L}'$ consists of disjunctive formulas and is closed under disjunctions. If $\mathtt{L}$ and $\mathtt{L}'$ satisfy some natural distributive laws we call $\mathtt{L}$ a disjunctive basis for $\mathtt{L}$.

If $\mathtt{L}$ has a disjunctive basis $\mathtt{L}'$ then every (alternating) $\mathtt{L}$–automaton can be simulated by an equivalent (nondeterministic) $\mathtt{L}'$-automaton. Furthermore, the fixpoint logic characterised by $\mathtt{L}$ has several nice properties, such as the finite model property and uniform interpolation. This is how D'Agostino and Hollenberg established this property for the modal μ-calculus.

Coalgebraic generalisations

The concept of a modal automaton, and all applications in the theory of fixpoint logics and monadic second-order logics that we mentioned above, can be generalised to the setting of universal coalgebra. This means that the concept and the results become available for other coalgebraic modal (fixpoint) logics, such as graded, monotone, or probabilistic modal logic. A further result that we didn't mention above concerns the transfer of axiomatic completeness.

Disjunctive bases and axiomatic completeness Any one-step axiomatisation H for $\mathtt{L}$ naturally induces an axiom system $\mu\mathsf{H}$ for the μ-calculus $\mu\mathtt{L}$ induced by $\mathtt{L}$. If H is sound and complete for $\mathtt{L}$ and $\mathtt{L}$ has a disjunctive basis, then $\mu\mathsf{H}$ is sound and complete for the set of valid $\mu\mathtt{L}$-formulas. By this result, which generalises Walukiewicz' completeness result for the modal μ-calculus [12], one may for instance obtain complete axiomatisations for the graded and the monotone modal μ-calculus.

Here are some pertinent publications in which the theory of modal automata has been developed: [2, 1, 4, 5, 6, 7, 10, 11]

References

1. Facundo Carreiro, Alessandro Facchini, Yde Venema, and Fabio Zanasi. The power of the weak. *ACM Trans. Comput. Log.*, 21(2):15:1–15:47, 2020. doi:10.1145/3372392.
2. Facundo Carreiro and Yde Venema. PDL inside the μ-calculus: A syntactic and an automata-theoretic characterization. In Rajeev Goré, Barteld P. Kooi, and Agi Kurucz, editors, *Advances in Modal Logic 10, invited and contributed papers from the tenth conference on "Advances in Modal Logic," held in Groningen, The Netherlands, August 5-8, 2014*, pages 74–93. College Publications, 2014. URL: http://www.aiml.net/volumes/volume10/Carreiro-Venema.pdf.
3. Giovanna D'Agostino and Marco Hollenberg. Logical questions concerning the μ-calculus: Interpolation, lyndon and los-tarski. *J. Symb. Log.*, 65(1):310–332, 2000. doi:10.2307/2586539.
4. Sebastian Enqvist, Fatemeh Seifan, and Yde Venema. An expressive completeness theorem for coalgebraic modal mu-calculi. *Log. Methods Comput. Sci.*, 13(2), 2017. doi:10.23638/LMCS-13(2:14)2017.
5. Sebastian Enqvist, Fatemeh Seifan, and Yde Venema. Completeness for μ-calculi: A coalgebraic approach. *Ann. Pure Appl. Log.*, 170(5):578–641, 2019. doi:10.1016/J.APAL.2018.12.004.
6. Sebastian Enqvist and Yde Venema. Disjunctive bases: normal forms and model theory for modal logics. *Log. Methods Comput. Sci.*, 15(1), 2019. doi:10.23638/LMCS-15(1:30)2019.
7. Gaëlle Fontaine and Yde Venema. Some model theory for the modal μ-calculus: syntactic characterisations of semantic properties. *Log. Methods Comput. Sci.*, 14(1), 2018. doi:10.23638/LMCS-14(1:14)2018.
8. David Janin and Igor Walukiewicz. Automata for the modal μ-calculus and related results. In Jirí Wiedermann and Petr Hájek, editors, *Mathematical Foundations of Computer Science 1995, 20th International Symposium, MFCS'95, Prague, Czech Republic, August 28 - September 1, 1995, Proceedings*, volume 969 of *Lecture Notes in Computer Science*, pages 552–562. Springer, 1995. doi:10.1007/3-540-60246-1_160.
9. David Janin and Igor Walukiewicz. On the expressive completeness of the propositional μ-calculus with respect to monadic second order logic. In Ugo Montanari and Vladimiro Sassone, editors, *CONCUR '96, Concurrency Theory, 7th International Conference, Pisa, Italy, August 26-29, 1996, Proceedings*, volume 1119 of *Lecture Notes in Computer Science*, pages 263–277. Springer, 1996. doi:10.1007/3-540-61604-7_60.
10. Johannes Marti, Fatemeh Seifan, and Yde Venema. Uniform interpolation for coalgebraic fixpoint logic. In Lawrence S. Moss and Pawel Sobocinski, editors, *6th Conference on Algebra and Coalgebra in Computer Science, CALCO 2015, June 24-26, 2015, Nijmegen, The*

Netherlands, volume 35 of *LIPIcs*, pages 238–252. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPICS.CALCO.2015.238.

11 Yde Venema. Expressiveness modulo bisimilarity: A coalgebraic perspective. In Alexandru Baltag and Sonja Smets, editors, *Johan van Benthem on Logic and Information Dynamics*, pages 33–65. Springer, 2014. doi:10.1007/978-3-319-06025-5_2.

12 Igor Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. *Inf. Comput.*, 157(1-2):142–182, 2000. doi:10.1006/INCO.1999.2836.

Equi-Rank Homomorphism Preservation Theorem on Finite Structures

Benjamin Rossman
Duke University, Durham, NC, USA

Abstract

The Homomorphism Preservation Theorem (HPT) of classical model theory states that a first-order sentence is preserved under homomorphisms if, and only if, it is equivalent to an existential-positive sentence. This theorem remains valid when restricted to finite structures, as demonstrated by the author in [33, 34] via distinct model-theoretic and circuit-complexity based proofs. In this paper, we present a third (and significantly simpler) proof of the finitary HPT based on a generalized Cai-Fürer-Immerman construction. This method establishes a tight correspondence between syntactic parameters of a homomorphism-preserved sentence (quantifier rank, variable width, alternation height) and structural parameters of its minimal models (tree-width, tree-depth, decomposition height). Consequently, we prove a conjectured "equi-rank" version of the finitary HPT. In contrast, previous versions of the finitary HPT possess additional properties, but incur blow-ups in the quantifier rank of the equivalent existential-positive sentence.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases finite model theory, preservation theorems, quantifier rank

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.6

Acknowledgements I thank Deepanshu Kush, William He and Ken-ichi Kawarabayashi for stimulating conversions that prompted me to revisit the topic of this paper. Anuj Dawar provided helpful comments on a preliminary draft of this paper. I am grateful to the anonymous referees for numerous suggestions that improved the clarity of this paper. This paper was partially written during a visit to the National Institute of Informations in Tokyo, Japan.

1 Introduction

A first-order sentence φ is said to be *preserved under homomorphisms* if, for any model $\mathfrak{A} \models \varphi$ and any structure $\mathfrak{B}$ such that there exists a homomorphism $\mathfrak{A} \to \mathfrak{B}$, it holds that $\mathfrak{B} \models \varphi$. One class of first-order sentences that are always preserved under homomorphisms are the *existential-positive sentences*, which are built from atomic formulas (of the form $x_1 = x_2$ and $R(x_1, \ldots, x_r)$ where R is an r-ary relation symbol) via conjunction $\varphi_1 \wedge \varphi_2$, disjunction $\varphi_1 \vee \varphi_2$, and existential quantification $\exists x\, \varphi$ (that is, without negation $\neg\varphi$ or universal quantification $\forall x\, \varphi$).[1]

The Homomorphism Preservation Theorem (HPT) of classical model theory, attributed to Łoś, Lyndon and Tarski [30, 39, 31], states that existential-positive sentences are – up to logical equivalence – the only first-order sentences that are preserved under homomorphisms.[2]

▶ **Theorem 1.1** (HPT). *A first-order sentence is preserved under homomorphisms if, and only if, it is equivalent to an existential-positive sentence.*

[1] A *sentence* is a formula with no free variables. Although the definition of *preserved under homomorphisms* extends to formulas with free variables, we speak of sentences for simplicity sake. We further restrict attention to *relational languages* (without functions or constant symbols), even though most definitions and results in this paper extend to general first-order languages.

[2] Here the semantic notions of *logical equivalence* and *preserved under homomorphisms* are with respect to all (finite or infinite) structures.

The closely related Łoś-Tarski and Lyndon Preservation Theorems state that a first-order sentence that is preserved under embeddings (respectively: surjective homomorphisms) if, and only if, it is equivalent to an existential sentence (respectively: positive sentence). In each of these classical preservation theorems, the "if" direction follows directly from the semantics of first-order logic, while the "only if" direction was originally proved non-constructively using the Compactness Theorem.

A line of work in finite model theory initiated by Gurevich [20] studies the question of which theorems of classical model theory remain valid, and which become false, when restricted to finite structures. For example, the Compactness Theorem is easily seen to be false on finite structures. Counterexamples in [38, 20] witness the failure on finite structure of the Łoś-Tarski and Lyndon Theorems (on preservation under embeddings and surjective homomorphisms). In contrast, previous work of the author [33, Theorem 1.7] and [34, Theorem 6] showed that the classical HPT remains valid when restricted to finite structures.

▶ **Theorem 1.2** (HPT on finite structures)**.** *Every first-order sentence that is preserved under homomorphisms* ***on finite structures*** *is equivalent* ***on finite structures*** *to an existential-positive sentence.*

Articles [33, 34] provide two entirely different proofs of Theorem 1.2, which we discuss in §2.1 and §2.2. Unfortunately, both proofs incur large blow-ups from the quantifier rank r of a homomorphism-preserved first-order sentence φ to the quantifier rank $r^{\exists+}$ ($\gg r$) of the equivalent existential-positive sentence $\varphi^{\exists+}$. The blow-up in [33] is non-elementary: $r^{\exists+}$ is a tower-of-exponentials of height r. (With respect to the *length* of φ and $\varphi^{\exists+}$, a non-elementary blow-up is necessary [33, Theorem 6.1].) The method of [34] improved the quantifier rank blow-up to merely polynomial: $r^{\exists+} = O(r^3 \log r)$.

At the same time, an additional result of [33, Theorem 1.6] showed the classical HPT (Theorem 1.1) requires *no blow-up at all* in quantifier rank:

▶ **Theorem 1.3** (Equi-rank HPT)**.** *Every first-order sentence that is preserved under homomorphisms (on all structures) is equivalent to an existential-positive sentence* ***with the same quantifier rank****.*

The proof of Theorem 1.3 avoids the non-constructive Compactness Theorem, using instead a method of $\exists^+$-saturated co-retracts (see the discussion in §2.1). It was left as an open question whether Theorem 1.3 remains valid on finite structures. The main result of the present paper answers this question in the affirmative, finally unifying the finitary and equi-rank versions of the HPT.

▶ **Theorem 1.4** (Equi-rank HPT on finite structures)**.** *Every first-order sentence that is preserved under homomorphisms on finite structures is equivalent on finite structures to an existential-positive sentence with the same quantifier rank, variable width, and alternation height.*

Theorem 1.4 is simultaneously tight with respect to three different parameters: quantifier rank, variable width, and alternation height (see §3.2 for definitions). The surprisingly simple proof utilizes a generalized Cai-Fürer-Immerman construction [6] on finite relational structures (see §2.3). In particular, we consider two CFI structures $\mathfrak{C}_{\text{even}}$ and $\mathfrak{C}_{\text{odd}}$ over a finite <u>core</u> $\mathfrak{C}$ (i.e., structure such that every homomorphism $\mathfrak{C} \to \mathfrak{C}$ is an isomorphism). We show that

$$\mathfrak{C}_{\text{even}} \rightleftarrows \mathfrak{C} \quad \text{and} \quad \mathfrak{C}_{\text{odd}} \to \mathfrak{C} \not\to \mathfrak{C}_{\text{odd}}$$

where $\to$ denotes the existence of a homomorphism (Lemma 5.5). Theorem 1.4 then follows from a characterization of existential-positive definability in terms of the minimal cores of a homomorphism-closed class of finite structures (Lemma 4.6).

Related work

Our proof of Theorem 1.4 combines a few standard techniques that appear in many prior works. Variants of the Cai-Fürer-Immerman construction have found numerous applications similar in nature to our main result. In particular, Fürer [16] and Neuen [32] considered very similar generalization of CFI structures as given in Definition 5.4, only with different applications in mind.

The correspondence between quantifier rank/variable width, tree-width/tree-depth, and the # of moves/# of cops parameters of the cops-and-robber game, has been developed in several works including [1, 3, 12, 15, 19]. Variants of Lemmas 4.6 can be found in many of these papers.

Anuj Dawar (personal communication) independently identified the "core" homomorphism property of CFI structures (Lemma 5.5) in the context of a different problem. It is perhaps surprising that the application of the CFI construction to prove the Homomorphism Preservation Theorem remained unrecognized for such an extended period.

Versions of the HPT relativized to various classes of structures have been investigated in [4, 5, 10, 21] (see also [11], which corrects some claims in articles [5, 10]). Versions of the HPT for different fragments, as well as extensions, of first-order logic have been studied in [2, 7, 14].

2 Comparing the three proofs of the finitary HPT

In this section, we discuss both previous proofs of the Homomorphism Preservation Theorem on finite structures (Theorem 1.2) from articles [33, 34]. We then give a brief overview of our new proof using the Cai-Fürer-Immerman construction.

2.1 First proof via $\exists^+$-indistinguishable co-retracts

The original proof of the finitary HPT actually establishes the following stronger result.

▶ **Theorem 2.1** ([33, Theorem 5.15]). *For every finite relational signature σ and integer $r \geq 0$, there exists an integer $r^{\exists+}$ $(\geq r)$ and an operation*

$$\mathfrak{A} \longmapsto \widehat{\mathfrak{A}} : \{\ \sigma\text{-structures}\} \longrightarrow \{\ \sigma\text{-structures}\}$$

with the following properties:

- **(I)** *$\widehat{\mathfrak{A}}$ is a co-retract of $\mathfrak{A}$ (i.e., $\mathfrak{A}$ is a substructure of $\widehat{\mathfrak{A}}$ and there is a homomorphism $\widehat{\mathfrak{A}} \to \mathfrak{A}$ that fixes each element of $\mathfrak{A}$).*
- **(II)** *Whenever $\mathfrak{A}$ is finite, so is $\widehat{\mathfrak{A}}$.*
- **(III)** *Whenever $\mathfrak{A}$ and $\mathfrak{B}$ satisfy the same existential-positive sentences of quantifier rank $r^{\exists+}$, their co-retracts $\widehat{\mathfrak{A}}$ and $\widehat{\mathfrak{B}}$ satisfy the same first-order sentences of quantifier rank r.*

The finitary HPT follows straightforwardly from Theorem 2.1, although inheriting the same blow-up in quantifier rank from r to $r^{\exists+}$ (a tower-of-exponentials of height r). It is unknown whether every operation $\mathfrak{A} \longmapsto \widehat{\mathfrak{A}}$ satisfying properties (I), (II) and (III) requires a large blow-up from r to $r^{\exists+}$; the method of the present paper sheds no light on this question.

An additional result in [33, Theorem 4.11] shows that the optimal value $r^{\exists+} = r$ can be achieved by sacrificing property (II), that is, allowing $\widehat{\mathfrak{A}}$ to be infinite even when $\mathfrak{A}$ is finite. This yields the equi-rank version of the classical HPT (Theorem 1.3). In this version of the hat operation, $\widehat{\mathfrak{A}}$ is an (infinite) $\exists^+$-saturated co-retract of $\mathfrak{A}$. The manner of "finitizing" this operation in [33] is responsible for the tower-of-exponentials blow-up in Theorem 2.1.

▶ Remark 2.2. The method of $\exists^+$-saturated co-retracts was recently generalized by Abramsky and Reggio [2], though the lens of game comonads and arboreal categories. They give general conditions leading to equi-resource homomorphism preservation theorems, both with respect to fragments of first-order logic and relativized to classes of structures satisfying certain axioms. Our proof of Theorem 1.4 does not follow this template, but it would be interesting to know if a $\exists^+$-saturation proof is possible in the finite setting.

2.2 Second proof via AC^0 lower bounds

Subsequent work of the author [34] provides an entirely different proof of the finitary HPT, with a merely polynomial blow-up in quantifier rank, based on lower bounds in circuit complexity.

▶ **Theorem 2.3** (Main result of [34]). *Let $\mathcal{C}$ be a homomorphism-closed class of finite structures.*

(1) *If membership in $\mathcal{C}$ is decidable on structures of size n by non-uniform AC^0 formulas of size $O(n^r)$, then $\mathcal{C}$ is definable on finite structures (of all sizes) by a single existential-positive sentence of quantifier rank $O(r^3 \log r)$.*

(2) *If membership in $\mathcal{C}$ is decidable on structures of size n by non-uniform AC^0 circuits of size $O(n^s)$, then $\mathcal{C}$ is definable on finite structures (of all sizes) by a single existential-positive sentence of variable width $O(s \log s)$.*

The proof of Theorem 2.3 relies on three different lower bounds in circuit complexity [27, 28, 35], in addition to a result in graph minor theory on the excluded-minor approximation of tree-depth [9, 25].[3] We remark that Theorem 2.3 has an equivalent *descriptive complexity* formulation via the well-known correspondence [13, 24] between the non-uniform complexity class AC^0 and the logic $\mathsf{FO}[\mathsf{Arb}]$ (first-order logic with arbitrary background predicates).

▶ **Corollary 2.4.** *Let $\mathcal{C}$ be a homomorphism-closed class of finite structures.*

(1) *If $\mathcal{C}$ is definable on finite structures by an $\mathsf{FO}[\mathsf{Arb}]$ sentence of quantifier rank r, then $\mathcal{C}$ is definable on finite structures by an existential-positive sentence of quantifier rank $O(r^3 \log r)$.*

(2) *If $\mathcal{C}$ is definable on finite structures by an $\mathsf{FO}[\mathsf{Arb}]$ sentence of quantifier width s, then $\mathcal{C}$ is definable on finite structures by an existential-positive sentence of variable width $O(s \log s)$.*

Corollary 2.4 strengthens Theorem 1.2 by expanding the hypothesis from first-order sentences to the more expressive class of $\mathsf{FO}[\mathsf{Arb}]$ sentences, while the equivalent existential-positive sentences remain first-order (without background predicates). The results of the present paper do not directly improve Corollary 2.4, but do show that improvements would

[3] Part (1) of Theorem 2.3 is stated in [34] with a weaker polynomial bound $O(r^5 \log r)$. The improvement to $O(r^3 \log r)$ relies on a subsequent result of Czerwinski, Nadara, and Pilipczuk [9]. Part (2) of Theorem 2.3 is not explicitly stated in [34], but follows by a similar argument to part (1) using the AC^0 circuit lower bound of Li, Razborov and the author [28].

follow from strong enough lower bounds on the AC^0 complexity of distinguishing "even" and "odd" CFI structures over any base graph. Recent size-depth tradeoffs for AC^0-Frege refutations of Tseitin formulas [22, 17] might be relevant to this question.

2.3 New proof via the Cai-Fürer-Immerman construction

An influential article of Cai, Fürer and Immerman [6] introduced a construction of non-isomorphic simple graphs of order n that are indistinguishable by $o(n)$-variable counting logic (equivalently, by the $o(n)$-dimensional Weisfeiler-Leman algorithm). The general construction associates any 3-regular graph G with a pair of non-isomorphic graphs G_{even} and G_{odd}, which are hard to distinguish when the "base graph" G is an expander. The CFI construction is closely related to 3-XOR-CNF formulas associated with G, studied by Tseitin [40] in the setting of proof complexity. The Tseitin formulas are a system of linear equations modulo 2, with a variable X_e for each edge and a constraint for each vertex v (either $X_e \oplus X_f \oplus X_g = 0$ or $X_e \oplus X_f \oplus X_g = 1$, where e, f, g are the edges incident to v). This system is satisfiable if, and only if, the number of inhomogeneous constraints is even. The CFI graphs G_{even} and G_{odd} encode the two different (satisfiable and unsatisfiable) Tseitin formulas over G, corresponding to the 0-homomology classes of G over $\mathbb{Z}_2$.

The CFI construction generalizes to arbitrary (non-3-regular, non-connected) simple graphs G, as well as to abelian coordinate groups other than $\mathbb{Z}_2$. Generalizations of Tseitin formulas and the CFI construction have found numerous applications in finite model theory and proof complexity. In this paper we consider a natural version of the CFI construction on finite structures with a fixed relational signature (Definition 5.4). For any finite "base" structure $\mathfrak{A}$, this construction produces a pair of non-isomorphic finite structures $\mathfrak{A}_{\text{even}}$ and $\mathfrak{A}_{\text{odd}}$. Like the original CFI graphs, these structures are indistinguishable by first-order sentences whose quantifier rank / number of variables is less than the tree-depth / tree-width of $\mathfrak{A}$ (Lemma 5.6).

Unlike some versions of the CFI construction (such as the original CFI graphs [6]), structures $\mathfrak{A}_{\text{even}}$ and $\mathfrak{A}_{\text{odd}}$ project homomorphically to the base structure $\mathfrak{A}$. In the key special case of a finite core $\mathfrak{C}$ (where every homomorphism $\mathfrak{C} \to \mathfrak{C}$ is an isomorphism), we show that $\mathfrak{C}$ admits a homomorphism to $\mathfrak{C}_{\text{even}}$ but not to $\mathfrak{C}_{\text{odd}}$ (Lemma 5.5). Our main result, the equi-rank finitary HPT (Theorem 1.4), then follows by essentially well-known arguments.

3 Preliminaries

This section includes all relevant definitions pertaining to structures, homomorphisms, and first-order logic. See [29, 24] for additional background on finite model theory.

3.1 Structures and homomorphisms

▶ **Definition 3.1.** A *(relational) signature* is a set σ of relation symbols, each with an associated positive integer "arity". A *σ-structure* $\mathfrak{A}$ consists a set A (called the *universe* of $\mathfrak{A}$) together with an *interpretation* $R^{\mathfrak{A}} \subseteq A^t$ for each t-ary relation symbol R in σ. A structure is *finite* if its signature and universe are both finite.

▶ **Definition 3.2** (Homomorphism and isomorphism)**.**

For structures $\mathfrak{A}, \mathfrak{B}$ with the same signature, a *homomorphism* $h : \mathfrak{A} \to \mathfrak{B}$ is a function from the universe of $\mathfrak{A}$ to the universe of $\mathfrak{B}$, which maps each tuple in each relation of $\mathfrak{A}$ to a tuple in the corresponding relation of $\mathfrak{B}$. An *isomorphism* is a homomorphism h which is a bijection and whose inverse h^{-1} is a homomorphism.

- Notation $\mathfrak{A} \to \mathfrak{B}$ expresses that there exists a homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$. We say that $\mathfrak{A}$ and $\mathfrak{B}$ are *homomorphically equivalent*, denoted $\mathfrak{A} \rightleftarrows \mathfrak{B}$, if $\mathfrak{A} \to \mathfrak{B}$ and $\mathfrak{B} \to \mathfrak{A}$.
- Notation $\mathfrak{A} \to *$ denotes the class of all (finite or infinite) structures $\mathfrak{B}$ such that $\mathfrak{A} \to \mathfrak{B}$.

▶ **Definition 3.3** (The core of a finite structure)**.**
- A structure $\mathfrak{C}$ is a *core* if every homomorphism $h : \mathfrak{C} \to \mathfrak{C}$ is an isomorphism.
- Every finite structure $\mathfrak{A}$ is homomorphically equivalent to a unique core (up to isomorphism), which we call "the" *core* of $\mathfrak{A}$ and denote by $\mathsf{Core}(\mathfrak{A})$. $\mathsf{Core}(\mathfrak{A})$ is isomorphic to an induced substructure of $\mathfrak{A}$, namely any minimal retract of $\mathfrak{A}$ [23].

▶ **Definition 3.4** (Gaifman graph)**.** The *Gaifman graph* of a structure $\mathfrak{A}$, denoted $\mathsf{Gaif}(\mathfrak{A})$, is the simple graph with vertex set $V(\mathsf{Gaif}(\mathfrak{A})) = A$ (the universe of $\mathfrak{A}$) and undirected edge set

$$E(\mathsf{Gaif}(\mathfrak{A})) = \left\{ \{v, w\} \in \binom{A}{2} : v, w \text{ occur together in any tuple of any relation of } \mathfrak{A} \right\}.$$

3.2 First-order logic

Definitions in this subsection are with respect to an arbitrary fixed relational signature (i.e., a finite set of relation symbols, each with an associated positive integer "arity").

▶ **Definition 3.5** (First-order formulas)**.** *Formulas* of first-order logic (denoted by φ, ψ, θ) are constructed from
- *atomic formulas* $x = y$ and $Rx_1 \dots x_t$ (for a t-ary relation symbol R) via
- *Boolean connectives* $\varphi \wedge \psi$ and $\varphi \vee \psi$ and $\neg\varphi$ and
- *quantifiers* $\exists x\, \varphi$ and $\forall x\, \varphi$.

(Here $x, y, x_1, \dots, x_t$ are arbitrary variable symbols.)

A formula is said to be:
- a *sentence* if it contains no free variables (i.e., if every occurrence of every variable symbol is bounded by a quantifier),
- *positive* if it contains no negations ($\neg$),
- *existential-positive* if it contains no negations ($\neg$) or universal quantifiers ($\forall$),
- *primitive-positive* if it contains no negations ($\neg$), universal quantifiers ($\forall$) or disjunction ($\vee$).

In other words, primitive-positive formulas are constructed from atomic formulas via existential quantification ($\exists$) and conjunction ($\wedge$) only; existential-positive formulas additionally allow disjunction ($\vee$).

▶ **Definition 3.6** (Quantifier rank and variable width)**.** Two important parameters of first-order formulas are:
- *quantifier rank*, defined as the maximum nesting depth of quantifiers, and
- *variable width*, defined as the maximum number of free variables in any subformula.

▶ **Definition 3.7** (Alternation height)**.** A first-order formula φ has alternation height 0 iff it is quantifier-free (i.e., a Boolean combination of atomic formulas). For $d \geq 1$, φ has alternation height at most d iff it is a Boolean combination of finitely many first-order formulas $\psi_1, \dots, \psi_m$, each of the form $\exists x_1 \dots \exists x_k\, \theta$ or $\forall x_1 \dots \forall x_k\, \theta$ some $k \geq 0$ and θ with alternation height at most $d - 1$.

▶ **Example 3.8.** To illustrate various tradeoffs in parameters, we present four different primitive-positive sentences, all which define the class $\vec{P}_9 \to *$ where $\vec{P}_9$ is the directed path graph of order 9.

(a) quantifier rank 5, variable width 3, alternation height 3

$$\exists x_0\,\exists x_8\,\exists x_4\left(\begin{array}{c}\exists x_2\left(\exists x_1\,(Ex_0x_1\wedge Ex_1x_2)\wedge\exists x_3\,(Ex_2x_3\wedge Ex_3x_4)\right)\\ \wedge\,\exists x_6\left(\exists x_5\,(Ex_4x_5\wedge Ex_5x_6)\wedge\exists x_7\,(Ex_6x_7\wedge Ex_7x_8)\right)\end{array}\right)$$

(b) quantifier rank 4 (the minimum possible), variable width 3, alternation height 4

$$\exists x_4\left(\begin{array}{c}\exists x_2\left(\exists x_1\,(\exists x_0\,Ex_0x_1\wedge Ex_1x_2)\wedge\exists x_3\,(Ex_2x_3\wedge Ex_3x_4)\right)\\ \wedge\,\exists x_6\left(\exists x_5\,(Ex_4x_5\wedge Ex_5x_6)\wedge\exists x_7\,(Ex_6x_7\wedge\exists x_8\,Ex_7x_8)\right)\end{array}\right)$$

(c) quantifier rank 9, variable width 2 (the minimum possible), alternation height 8

$$\exists x_0\,\exists x_1\,(Ex_0x_1\wedge\exists x_2\,(Ex_1x_2\wedge\exists x_3\,(Ex_2x_3\wedge\cdots\exists x_7\,(Ex_6x_7\wedge\exists x_8\,Ex_7x_8)\cdots)))$$

(d) quantifier rank 9, variable width 9, alternation height 1 (the minimum possible)

$$\exists x_0\,\exists x_1\,\exists x_2\cdots\exists x_7\,\exists x_8\,(Ex_0x_1\wedge(Ex_1x_2\wedge(Ex_2x_3\wedge\cdots(Ex_6x_7\wedge Ex_7x_8)\cdots)))$$

4 Characterization of $\exists^+$ definability via the cops-and-robber game

The main result of this section (Lemma 4.6) provides a useful characterization of the classes of structures that are definable by existential-positive sentences with a given quantifier rank r, variable width s, and alternation height d. This characterization uses the well-known cops-and-robber game, which is also a means of defining the graph parameters *tree-width* and *tree-depth*.

4.1 Cops-and-robber game

Let $G=(V,E)$, $E\subseteq\binom{V}{2}$, be a finite simple graph. *Tree-width* and *tree-depth*, denoted by $\mathbf{tw}(G)$ and $\mathbf{td}(G)$, are well-studied graphs parameters that are usually defined in terms of tree-like decompositions of G. Below, we present an alternative definition of these parameters in terms of a two-player pursuit-evasion game. This "cops-and-robbers" characterization of $\mathbf{tw}(G)$ and $\mathbf{td}(G)$ is better suited to our purposes in this paper.

▶ **Definition 4.1** (Cops-and-robber game)**.** For any $d\geq 0$, the *height-d cops-and-robber game* on a graph G is a pursuit-evasion between two players: a team of *cops* and a sole *robber* (each with complete knowledge of the other's moves). The game is played in a sequence of d rounds as follows:

- Initially, the robber positions himself on any vertex of his choice.
- In round 1 of the game, any number of cops take up positions on their choice of vertices. The robber then moves to any vertex in the same connected component (bypassing any newly positioned cops that stand in the way). The game ends immediately only if the robber moves to a position guarded by a cop.
- In round 2 of the game, any subset of the assigned cops remain on their stationed vertices, and any number of (reassigned or additional) cops take up new positions on their choice of vertices. The robber then moves to any vertex in that reachable via a path that avoids the stationary cops (but may bypass any newly (re)positioned cops).
- The game proceeds in this manner for up to d rounds, ending immediately if the robber ever occupies the same vertex as a cop at the end of a round. This situation is a win for the cop team; otherwise the robber wins if not caught after d rounds.

The team of cops clearly have a winning strategy for any $d \geq 1$: in the first round, simply occupy all vertices in the connected component of the robber. The two questions that concern us are:

- *How many distinct cops are required to catch the robber?*
- *How many distinct cop "moves" (i.e., instances of positioning a cop on a vertex) are required to catch the robber?*

When the "height" of the game (i.e., number of rounds) is unbounded, the answers to these questions respectively characterize the *tree-width* and *tree-depth* of G [37, 18].

Taking into account the height d, we get two hierarchies of parameters

$$\mathbf{tw}_1(G) \geq \mathbf{tw}_2(G) \geq \cdots \quad \text{and} \quad \mathbf{td}_1(G) \geq \mathbf{td}_2(G) \geq \cdots$$

defined as follows:

- $\mathbf{tw}_d(G)$ is the maximum $s \geq 0$ such that the **robber** has a winning strategy in the height-d ∞-move s-cops-and-robber game on G.
- $\mathbf{td}_d(G)$ is the minimum $r \geq 1$ such that the **cops** have a winning strategy in height-d r-move ∞-cops-and-robber game.

Observe that

$$\mathbf{tw}_1(G) + 1 = \mathbf{td}_1(G) = \text{maximum } \# \text{ of vertices in a connected component of } G.$$

Also note that $\mathbf{tw}_d(G) + 1 \leq \mathbf{td}_d(G)$ for all d. Finally, note that $\mathbf{tw}_{|V(G)|}(G) = \mathbf{tw}(G)$ and $\mathbf{td}_{|V(G)|}(G) = \mathbf{td}(G)$.

▶ **Example 4.2.** With respect to the path graph P_k of order k, it is well-known that $\mathbf{tw}(P_k) = 2$ and $\mathbf{td}(P_k) = \log_2(k) + O(1)$. Moreover it is not hard to show that

$$\mathbf{tw}_d(P_k) = k^{1/d} + O(1) \quad \text{and} \quad \mathbf{td}_d(P_k) = (d + o(d))k^{1/d}$$

for all $d \leq \log_2(k)$.

In the remainder of this paper, we are not interested in $\mathbf{tw}_d(G)$ and $\mathbf{td}_d(G)$ *per se*, but rather in tradeoffs among all three parameters in the cops-and-robbers game: the numbers of cops, cop moves, and height. That is, for any triple of parameters r, s, d, which side has a winning strategy in the ***height-d r-move s-cops-and-robber game*** on G?

▶ Remark 4.3. Tradeoffs between r and s (when d is unbounded) were recently studied in [15], and monotonicity of an optimal cops strategy in the r-move s-cops-and-robber game was established in [3]. In both of those articles, the r-move s-cops-and-robber game is called the "r-round s-cops-and-robber game". Optimizing s with respect to fixed r characterizes a parameter called the *depth-r tree-width* of G.

In the present article, we use terminology "height-d" instead of "d-round" to avoid confusion with the previous terminology. We propose names *height-d tree-width* and *height-d tree-depth* for parameters $\mathbf{tw}_d(G)$ and $\mathbf{td}_d(G)$.

4.2 $\exists^+$ definability of homomorphism-closed classes

We shall now review a well-known characterization of the parameters (quantifier rank and variable width) required to define the class of structures $\mathfrak{C} \to *$ by a primitive-positive sentence, for finite core $\mathfrak{C}$. In fact, we slightly extend this characterization by including alternation height as a third parameter.

▶ **Lemma 4.4.** *For any finite core $\mathfrak{C}$ and integer $d, r, s \geq 0$, the following statements are equivalent:*

(i) *The class $\mathfrak{C} \to *$ is definable by a primitive-positive sentence with quantifier rank r, variable width s, and alternation height d.*

(ii) *The* **cops** *have a winning strategy in the height-d r-move s-cops-and-robber game on $\mathsf{Gaif}(\mathfrak{C})$.*

Results very similar to Lemma 4.4, which consider only one or two of the parameters d, r, s, have appeared before in the literature [1, 3, 12, 15, 18, 19, 37]. Lemma 4.4 may be used to characterize the parameters of existential-positive sentences that define a given homomorphism-closed class of finite structures.

▶ **Definition 4.5** (Minimal cores of a homomorphism-closed class of finite structures)**.**

- A class of finite structures $\mathcal{C}$ is *homomorphism-closed* if

$$(\mathfrak{A} \in \mathcal{C} \text{ and } \mathfrak{A} \to \mathfrak{B}) \implies \mathfrak{B} \in \mathcal{C}$$

  for all finite structures $\mathfrak{A}$ and $\mathfrak{B}$.
- A *minimal core* in $\mathcal{C}$ is a core $\mathfrak{C} \in \mathcal{C}$ such that

$$(\mathfrak{A} \in \mathcal{C} \text{ and } \mathfrak{A} \to \mathfrak{C}) \implies \mathfrak{A} \rightleftarrows \mathfrak{C}$$

  for all finite structures $\mathfrak{A}$.

Note that $\mathcal{C}$ is determined by its set of minimal cores: a finite structures $\mathfrak{A}$ belongs to $\mathcal{C}$ if, and only if, there is a homomorphism to $\mathfrak{A}$ from at least one minimal core in $\mathcal{C}$.

▶ **Lemma 4.6.** *Let $\mathcal{C}$ be a homomorphism-closed class of finite structures. For any $d, r, s \geq 0$, the following statements are equivalent:*

(i) *$\mathcal{C}$ is definable on finite structures by an existential-positive sentence with quantifier rank r, variable width s, and alternation height d.*

(ii) *The* **cops** *have a winning strategy in the height-d r-move s-cops-and-robber game on $\mathsf{Gaif}(\mathfrak{C})$, for every minimal core $\mathfrak{C}$ in $\mathcal{C}$.*

Lemma 4.6 follows from Lemma 4.4 by standard arguments (see [34, Proposition 2.16]). The only minor subtlety in the proof is a reliance on the fact that, for any given finite relational signature and $r \geq 0$, there are only a finite number of non-isomorphic cores with tree-depth r [23]. This is required so that the disjunction of primitive-positive sentences defining $\mathfrak{C} \to *$, over the non-isomorphic minimal cores $\mathfrak{C}$ in $\mathcal{C}$, constitutes a well-defined (finite length) existential-positive sentence.

5 Generalized Cai-Fürer-Immerman construction

In this section we present the generalized Cai-Fürer-Immerman construction discussed in §2.3 and establish its key properties given by Lemmas 5.5 and 5.6.

▶ **Notation 5.1.** Let $\mathbb{Z}_2$ denote the group $\{0, 1\}$ with addition modulo 2.

We will make use of the following basic lemma of graph homology, stated here over the coefficient group $\mathbb{Z}_2$.

▶ **Lemma 5.2.** *For any graph $G = (V, E)$, $E \subseteq \binom{V}{2}$, and function $\xi : V \to \mathbb{Z}_2$, the following statements are equivalent:*

(i) *$\sum_{u \in U} \xi(u) = 0$ for every connected component $U \subseteq V$.*

(ii) *ξ is a "1-boundary", that is, there exists a function $\varepsilon : E \to \mathbb{Z}_2$ such that for every $v \in V$, we have $\sum_{e \in E \,:\, v \in e} \varepsilon(e) = \xi(v)$.*

▶ **Notation 5.3.** Let $\mathfrak{A}$ be a structure with Gaifman graph $G = (A, E)$. For an element $v \in A$, let E_v, N_v and $N_v^\bullet$ respectively denote the incident-edge set, neighbor set and 1-neighborhood of v in G. That is,

$$E_v := \Big\{\{v, w\} : w \in A \text{ such that } \{v, w\} \text{ is an edge in } G\Big\},$$
$$N_v := \Big\{w \in A : \{v, w\} \text{ is an edge in } G\Big\},$$
$$N_v^\bullet := N_v \cup \{v\}.$$

▶ **Definition 5.4** (Generalized Cai-Fürer-Immerman structures $\mathfrak{A}_\xi$)**.** For any relational structure $\mathfrak{A}$ with universe A and any function $\xi : A \to \mathbb{Z}_2$, we define a structure $\mathfrak{A}_\xi$ (with the same signature) as follows:

- $\mathfrak{A}_\xi$ has universe

$$A_\xi := \Big\{\langle v, \alpha\rangle : v \in A \text{ and } \alpha : N_v^\bullet \to \mathbb{Z}_2 \text{ such that } \alpha(v) = 0 \text{ and } \sum_{w \in N_v} \alpha(w) = \xi(v)\Big\}.$$

  (Here for readability sake we use a distinctive notation $\langle v, \alpha\rangle$ for the ordered pair (v, α).)
- For each t-ary relation $R \subseteq A^t$ in $\mathfrak{A}$, the corresponding relation $R_\xi \subseteq A_\xi^t$ in $\mathfrak{A}_\xi$ is defined by

$$R_\xi := \left\{(\langle v_1, \alpha_1\rangle, \ldots, \langle v_t, \alpha_t\rangle) \in A_\xi^t : \begin{array}{l}(v_1, \ldots, v_t) \in R \text{ and} \\ \alpha_i(v_j) = \alpha_j(v_i) \text{ for all } i, j \in \{1, \ldots, t\}\end{array}\right\}.$$

Note that the projection $\langle v, \alpha\rangle \mapsto v$ is a homomorphism $\mathfrak{A}_\xi \to \mathfrak{A}$. Also note that this homomorphism need not be surjective (for instance, if $v \in A$ is an isolated vertex in $\mathsf{Gaif}(\mathfrak{A})$ and $\xi(v) = 1$).

Our first key lemma gives a necessary and sufficient condition for the existence of a homomorphism in the opposite direction in the special case that $\mathfrak{A}$ is a core.

▶ **Lemma 5.5.** *Let $\mathfrak{C}$ be a finite core with Gaifman graph $G = (C, E)$, and let $\xi : C \to \mathbb{Z}_2$. There exists a homomorphism $\mathfrak{C} \to \mathfrak{C}_\xi$ if, and only if, $\sum_{u \in U} \xi(u) = 0$ for each connected component U of G.*

Proof of Lemma 5.5. We first prove the "if" direction. Assume that $\sum_{u \in U} \xi(u) = 0$ for each connected component U of $\mathfrak{C}$. By Lemma 5.2, there exists a function $\varepsilon : E \to \mathbb{Z}_2$ such that for all $v \in C$, we have

$$\sum_{e \in E_v} \varepsilon(e) = \xi(v).$$

For each $v \in C$, define $\alpha_v : N_v \to \mathbb{Z}_2$ by $\alpha_v(v) := 0$ and $\alpha_v(w) := \varepsilon(\{v, w\})$ for all $w \in N_v \setminus \{v\}$. Note that $\langle v, \alpha_v\rangle \in C_\xi$. The function $h : v \mapsto \langle v, \alpha_v\rangle$ is the desired homomorphism $\mathfrak{C} \to \mathfrak{C}_\xi$.

We now prove the "only if" direction. Assume that h is an arbitrary homomorphism $\mathfrak{C} \to \mathfrak{C}_\xi$. Consider the projection homomorphism $\langle v, \alpha\rangle \mapsto v : \mathfrak{C}_\xi \to \mathfrak{C}$ that maps $\langle v, \alpha\rangle$ to v, and let f be the composition

$$f = (\langle v, \alpha\rangle \mapsto v) \circ h : \mathfrak{C} \to \mathfrak{C}.$$

Since $\mathfrak{C}$ is a core, f is an isomorphism. In particular, note that f restricts to a bijection from N_v to $N_{f(v)}$ for each $v \in C$.

For each $v \in C$, we have $h(v) = \langle f(v), \alpha_v \rangle$ for some $\alpha_v : N^{\bullet}_{f(v)} \to \mathbb{Z}_2$. Since $h(v) \in C_\xi$, we have

$$\alpha_v(f(v)) = 0 \quad \text{and} \quad \sum_{x \in N_{f(v)}} \alpha_v(x) = \xi(f(v)).$$

We now define $\widetilde{\alpha}_v : N^{\bullet}_v \to \mathbb{Z}_2$ by

$$\widetilde{\alpha}_v(w) := \alpha_v(f(w)).$$

Using the fact that f maps N_v bijectively to $N_{f(v)}$, we have

$$\widetilde{\alpha}_v(v) = \alpha_v(f(v)) = 0 \quad \text{and} \quad \sum_{w \in N_v} \widetilde{\alpha}_v(w) = \sum_{w \in N_v} \alpha_v(f(w)) = \sum_{x \in N_{f(v)}} \alpha_v(x) = \xi(f(v)).$$

Therefore, we have $\langle v, \widetilde{\alpha}_v \rangle \in C_{\widetilde{\xi}}$ where $\widetilde{\xi} : C \to \mathbb{Z}_2$ is the function $\widetilde{\xi}(v) := \xi(f(v))$.

Let us next consider the function $\widetilde{h} : C \to C_{\widetilde{\xi}}$ defined by

$$\widetilde{h}(v) := \langle v, \widetilde{\alpha}_v \rangle.$$

We claim that $\widetilde{h}$ is a homomorphism $\mathfrak{C} \to \mathfrak{C}_{\widetilde{\xi}}$. (We prove this claim only in order to show that $\widetilde{\alpha}_v(w) = \widetilde{\alpha}_w(v)$ for all $\{v, w\} \in E$.) To see why, consider any tuple $(v_1, \ldots, v_t) \in R$ in any relation of $\mathfrak{C}$. Since h is a homomorphism $\mathfrak{C} \to \mathfrak{C}_\xi$, we have

$$(h(v_1), \ldots, h(v_r)) = (\langle f(v_1), \alpha_{v_1} \rangle, \ldots, \langle f(v_t), \alpha_{v_t} \rangle) \in R_\xi.$$

By definition of R_ξ, for all $i, j \in \{1, \ldots, t\}$, we have $\alpha_{v_i}(f(v_j)) = \alpha_{v_j}(f(v_i))$ and hence

$$\widetilde{\alpha}_{v_i}(v_j) = \alpha_{v_i}(f(v_j)) = \alpha_{v_j}(f(v_i)) = \widetilde{\alpha}_{v_j}(v_i).$$

So we see that (by definition of $R_{\widetilde{\xi}}$)

$$(\widetilde{h}(v_1), \ldots, \widetilde{h}(v_t)) = (\langle v_1, \widetilde{\alpha}_{v_1} \rangle, \ldots, \langle v_t, \widetilde{\alpha}_{v_t} \rangle) \in R_{\widetilde{\xi}}.$$

This argument shows that $\widetilde{h}$ is a homomorphism $\mathfrak{C} \to \mathfrak{C}_{\widetilde{\xi}}$ as claimed.

We now define a function $\varepsilon : E \to \mathbb{Z}_2$ by

$$\varepsilon(\{v, w\}) := \widetilde{\alpha}_v(w).$$

This is well-defined, since (as established in previous paragraph) we have $\widetilde{\alpha}_v(w) = \widetilde{\alpha}_w(v)$ for all $\{v, w\} \in E$ (i.e., for all distinct $v, w \in C$ that appear together in any tuple of any relation of $\mathfrak{C}$).

For all $v \in C$, we have

$$\sum_{e \in E_v} \varepsilon(e) = \sum_{w \in N_v} \varepsilon(\{v, w\}) = \sum_{w \in N_v} \widetilde{\alpha}_v(w) = \xi(f(v)) = \widetilde{\xi}(v).$$

By Lemma 5.2, it follows that $\sum_{u \in U} \widetilde{\xi}(u) = 0$ for each connected component U of C. Since $\mathfrak{C}$ is a core, $f : \mathfrak{C} \to \mathfrak{C}$ restricts to a bijection on each connected component. We conclude that

$$\sum_{u \in U} \xi(u) = \sum_{u \in U} \xi(f(u)) = \sum_{u \in U} \widetilde{\xi}(u) = 0$$

as required. ◀

The second key lemma concerns the parameters of first-order sentences that distinguish any two structures in the class $\{\mathfrak{A}_\xi : \xi \text{ is a function from } A \text{ to } \mathbb{Z}_2\}$.

▶ **Lemma 5.6.** *Let $\mathfrak{A}$ be a finite structure with Gaifman graph $G = (A, E)$. Assume that the* **robber** *has a winning strategy starting on vertex $u \in A$ in the height-d r-move s-ccops-and-robber game on G. Further assume that $\xi, \zeta : A \to \mathbb{Z}_2$ and $\varepsilon : E \to \mathbb{Z}_2$ are functions satisfying*

$$\xi(v) + \zeta(v) + \sum_{e \in E_v} \varepsilon(e) = \mathbb{1}[\ v = u\ ] \text{ for all } v \in A.$$

Then structures $\mathfrak{A}_\xi$ and $\mathfrak{A}_\zeta$ are indistinguishable by first-order sentences with quantifier rank r, variable width s, and alternation height d.

We obtain Lemma 5.6 as the $k = 0$ case of the following more general lemma, whose statement is suited for proof by induction on the alternation height d.

▶ **Lemma 5.7.** *Let $\mathfrak{A}$ be a finite structure with Gaifman graph $G = (A, E)$. Assume that the* **robber** *has a winning strategy in the height-d r-move s-ccops-and-robber game on G with k ($\leq s$) cops initially positioned at vertices $v_1, \ldots, v_k \in A$ and the robber initially positioned at vertex $u \in A \setminus \{v_1, \ldots, v_k\}$. Further assume that $\xi, \zeta : A \to \mathbb{Z}_2$ and $\varepsilon : E \to \mathbb{Z}_2$ and $\alpha_i, \beta_i : N^\bullet_{v_i} \to \mathbb{Z}_2$ are functions satisfying*

$$\begin{aligned}
\langle v_1, \alpha_1\rangle, \ldots, \langle v_k, \alpha_k\rangle &\in A_\xi,\\
\langle v_1, \beta_1\rangle, \ldots, \langle v_k, \beta_k\rangle &\in A_\zeta,\\
\alpha_i(w) + \beta_i(w) + \varepsilon(\{v_i, w\}) = 0 &\text{ for all } i \in \{1, \ldots, k\} \text{ and } w \in N_{v_i},\\
\xi(v) + \zeta(v) + \sum_{e \in E_v} \varepsilon(e) = \mathbb{1}[\ v = u\ ] &\text{ for all } v \in A.
\end{aligned}$$

Then for every first-order formula $\varphi(x_1, \ldots, x_k)$ with quantifier rank r, variable width s, and alternation height d, we have

$$\mathfrak{A}_\xi \models \varphi(\langle v_1, \alpha_1\rangle, \ldots, \langle v_k, \alpha_k\rangle) \iff \mathfrak{A}_\zeta \models \varphi(\langle v_1, \beta_1\rangle, \ldots, \langle v_k, \beta_k\rangle).$$

Proof. We argue by induction on d. The base case $d = 0$ is equivalent to showing that $\mathfrak{A}_\xi$ and $\mathfrak{A}_\zeta$ satisfy the same quantifier-free formulas. Here it suffices to consider only the atomic formulas. That is, we must show

- $\langle v_i, \alpha_i\rangle = \langle v_j, \alpha_j\rangle \iff \langle v_i, \beta_i\rangle = \langle v_j, \beta_j\rangle$ for all indices $i, j \in \{1, \ldots, k\}$, and
- $(\langle v_{i_1}, \alpha_{i_1}\rangle, \ldots, \langle v_{i_t}, \alpha_{i_t}\rangle) \in R_\xi \iff (\langle v_{i_1}, \beta_{i_1}\rangle, \ldots, \langle v_{i_t}, \beta_{i_t}\rangle) \in R_\zeta$ for every t-ary relation symbol R and indices $i_1, \ldots, i_t \in \{1, \ldots, k\}$.

Both equivalences follow from our assumptions on $\xi, \zeta, \alpha_i, \beta_i, \varepsilon$. In particular, the second equivalence follows from the definition of relations R_ξ, R_ζ and the observation that

$$\alpha_i(v_j) = \alpha_j(v_i) \iff \alpha_i(v_j) + \varepsilon(\{v_i, v_j\}) = \alpha_j(v_i) + \varepsilon(\{v_i, v_j\}) \iff \beta_i(v_j) = \beta_j(v_i)$$

for all $i, j \in \{1, \ldots, k\}$.

For the induction step, assume that $d \geq 1$. By definition of having alternation height d, $\varphi(x_1, \ldots, x_k)$ is a Boolean combination of finitely many first-order formulas $\psi(x_{i_1}, \ldots, x_{i_j})$, each of the form

$$\exists y_1 \ldots \exists y_\ell\ \theta(x_{i_1}, \ldots, x_{i_j}, y_1, \ldots, y_\ell) \quad \text{or} \quad \forall y_1 \ldots \forall y_\ell\ \theta(x_{i_1}, \ldots, x_{i_j}, y_1, \ldots, y_\ell)$$

for some $j, \ell \geq 0$ and $1 \leq i_1 < \cdots < i_j \leq k$ and first-order formula θ with quantifier rank (at most) $r - \ell$, variable width (at most) s, and alternation depth (at most) $d - 1$. Consider any such formula $\psi(x_1, \ldots, x_j)$, without loss of generality of the form $\exists y_1 \ldots \exists y_\ell \, \theta(x_1, \ldots, x_j, y_1, \ldots, y_\ell)$ where $(i_1, \ldots, i_j) = (1, \ldots, j)$. It suffices to show that

$$\mathfrak{A}_\xi \models \psi(\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle) \iff \mathfrak{A}_\zeta \models \psi(\langle v_1, \beta_1 \rangle, \ldots, \langle v_j, \beta_j \rangle).$$

We will prove the implication $\Longrightarrow$; the reverse implication follows by a symmetric argument.

Assume that $\mathfrak{A}_\xi \models \psi(\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle)$ and fix a choice of $\langle \widehat{v}_1, \widehat{\alpha}_1 \rangle, \ldots, \langle \widehat{v}_\ell, \widehat{\alpha}_\ell \rangle \in A_\xi$ such that

$$\mathfrak{A}_\xi \models \theta(\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle, \langle \widehat{v}_1, \widehat{\alpha}_1 \rangle, \ldots, \langle \widehat{v}_\ell, \widehat{\alpha}_\ell \rangle).$$

In the remainder of this proof, we will show that there exist functions $\widehat{\beta}_i : N^\bullet_{\widehat{v}_\ell} \to \mathbb{Z}_2$ with $\langle \widehat{v}_i, \widehat{\beta}_i \rangle \in A_\zeta$ $(i \in \{1, \ldots, \ell\})$ such that

$$\mathfrak{A}_\zeta \models \theta(\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle, \langle \widehat{v}_1, \widehat{\beta}_1 \rangle, \ldots, \langle \widehat{v}_\ell, \widehat{\beta}_\ell \rangle).$$

It then follows that $\mathfrak{A}_\zeta \models \psi(\langle v_1, \beta_1 \rangle, \ldots, \langle v_j, \beta_j \rangle)$, which establishes the required implication

$$\mathfrak{A}_\xi \models \psi(\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle) \implies \mathfrak{A}_\zeta \models \psi(\langle v_1, \beta_1 \rangle, \ldots, \langle v_j, \beta_j \rangle).$$

In order to define suitable functions $\widehat{\beta}_i$, we invoke the robber's winning strategy in the height-d r-move s-ccops-and-robber game on G with cops starting at $v_1, \ldots, v_k$ and the robber starting at u. Suppose that in the round 1 of the game, the first j cops remain at $v_1, \ldots, v_j$ while the next ℓ cops redeploy to $\widehat{v}_1, \ldots, \widehat{v}_\ell$. There exists $\widehat{u} \in V \setminus \{v_1, \ldots, v_j, \widehat{v}_1, \ldots, \widehat{v}_\ell\}$ and a path $u = p_0, p_1, \ldots, p_m = \widehat{u}$ in G (with $m \geq 0$ and $\{p_{i-1}, p_i\} \in E$ for all $1 \leq i \leq m$) such that $\{v_1, \ldots, v_j\} \cap \{p_0, \ldots, p_m\} = \emptyset$ and the robber has a winning strategy in the $d-1$-round $r - \ell$-move s-ccops-and-robber game on G with cops starting at $v_1, \ldots, v_j, \widehat{v}_1, \ldots, \widehat{v}_\ell$ and the robber starting at $\widehat{u}$.

Define $\widehat{\varepsilon} : E \to \mathbb{Z}_2$ and $\widehat{\beta}_i : N^\bullet_{\widehat{v}_i} \to \mathbb{Z}_2$ $(i \in \{1, \ldots, \ell\})$ by

$$\widehat{\varepsilon}(e) := \varepsilon(e) + \sum_{i=1}^{m} \mathbb{1}[\, e = \{p_{i-1}, p_i\} \,],$$

$$\widehat{\beta}_i(w) := \begin{cases} 0 & \text{if } w = \widehat{v}_i, \\ \widehat{\alpha}_i(w) + \widehat{\varepsilon}(\{\widehat{v}_i, w\}) & \text{if } w \in N_{\widehat{v}_i}. \end{cases}$$

We will next show that $\xi, \zeta, \alpha_1, \ldots, \alpha_j, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_\ell, \beta_1, \ldots, \beta_j, \widehat{\beta}_1, \ldots, \widehat{\beta}_\ell$ and $\widehat{e}$ satisfy the conditions of the lemma with respect to $v_1, \ldots, v_j, \widehat{v}_1, \ldots, \widehat{v}_\ell, \widehat{u}$ and the first-order formula θ.

First, note that

$$\langle v_1, \alpha_1 \rangle, \ldots, \langle v_j, \alpha_j \rangle, \langle \widehat{v}_1, \widehat{\alpha}_1 \rangle, \ldots, \langle \widehat{v}_\ell, \widehat{\alpha}_\ell \rangle \in A_\xi.$$

Second, to establish that

$$\langle v_1, \beta_1 \rangle, \ldots, \langle v_j, \beta_j \rangle, \langle \widehat{v}_1, \widehat{\beta}_1 \rangle, \ldots, \langle \widehat{v}_\ell, \widehat{\beta}_\ell \rangle \in A_\zeta,$$

we observe that for each $i \in \{1, \dots, \ell\}$,

$$\begin{aligned}
\sum_{w \in N_{\widehat{v}_i}} \widehat{\beta}_i(w) &= \sum_{w \in N_{\widehat{v}_i}} \Big(\widehat{\alpha}_i(w) + \widehat{\varepsilon}(\{\widehat{v}_i, w\}) \Big) \\
&= \sum_{w \in N_{\widehat{v}_i}} \Big(\widehat{\alpha}_i(w) + \varepsilon(\{\widehat{v}_i, w\}) + \sum_{i=1}^{m} \mathbb{1}[\ \{\widehat{v}_i, w\} = \{p_{i-1}, p_i\}\ ] \Big) \\
&= \xi(\widehat{v}_i) + \sum_{e \in E\widehat{\ }_{v_i}} \varepsilon(e) + \sum_{w \in N_{\widehat{v}_i}} \sum_{i=1}^{m} \mathbb{1}[\ \{\widehat{v}_i, w\} = \{p_{i-1}, p_i\}\ ] \\
&= \zeta(\widehat{v}_i) + \mathbb{1}[\ \widehat{v}_i = u\ ] + \mathbb{1}[\ \widehat{v}_i = p_0\ ] + \mathbb{1}[\ \widehat{v}_i = \widehat{u}\ ] \\
&= \zeta(\widehat{v}_i) \quad (\text{since } p_0 = u \text{ and } \widehat{u} \notin \{v_1, \dots, v_j, \widehat{v}_1, \dots, \widehat{v}_\ell\}).
\end{aligned}$$

Third, by definition of $\widehat{\beta}_i$, we have

$$\widehat{\alpha}_i(w) + \widehat{\beta}_i(w) + \widehat{\varepsilon}(\{\widehat{v}_i, w\}) = 0 \text{ for all } i \in \{1, \dots, \ell\} \text{ and } w \in N_{\widehat{v}_i}.$$

Fourth and finally, for all $v \in A$, we have

$$\begin{aligned}
\xi(v) + \zeta(v) + \sum_{e \in E_v} \widehat{\varepsilon}(e) &= \xi(v) + \zeta(v) + \sum_{e \in E_v} \varepsilon(e) + \sum_{w \in N_v} \sum_{i=1}^{m} \mathbb{1}[\ \{v, w\} = \{p_{i-1}, p_i\}\ ] \\
&= \mathbb{1}[\ v = u\ ] + \mathbb{1}[\ v = p_0\ ] + \mathbb{1}[\ v = p_m\ ] \\
&= \mathbb{1}[\ v = \widetilde{u}\ ].
\end{aligned}$$

By the induction hypothesis applied to θ, we conclude that

$$\mathfrak{A}_\zeta \models \theta(\langle v_1, \alpha_1 \rangle, \dots, \langle v_j, \alpha_j \rangle, \langle \widehat{v}_1, \widehat{\beta}_1 \rangle, \dots, \langle \widehat{v}_\ell, \widehat{\beta}_\ell \rangle),$$

finishing the proof. ◀

6 Proof of the equi-rank finitary HPT

Proof of Theorem 1.4. Let φ be a first-order sentence that is preserved under homomorphisms on finite structures. Let r, s and d be the quantifier rank, variable width and alternation height of φ.

Assume that φ has at least one finite model, since otherwise the theorem is trivial (allowing $\bot$ as a special primitive-positive sentence with no models). Consider any minimal core $\mathfrak{C}$ (with universe C) in the class of finite models of φ. By Lemma 4.6, it suffices to show that the **cops** have a winning strategy in the height-d r-move s-cops-and-robber game on $\mathsf{Gaif}(\mathfrak{C})$, starting from an (adversarial) choice of initial position $u \in C$ for the robber.

Let $\mathfrak{C}_0$ be the CFI structure where 0 stands for the all-zero function $C \to \mathbb{Z}_2$, and let $\mathfrak{C}_{1_u}$ be the CFI structure where 1_u stands for the function $C \to \mathbb{Z}_2$ with value 1 at u and 0 elsewhere. Additionally, let ε be the all-zero function $E \to \mathbb{Z}_2$. Note that

$$0(v) + 1_u(v) + \sum_{e \in E_v} \varepsilon(e) = 0 + \mathbb{1}[\ v = u\ ] + \sum_{e \in E_v} 0 = \mathbb{1}[\ v = u\ ] \text{ for all } v \in A.$$

By Lemma 5.5, we have $\mathfrak{C} \to \mathfrak{C}_0$. Since φ is closed under homomorphisms on finite structures, it follows that $\mathfrak{C}_0 \models \varphi$. Lemma 5.5 also implies $\mathfrak{C} \not\to \mathfrak{C}_{1_u}$. Since $\mathfrak{C}_{1_u} \to \mathfrak{C}$, our assumption that $\mathfrak{C}$ is a minimal core in the class of finite models of φ implies that $\mathfrak{C}_{1_u} \not\models \varphi$.

We have established that φ is a first-order sentence with quantifier rank r, variable width s and alternation height d, which distinguishes the pair of structures $\mathfrak{C}_0$ and $\mathfrak{C}_{1_u}$. Therefore, by (the contrapositive of) Lemma 5.6, the **cops** have a winning strategy with the robber starting on $u \in C$ in the height-d r-move s-cops-and-robber game on $\mathsf{Gaif}(\mathfrak{C})$.

By Lemma 4.6, we conclude that φ is equivalent on finite structures to an existential-positive sentence with quantifier rank r, variable width s, and alternation height d, as required. ◀

7 Open questions

As discussed in §2, it remains an open question whether the quantifier-rank blow-up can be eliminated or significantly reduced in either Theorem 2.1 or 2.3 (the main results of [33, 34]).

Another interesting question is to investigate tradeoffs in Theorems 2.1 or 2.3 involving alternation depth d. There is a natural correspondence between *primitive-positive sentences* and *monotone* SAC^0 *circuits* (with unbounded $\bigvee$ gates and fan-in 2 $\wedge$ gates). For any finite graph G, this correspondence gives the following upper bounds on the COLORED G-SUBGRAPH ISOMORPHISM problem (equivalent to the G-HOMOMORPHISM problem when G is a core).

▶ **Proposition 7.1.** *For any finite graph G, the* COLORED G-SUBGRAPH ISOMORPHISM *problem, as a sequence of monotone Boolean functions $\{0,1\}^{|E(G)|\cdot n^2} \to \{0,1\}$, is computable for all $d \geq 1$ by both*

- *monotone* SAC^0 *formulas with $\bigvee$-depth d and size $n^{\mathbf{td}_d(G)+O(1)}$, and*
- *monotone* SAC^0 *circuits with $\bigvee$-depth d and size $n^{\mathbf{tw}_d(G)+O(1)}$.*

In the arithmetic setting, the corresponding set-multilinear polynomials are computable by monotone arithmetic SAC^0 *formulas and circuits with $\sum$-depth d and size $n^{\mathbf{td}_d(G)+O(1)}$ and $n^{\mathbf{tw}_d(G)+O(1)}$, respectively.*

It would be interesting to establish lower bounds in circuit complexity that nearly match the size-depth tradeoffs of Proposition 7.1. Recent work of the author [36] takes a step in this direction by establishing $n^{\Omega(\mathbf{td}_d(G))}$ and $n^{\Omega(\mathbf{tw}_d(G))}$ lower bounds in the case of path graphs $G = P_k$. With respect to monotone arithmetic circuits and formulas, even tighter $n^{\mathbf{td}_d(G)-O(1)}$ and $n^{\mathbf{tw}_d(G)-O(1)}$ lower bounds for general graphs G might be possible using the technique of Komarath, Pandey and Rahul [26].

References

1 Samson Abramsky, Anuj Dawar, and Pengming Wang. The pebbling comonad in finite model theory. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE, 2017. doi:10.1109/LICS.2017.8005129.

2 Samson Abramsky and Luca Reggio. Arboreal categories and equi-resource homomorphism preservation theorems. *Annals of Pure and Applied Logic*, 175(6):103423, 2024. doi:10.1016/J.APAL.2024.103423.

3 Isolde Adler and Eva Fluck. Monotonicity of the Cops and Robber Game for Bounded Depth Treewidth. In *49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024)*, volume 306, pages 6:1–6:18, 2024. doi:10.4230/LIPICS.MFCS.2024.6.

4 Albert Atserias. On digraph coloring problems and treewidth duality. *European Journal of Combinatorics*, 29(4):796–820, 2008. doi:10.1016/J.EJC.2007.11.004.

5 Albert Atserias, Anuj Dawar, and Phokion G. Kolaitis. On preservation under homomorphisms and unions of conjunctive queries. *J. ACM*, 53(2):208–237, 2006. doi:10.1145/1131342.1131344.

6 Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

7 James Carr. Homomorphism preservation theorems for many-valued structures. *arXiv preprint arXiv:2403.00217*, 2024.

8 Hubie Chen. On the complexity of existential positive queries. *ACM Transactions on Computational Logic (TOCL)*, 15(1):1–20, 2014. doi:10.1145/2559946.

9 Wojciech Czerwinski, Wojciech Nadara, and Marcin Pilipczuk. Improved bounds for the excluded-minor approximation of treedepth. In *27th Annual European Symposium on Algorithms (ESA 2019)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

10 Anuj Dawar. Homomorphism preservation on quasi-wide classes. *Journal of Computer and System Sciences*, 76(5):324–332, 2010. doi:10.1016/J.JCSS.2009.10.005.

11 Anuj Dawar and Ioannis Eleftheriadis. Preservation theorems on sparse classes revisited. In *Proceedings of the 49th International Symposium on Mathematical Foundations of Computer Science*, 2024.

12 Anuj Dawar, Tomáš Jakl, and Luca Reggio. Lovász-type theorems and game comonads. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13. IEEE, 2021.

13 Larry Denenberg, Yuri Gurevich, and Saharon Shelah. Definability by constant-depth polynomial-size circuits. *Information and control*, 70(2-3):216–240, 1986. doi:10.1016/S0019-9958(86)80006-7.

14 Tomás Feder and Moshe Y. Vardi. Homomorphism closed vs. existential positive. In *Proceedings of the 18th IEEE Symposium on Logic in Computer Science*, pages 310–320, 2003. doi:10.1109/LICS.2003.1210071.

15 Eva Fluck, Tim Seppelt, and Gian Luca Spitzer. Going deep and going wide: Counting logic and homomorphism indistinguishability over graphs of bounded treedepth and treewidth. In *32nd EACSL Annual Conference on Computer Science Logic (CSL 2024)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2024.

16 Martin Fürer. On the combinatorial power of the Weisfeiler-Lehman algorithm. In *International Conference on Algorithms and Complexity*, pages 260–271. Springer, 2017. doi:10.1007/978-3-319-57586-5_22.

17 Nicola Galesi, Dmitry Itsykson, Artur Riazanov, and Anastasia Sofronova. Bounded-depth Frege complexity of Tseitin formulas for all graphs. *Annals of Pure and Applied Logic*, 174(1):103166, 2023. doi:10.1016/J.APAL.2022.103166.

18 Archontia C Giannopoulou, Paul Hunter, and Dimitrios M Thilikos. LIFO-search: A min–max theorem and a searching game for cycle-rank and tree-depth. *Discrete Applied Mathematics*, 160(15):2089–2097, 2012. doi:10.1016/J.DAM.2012.03.015.

19 Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *Journal of the ACM*, 54(1):1–24, 2007. doi:10.1145/1206035.1206036.

20 Yuri Gurevich. Toward logic tailored for computational complexity. In M. M. Richter et al., editor, *Computation and Proof Theory*, pages 175–216. Springer Lecture Notes in Mathematics, 1984.

21 Lucy Ham. Relativised homomorphism preservation at the finite level. *Studia Logica*, 105(4):761–786, 2017. doi:10.1007/S11225-017-9710-7.

22 Johan Håstad. On small-depth Frege proofs for Tseitin for grids. *Journal of the ACM (JACM)*, 68(1):1–31, 2020. doi:10.1145/3425606.

23 Pavol Hell and Jaroslav Nešetřil. The core of a graph. *Discrete Math.*, 109:117–126, 1992. doi:10.1016/0012-365X(92)90282-K.

24 Neil Immerman. *Descriptive Complexity Theory*. Graduate Texts in Computer Science. Springer, New York, 1999.

25 Ken-ichi Kawarabayashi and Benjamin Rossman. A polynomial excluded-minor characterization of treedepth. In *29th ACM-SIAM Symposium on Discrete Algorithms*, pages 234–246, 2018.

26 Balagopal Komarath, Anurag Pandey, and Chengot Sankaramenon Rahul. Monotone arithmetic complexity of graph homomorphism polynomials. *Algorithmica*, 85(9):2554–2579, 2023. doi:10.1007/S00453-023-01108-0.

27 Deepanshu Kush and Benjamin Rossman. Tree-depth and the formula complexity of subgraph isomorphism. *SIAM Journal on Computing*, 52(1):273–325, 2023. doi:10.1137/20M1372925.

28 Yuan Li, Alexander Razborov, and Benjamin Rossman. On the AC^0 complexity of subgraph isomorphism. *SIAM Journal on Computing*, 46(3):936–971, 2017.

29 Leonid Libkin. *Elements of Finite Model Theory*. Springer-Verlag, 2004.

30 Jerzy Łoś. On the extending of models (I). *Fundamenta Mathematicae*, 42:38–54, 1955.

31 Roger C. Lyndon. Properties preserved under homomorphism. *Pacific J. Math.*, 9:129–142, 1959.

32 Daniel Neuen. Homomorphism-distinguishing closedness for graphs of bounded tree-width. In *41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024.

33 Benjamin Rossman. Homomorphism preservation theorems. *Journal of the ACM*, 55(3):15, 2008.

34 Benjamin Rossman. An improved homomorphism preservation theorem from lower bounds in circuit complexity. In *8th Innovations in Theoretical Computer Science*, volume 67 of *LIPIcs*, pages 27:1–17, 2017. doi:10.4230/LIPICS.ITCS.2017.27.

35 Benjamin Rossman. Formulas versus circuits for small distance connectivity. *SIAM Journal on Computing*, 47(5):1986–2028, 2018. doi:10.1137/15M1027310.

36 Benjamin Rossman. Formula size-depth tradeoffs for iterated sub-permutation matrix multiplication. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1386–1395, 2024. doi:10.1145/3618260.3649628.

37 Paul D Seymour and Robin Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22–33, 1993. doi:10.1006/JCTB.1993.1027.

38 William W. Tait. A counterexample to a conjecture of Scott and Suppes. *J. Symbolic Logic*, 24:15–16, 1959. doi:10.2307/2964569.

39 Alfred Tarski. Contributions to the theory of models. I. In *Indagationes Mathematicae (Proceedings)*, volume 57, pages 572–581. Elsevier BV, 1954.

40 Grigori S Tseitin. On the complexity of derivation in propositional calculus. *Automation of reasoning: 2: Classical papers on computational logic 1967–1970*, pages 466–483, 1983.

Extension Preservation on Dense Graph Classes

Ioannis Eleftheriadis
Department of Computer Science and Technology, University of Cambridge, UK

Abstract

Preservation theorems provide a direct correspondence between the syntactic structure of first-order sentences and the closure properties of their respective classes of models. A line of work has explored preservation theorems relativised to combinatorially tame classes of sparse structures [Atserias et al., JACM 2006; Atserias et al., SiCOMP 2008; Dawar, JCSS 2010; Dawar and Eleftheriadis, MFCS 2024]. In this article we initiate the study of preservation theorems for dense classes of graphs. In contrast to the sparse setting, we show that extension preservation fails on most natural dense classes of low complexity. Nonetheless, we isolate a technical condition which is sufficient for extension preservation to hold, providing a dense analogue to a result of [Atserias et al., SiCOMP 2008].

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases Extension preservation, finite model theory, dense graphs, cliquewidth

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.7

Funding The author is supported by a George and Marie Vergottis Scholarship awarded through Cambridge Trust, an Onassis Foundation Scholarship, and a Robert Sansom Studentship.

1 Introduction

The early days of finite model theory were considerably guided by attempts aiming to relativise theorems and techniques of classical model theory to the finite realm. While many of these were trivially shown to admit no meaningful relativisation, others were extended in a way that broadened their applicability and rendered them extremely useful tools in the study of finite models. Preservation theorems were at the heart of this approach. Most notably, the Łoś-Tarski preservation theorem which asserts that a first-order formula is preserved by extensions between all structures if and only if it is equivalent to an existential formula, was shown to fail in the finite from early on [28, 22]. On the contrary, the homomorphism preservation theorem asserting that a formula is preserved by homomorphisms if and only if it is existential-positive, was open for several years until it was surprisingly shown to extend to finite structures [27], leading to applications in constraint satisfaction problems and database theory.

Still, considering all finite structures allows for combinatorial complexity, giving rise to wildness from a model-theoretic perspective, and intractability from a computational perspective. Indeed, problems which are hard in general become tractable when restring to classes of finite structures which are, broadly-speaking, tame [12]. In the context of preservation theorems, restricting on a subclass weakens both the hypothesis and the conclusion, therefore leading to an entirely new question. A study of preservation properties for such restricted classes of finite structures was initiated in [4] and [3] for homomorphism and extension preservation respectively. This investigation led to the introduction of different notions of wideness, which allow for arguments based on the *locality* of first-order logic. However, as it was recently realised [14], these arguments require slightly restrictive closure assumptions which are not always naturally present. In particular, it was shown that homomorphism preservation holds over any hereditary *quasi-wide* class which is closed under *amalgamation over bottlenecks*.

Quasi-wideness is a Ramsey-theoretic condition which informally says that in any large enough structure in the class one can remove a bounded number of elements, called bottleneck points, so that there remains a large set of pairwise far-away elements. Here, the number of bottleneck points is allowed to dependent on the choice of distance. Hereditary quasi-wide classes were later identified with *nowhere dense* classes [24]. Over the years, a successful program was developed aiming to understand the combinatorial and model-theoretic features of nowhere dense classes, and exploit them for algorithmic purposes [25]. The culmination of this was the seminal result that first-order model checking is fixed-parameter tractable on nowhere dense classes [21].

In recent years, the focus has shifted towards extending this well understood theory to more general, possibly dense, well-behaved classes, which fall out of the classification provided by the sparsity program. In these efforts, the model-theoretic notions of *monadic stability* and *monadic dependence* have played central roles. Monadic stability, initially introduced by Baldwin and Shelah [5] in the context of classification of complete first-order theories, prohibits arbitrarily large definable orders in monadic expansions. In the language of first-order transductions, a class is monadically stable whenever it does not transduce the class of finite linear orders. More generally, a class is monadically dependent if it does not transduce the class of all graphs. In the context of monotone classes of graphs, Adler and Adler [1] first observed that the above notions coincide with nowhere density, a result which was also extended to arbitrary relational structures [7]. The generalisation of sparsity theory to dense classes eventually led to the result that first-order model checking is fixed-parameter tractable on all monadically stable graph classes [15], which in particular include transductions of nowhere dense classes. It is conjectured that the above result extends to all monadically dependent classes, while a converse was recently established for hereditary graph classes (under standard complexity-theoretic assumptions) [18].

The purpose of the present article is to initiate the investigation of preservation theorems on tame dense graph classes. Much like nowhere dense classes are equivalently characterised by quasi-wideness, monadically stable and monadically dependent graph classes also admit analogous wideness-type characterisations. In the case of monadic stability, the relevant condition is known as *flip-flatness* [17]; this may be viewed as a direct analogue of quasi-wideness which replaces the vertex deletion operation by flips, i.e. edge-complementations between subsets of the vertex set. For monadically dependent classes the relevant condition, known as *flip-breakability* [18], allows to find two large sets such that elements in one are far away from elements in the other, again after performing a bounded number of flips. However, unlike quasi-wideness which was introduced in the context of preservation and then shown to coincide with nowhere density, these conditions were introduced purely for the purpose of providing combinatorial characterisations of monadic stability and monadic dependence respectively. The immediate question thus becomes whether these conditions, or variants thereof, can be used to obtain preservation in restricted tame dense classes, in analogy to the use of wideness in [3, 4, 13, 14].

The first observation is that the arguments for homomorphism preservation are not directly adaptable in this context due to the nature of flips. Indeed, while the vertex-deletion operation respects the existence of a homomorphism between two structures, the flip operation is not at all rigid with respect to homomorphisms precisely because the latter do not reflect relations, e.g. the graph $K_1 + K_1$ homomorphically maps to K_2, but $\overline{(K_1 + K_1)} = K_2$ does not map to $\overline{K_2} = K_1 + K_1$. This issue evidently disappears if one considers embeddings. As it was observed in [14, Corollary 2.3], the extension preservation property implies the homomorphism preservation property in hereditary classes of finite structures, so considering extension preservation is more general for our purposes.

However, this generality comes at a cost. Indeed, the argument for extension preservation from [3] requires that the number of vertex-deletions is independent of the choice of radius, a condition known as *almost-wideness*. This is a more restrictive assumption which therefore applies to fewer sparse classes. It is not known whether extension preservation is obtainable for quasi-wide classes. At the same time, unlike [14, Theorem 4.2] whose proof is essentially a direct application of Gaifman's locality theorem based on an argument of Ajtai and Gurevich [2], the proof of extension preservation [3, Theorem 4.3] is admittedly much more cumbersome. One explanation for this is that the homomorphism preservation argument relies on the fact that the disjoint union operation endows the category of graphs and homomorphisms with *coproducts*, i.e. for any graphs A, B, C if there are homomorphisms $f : A \to C$ and $g : B \to C$ then there is a homomorphism $f + g : A + B \to C$ whose pre-compositions with the respective inclusion homomorphisms $\iota_A : A \to A + B$ and $\iota_B : B \to A+B$ are equal to f and g respectively. On the other hand, no construction satisfies the above property in the category of graphs with embeddings; in fact coproducts do not even exist in the category of graphs with strong homomorphisms (see [23, Corollary 4.3.15]).

Our first contribution is negative, showing that extension preservation can fail on tame dense classes of low complexity. In particular, we show that extension preservation fails on the class of all graphs of (linear) cliquewidth at most k, for all $k \geq 4$. This answers negatively a question of [14]. This is contrary to the sparse picture, where it was shown that extension preservation holds in the class of graphs of treewidth at most k, for every $k \in \mathbb{N}$ [3, Theorem 5.2]. Interestingly, extension preservation holds for the class of all graphs of cliquewidth 2 as this class coincides with the class of cographs which is known to be well-quasi-ordered [11]. Our construction is based on the encoding of linear orders via the neighbourhoods of certain vertices. Orders are also central to the original counterexample for the failure of extension preservation in the finite due to Tait [28]. There, the orders are crucially presented over a signature with two relation symbols and one constant, which does not allow for a direct translation to undirected graphs. Sadly, the fact that orders appear to provide counterexamples rules out the possibility of using an argument based on flip-breakability to establish preservation.

The second contribution of the article is positive. In particular, we provide a dense analogue to [3, Theorem 4.3]. For this, we introduce *strongly flip-flat* classes, i.e. those flip-flat classes such that the number of flips is independent of the choice of radius. Moreover, we formulate the dense analogue of the amalgamation construction, which we call the *flip-sum*, whose existence in the class is necessary for the argument to be carried out. The main theorem (Theorem 18 below) may thus be formulated as saying that extension preservation holds over any hereditary strongly flip-flat class which is closed under flip-sums over bottleneck partitions.

2 Preliminaries

We assume familiarity with the standard notions of finite model theory and structural graph theory, and refer to [19] and [25] for reference. In this article, graphs shall always refer to simple undirected graphs i.e. structures over the signature $\tau_E = \{E\}$ where E is interpreted as a symmetric and anti-reflexive binary relation. For a graph G we write $V(G)$ for its domain (or vertex set), and $E(G)$ for its edge set. In general, for a τ-structure A and a relation symbol $R \in \tau$ of arity $r \in \mathbb{N}$ we write $R^A \subseteq A^r$ for the interpretation of R in A. We shall abuse notation and not distinguish between structures and their respective domains.

Given two structures A, B in the same relational signature τ, a homomorphism $f : A \to B$ is a map that preserves all relations, i.e. for all $R \in \tau$ and tuples $\bar{a}$ from A we have $\bar{a} \in R^A \implies f(\bar{a}) \in R^B$. A *strong* homomorphism is a homomorphism $f : A \to B$ that additionally reflects all relations, i.e. $f(\bar{a}) \in R^B \implies \bar{a} \in R^A$. An injective strong homomorphism is called an *embedding* or *extension*.

A τ-structure B is said to be a *weak substructure* of a τ-structure A if $B \subseteq A$ and the inclusion map $\iota : B \hookrightarrow A$ is a homomorphism. Likewise, B is an *induced substructure* of A if the inclusion map is an embedding; we write $B \leq A$ for this. Given a structure A and a subset $S \subseteq A$ we write $A[S]$ for the unique induced substructure of A with domain S. An induced substructure B of A is said to be *proper* if $B \subsetneq A$; we write $B \lneq A$ for this. We say that a class of structures in the same signature is *hereditary* if it is closed under induced substructures. Moreover a class is called *addable* if it is closed under taking disjoint unions, which we denote by $A + B$.

By the *Gaifman graph* of a structure A we mean the undirected graph $\mathsf{Gaif}(A)$ with vertex set A such that two elements are adjacent if, and only if, they appear together in some tuple of a relation of A. Given a structure A, $r \in \mathbb{N}$, and $a \in A$, we write $N_r^A(a)$ for the *r-neighbourhood of a in A*, that is, the set of elements of A whose distance from a in $\mathsf{Gaif}(A)$ is at most r. We shall often abuse notation and write $N_r^A(a)$ for the induced substructure $A[N_r^A(a)]$ of A. For a set $C \subseteq A$ we define $N_r^A(C) := \bigcup_{a \in C} N_r^A(a)$. A set $S \subseteq A$ is said to be *r-independent* if $b \notin N_r^A(a)$ for any $a, b \in S$.

For $r \in \mathbb{N}$, let $\mathrm{dist}(x, y) \leq r$ be the first-order formula expressing that the distance between x and y in the Gaifman graph is at most r, and $\mathrm{dist}(x, y) > r$ its negation. Clearly, the quantifier rank of $\mathrm{dist}(x, y) \leq r$ is at most r. A *basic local sentence* is a sentence

$$\exists x_1, \ldots, x_n \Big(\bigwedge_{i \neq j} \mathrm{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i \in [n]} \psi^{N_r(x_i)}(x_i) \Big),$$

where $\psi^{N_r(x_i)}(x_i)$ denotes the relativisation of ψ to the r-neighbourhood of x_i, i.e. the formula obtained from ψ by replacing every quantifier $\exists x\ \theta$ with $\exists x(\mathrm{dist}(x_i, x) \leq r \wedge \theta)$, and likewise every quantifier $\forall x\ \theta$ with $\forall x(\mathrm{dist}(x_i, x) \leq r \to \theta)$. We call r the *locality radius*, n the *width*, and ψ the *local condition* of ϕ. Recall the Gaifman locality theorem [19, Theorem 2.5.1].

▶ **Theorem 1** (Gaifman Locality). *Every first-order sentence of quantifier rank q is equivalent to a Boolean combination of basic local sentences of locality radius 7^q.*

A class $\mathcal{C}$ of structures is said to be *quasi-wide* if for every $r \in \mathbb{N}$ there exist $k_r \in \mathbb{N}$ and $f_r : \mathbb{N} \to \mathbb{N}$ such that for all $m \in \mathbb{N}$ and all $A \in \mathcal{C}$ of size at least $f_r(m)$ there exists $S \subseteq A$ such that $A \setminus S$ contains an r-independent set of size m. Moreover, if $k_r := k \in \mathbb{N}$ is independent of r, then $\mathcal{C}$ is said to be *almost-wide*. Finally, we say that a class $\mathcal{C}$ is *uniformly* quasi-wide (uniformly almost-wide respectively) if the hereditary closure of $\mathcal{C}$ is quasi-wide (almost-wide respectively).

For a graph G and a pair of disjoint vertex subsets U and V, the subgraph *semi-induced* by U and V is the bipartite graph with sides U and V that contains all edges of G with one endpoint in U and second in V. By the *half-graph of order n* we mean the bipartite graph with vertices $\{u_i, v_i : i \in [n]\}$ and edges $\{(u_i, v_j) : i \leq j\}$.

For first-order formulas $\delta(x)$ and $\phi(x, y)$ the interpretation $I_{\delta,\phi}$ is defined to be the operation that maps a graph G to the graph $H := I_{\delta,\phi}(G)$ with vertex set $V(H) := \{v \in V(G) \colon G \models \delta(v)\}$ and edge set

$$E(H) := \left\{(u, v) \in V(H)^2 \colon u \neq v \wedge G \models (\phi(u, v) \vee \phi(v, u))\right\}.$$

For a graph class $\mathcal{C}$, we write $I_{\delta,\phi}(\mathcal{C}) := \{I_{\delta,\phi}(G) : G \in \mathcal{C}\}$. We say that a class $\mathcal{C}$ *is an interpretation of a class* $\mathcal{D}$, or that $\mathcal{D}$ *interprets* $\mathcal{C}$, if there is some $I_{\delta,\phi}$ such that $\mathcal{C} \subseteq I_{\delta,\phi}(\mathcal{D})$. We say that $\mathcal{C}$ *is a transduction of a class* $\mathcal{D}$, or $\mathcal{D}$ *transduces* $\mathcal{C}$, if there are $k \in \mathbb{N}$ and unary predicates $P_1, \ldots, P_k$ and formulas $\delta(x)$ and $\phi(x, y)$ over the signature $\tau_E \cup \{P_1, \ldots, P_k\}$ such that $\mathcal{C} \subseteq I_{\delta,\phi}(\mathcal{D}^k)$, where $\mathcal{D}^k$ is the class of all $\tau_E \cup \{P_1, \ldots, P_k\}$-structures whose τ_E-reducts are in $\mathcal{D}$. A graph class $\mathcal{C}$ is *monadically dependent* if $\mathcal{C}$ does not transduce the class of all graphs. $\mathcal{C}$ is moreover *monadically stable* if $\mathcal{C}$ does not transduce the class of all half-graphs.

We say that a formula ϕ is preserved by extensions over a class of structures $\mathcal{C}$ if for all $A, B \in \mathcal{C}$ such that there is a embedding from B to A, $B \models \phi$ implies that $A \models \phi$. We say that a class of structures $\mathcal{C}$ has the *extension preservation property* if for every formula ϕ preserved by extensions over $\mathcal{C}$ there is an existential formula ψ such that $M \models \phi \iff M \models \psi$ for all $M \in \mathcal{C}$. We analogously define the *homomorphism preservation property*, replacing "embeddings" with "homomorphisms" and "existential" with "existential positive" in the above.

Given a formula ϕ and a class of structures $\mathcal{C}$, we say that $M \in \mathcal{C}$ is a *minimal induced model* of ϕ in $\mathcal{C}$ if $M \models \phi$ and for any proper induced substructure N of M with $N \in \mathcal{C}$ we have $N \not\models \phi$. The relationship between minimal models and extensions preservation is highlighted by the following folklore lemma. We provide a proof for completeness.

▶ **Lemma 2.** *Let $\mathcal{C}$ be a hereditary class of finite structures. Then a sentence preserved by extensions in $\mathcal{C}$ is equivalent to an existential sentence over $\mathcal{C}$ if and only if it has finitely many minimal induced models in $\mathcal{C}$.*

Proof. Suppose that ϕ has finitely many minimal induced models in $\mathcal{C}$, say $M_1, \ldots, M_n$. For each $i \in [n]$, let ψ_i be the primitive sentence inducing a copy of M_i and write $\psi := \bigvee_{i\in[n]} \psi_i$; evidently ψ is existential. We argue that ϕ is equivalent to ψ over $\mathcal{C}$. Indeed, if $A \in \mathcal{C}$ models ϕ then A contains a minimal induced model B of ϕ as an induced substructure. By hereditariness $B \in \mathcal{C}$ and so B is isomorphic to some M_i. Since there is clearly an embedding $B \to A$ it follows that $A \models \psi$. On the other hand if $A \models \psi$, then $A \models \psi_i$ for some $i \in [n]$ and so some M_i embeds into A. Since $M_i \models \phi$ and ϕ is preserved by extensions this implies that $A \models \phi$ as required.

Conversely, assume that ϕ is equivalent to an existential sentence over $\mathcal{C}$. In particular, ϕ is equivalent to some disjunction $\bigvee_{i\in[n]} \psi_i$ where each ψ_i is primitive. It follows that each ψ_i is the formula inducing one of finitely many structures $M_1^i, \ldots, M_{k_i}^i$. Now, if A is a minimal induced model of ϕ then in particular $A \models \psi_i$ for some $i \in [n]$, i.e. there is some $j \in [k_i]$ and an embedding $h : M_j^i \to A$. If h is not surjective, then $A[h[M_j^i]]$ is a proper induced substructure of A, which is in $\mathcal{C}$ by hereditariness, and models ϕ; this contradicts the minimality of A. Hence, the size of every minimal induced model of ϕ in $\mathcal{C}$ is bounded by $\max_{i\in[n]} \max_{j\in[k_i]} |M_j^i|$. It follows that ϕ can have only finitely many minimal induced models in $\mathcal{C}$. ◀

3 Failure of preservation on graphs of cliquewidth 4

One consequence of Lemma 2 is that extension preservation holds over any class $\mathcal{C}$ that is *well-quasi-ordered* by the induced substructure relation, i.e. classes for which there exists no infinite collection of members which pairwise do not embed into one another. In particular, this applies to the class of *cographs* [11], which are precisely the graphs of cliquewidth 2 (see [9] for background on cliquewidth). Hence, one may reasonably inquire whether extension preservation is generally true for the class $\mathcal{CW}_k$ of all graphs of cliquewidth at most k. This

would in particular reflect an analogous phenomenon that is true in the sparse setting, that is, that extension preservation holds over the class $\mathcal{TW}_k$ of all graphs of treewidth at most k [3, Theorem 5.2].

Classes of bounded cliquewidth are not monadically stable, as even the class of cographs contains arbitrarily large semi-induced half-graphs, but they are monadically dependent. In fact, their structural properties imply tame behaviour going much beyond the context of first-order logic (see [10] for a survey). Still, as it turns out, extension preservation fails even at the level of cliquewidth 4. To show this, we produce a formula ϕ preserved by extensions over the class of all finite undirected graphs, which admits infinitely many minimal models of cliquewidth 4. In particular, Lemma 2 implies that extension preservation fails on any class that includes these minimal models. Our idea is based on encoding two interweaving linear orders on the two parts of a semi-induced graph. Two vertices on the same part are comparable in this ordering whenever their neighbourhoods in the other part are set-wise comparable. This effectively forces a semi-induced half-graph.

Our formula is in the form of an implication, proceeded by a primitive part which induces a gadget corresponding to the beginning and end of the two linear orders. The antecedent of the implication first makes sure that the above relation is a pre-ordering on each side of the semi-induced graph, while it imposes that the vertices of the gadget corresponding to the minimal and maximal elements are indeed minimal and maximal in this pre-ordering. Moreover, it essentially ensures that successors, i.e. vertices of the same part whose neighbourhoods over the other part differ by a single element, are adjacent on one part and non-adjacent on the other. The consequent then imposes that any vertex on the first side has an adjacent successor, while every vertex on the other side has a non-adjacent successor. Because each one of the two pre-orders precisely compares neighbourhoods over the other part, this forces the pre-orders to be anti-symmetric, and thus the two parts to have the same number of vertices.

Finally, two additional vertices are also added on one side of the semi-induced bipartite graph, which are part of the gadget and serve no role in this ordering. These make sure that our intended minimal models form an anti-chain in the embedding relation, as they crucially result in the existence of a unique embedding of the gadget into the models (Lemma 5 below).

We now turn to formal definitions. Let $I(v_1, v_2, v_3, v_4, v_5, v_6, u_1, u_2, u_3, u_4, u_5, u_6, a, b)$ be the formula that induces the graph of Figure 1 below. In the following, we treat the free variables of I as constants for simplicity. The notation $\forall(x \in U)$ will denote the relativisation of the universal quantifier to the neighbours of v_1 that are not v_2, i.e. $\forall(x \in U)\ \psi(x)$ is shorthand for $\forall x(E(x, v_1) \wedge x \neq v_2 \to \psi(x))$. Likewise, the notation $\forall(x \in V)$ denotes the relativisation of the universal quantifier to the non-neighbours of v_1 that are not a or b, i.e. $\forall(x \in V)\ \psi(x)$ is shorthand for $\forall x(\neg E(x, v_1) \wedge x \neq a \wedge x \neq b \to \psi(x))$. Existential quantifiers relativised to U and V are defined analogously. Consider the auxiliary formulas:

$$x \leq_V y := \forall(z \in U)[E(z, x) \to E(z, y)];$$

$$x <_V y := x \leq_V y \wedge \neg(y \leq_V x);$$

$$\chi_1 := \forall(x \in V)\forall(y \in V)[x \leq_V y \vee y \leq_V x];$$

$$\chi_2 := \forall(x \in U)[E(x, v_6) \to x = u_6];$$

$$\chi_3 := \forall(x \in V)\forall(y \in V)[x <_V y \wedge E(x, y) \to \exists!(z \in U)(E(y, z) \wedge \neg E(x, z))].$$

In analogy, we define:

$$x \leq_U y := \forall(z \in V)[E(z, x) \to E(z, y)];$$

$x <_U y := x \leq_U y \wedge \neg(y \leq_U x)$;

$\xi_1 := \forall(x \in U)\forall(y \in U)[x \leq_U y \vee y \leq_U x]$;

$\xi_2 := \forall(x \in V)[E(x, u_1) \to x = v_1]$;

$\xi_{2^*} := \forall(x \in V)\ E(x, u_6)$;

$\xi_3 := \forall(x \in U)\forall(y \in U)[x <_U y \wedge \neg E(x, y) \to \exists!(z \in V)(E(y, z) \wedge \neg E(x, z))]$.

We then define:

$\phi_1 := \chi_1 \wedge \chi_2 \wedge \chi_3$;

$\phi_2 := \forall(x \in V)[x \neq v_1 \to \exists(y \in V)(E(x, y)\ \wedge\ x <_V y)]$;

$\psi_1 := \xi_1 \wedge \xi_2 \wedge \xi_{2^*} \wedge \xi_3$;

$\psi_2 := \forall(x \in U)[x \neq u_6 \to \exists(y \in U)(\neg E(x, y) \wedge x <_U y)]$.

Putting the above together we finally define:

$\phi := \exists \bar{v}, \bar{u}, a, b(I(\bar{v}, \bar{u}, a, b) \wedge [\phi_1(\bar{v}, \bar{u}, a, b) \wedge \psi_1(\bar{v}, \bar{u}, a, b) \to \phi_2(\bar{v}, \bar{u}, a, b) \wedge \psi_2(\bar{v}, \bar{u}, a, b)])$

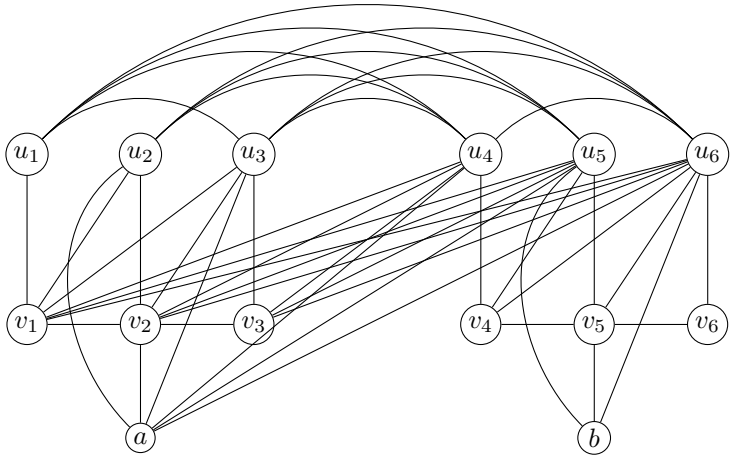


Figure 1 The gadget induced by $I(\bar{v}, \bar{u}, a, b)$.

▶ **Proposition 3.** *The formula ϕ is preserved by extensions over the class of all finite graphs.*

Proof. Let G, H be two graphs such that G embeds into H, and $G \models \phi$. Without loss of generality we assume that $V(G) \subseteq V(H)$ and that the identity map is an embedding. We shall argue that $H \models \phi$.

Since $G \models \phi$, we may fix (tuples of) vertices $\bar{v}, \bar{u}, a, b \in V(G)$ such that $G \models I(\bar{v}, \bar{u}, a, b)$. Evidently, H also models $I(\bar{v}, \bar{u}, a, b)$. If $H \not\models (\phi_1(\bar{v}, \bar{u}, a, b) \wedge \psi_1(\bar{v}, \bar{u}, a, b))$ then $H \models \phi$; we may therefore assume that $H \models (\phi_1(\bar{v}, \bar{u}, a, b) \wedge \psi_1(\bar{v}, \bar{u}, a, b))$. Let $U := N_H(v_1) \setminus \{v_2\} \subseteq V(H)$ be the neighbours of v_1 in H that are not v_2, and $V := V(H) \setminus (U \cup \{a, b\}) \subseteq V(H)$ be the non-neighbours of v_1 that are not a or b (in particular $v_1 \in V$). We call the vertices $x \in V$ *V-elements*. For each V-element x we write $U_x := \{y \in U : H \models E(x, y)\}$ for the *U-neighbourhood of x*. We similarly define *U-elements* and *V-neighbourhoods*. From each of the conjuncts χ_i of ϕ_1 we deduce that in H:

χ_1: V-elements have pairwise comparable U-neighbourhoods;
χ_2: the only member of U_{v_6} is u_6;
χ_3: if two adjacent V-elements x, y satisfy $U_x \subsetneq U_y$ then $|U_y| = |U_x| + 1$.

We shall argue that items (1)-(3) are still true within G, replacing V with $V' := V \cap V(G)$ and U with $U' := U \cap V(G)$, and so $G \models \phi_1(\bar{v}, \bar{u}, a, b)$. We write $U'_x := U_x \cap V(G)$ for the relativised U'-neighbourhoods. Clearly, items (1) and (2) are still true in G. For item (3), suppose that $x, y \in V'$ are two adjacent V'-elements, such that $V'_x \subsetneq V'_y$. Since V-elements in H have pairwise comparable U-neighbourhoods, we deduce that $V_x \subsetneq V_y$ and therefore that $|V_y| = |V_x| + 1$ as H satisfies χ_3. In particular, it follows that $|V'_y| = |V'_x| + 1$ as required, and so G models $\chi_3(\bar{v}, \bar{u}, a, b)$ and consequently $\phi_1(\bar{v}, \bar{u}, a, b)$.

Similarly, from each of the conjuncts ξ_i of ψ_1 we deduce that in H:
ξ_1: U-elements have pairwise comparable V-neighbourhoods;
ξ_2: the only element of V_{u_1} is v_1;
ξ_{2^*}: V_{u_6} is equal to V;
ξ_3: if two non-adjacent U-elements satisfy $V_x \subsetneq V_y$ then $|V_y| = |V_x| + 1$.

Arguing as before, we obtain that $G \models \psi_1(\bar{v}, \bar{u}, a, b)$. Since $G \models \phi$ and $G \models (\phi_1(\bar{v}, \bar{u}, a, b) \wedge \psi_1(\bar{v}, \bar{u}, a, b))$ we deduce that $G \models (\phi_2(\bar{v}, \bar{u}, a, b) \wedge \psi_2(\bar{v}, \bar{u}, a, b))$, i.e. the following are true in G:
ϕ_2: every V'-element that is not v_1 is adjacent to a V'-element of strictly greater U'-neighbourhood;
ψ_2: every U'-element that is not u_6 is non-adjacent to a U'-element of strictly greater V'-neighbourhood.

We proceed to show that the above implies that $V = V'$ and $U' = U$, and hence $G = H$. In particular, this implies that $H \models \phi$ as claimed.

Since G is finite and satisfies ϕ_2 we obtain some $n \in \mathbb{N}$ and a sequence of distinct elements $\alpha_1 := v_6, \alpha_2, \ldots, \alpha_n := v_1$ of V' such that $U'_{\alpha_i} \subsetneq U'_{\alpha_{i+1}}$ and $G \models E(\alpha_i, \alpha_{i+1})$ for all $i \in [n-1]$. In particular, $U_{\alpha_i} \subsetneq U_{\alpha_{i+1}}$ and $H \models E(\alpha_i, \alpha_{i+1})$ for all $i \in [n]$. As H satisfies χ_3 we obtain that $|U_{\alpha_{i+1}}| = |U_{\alpha_i}| + 1$ for all i. Moreover, since H satisfies χ_2 and every element of U is adjacent to v_1, we obtain that $U_{\alpha_1} = \{u_6\}$ and $U_{\alpha_n} = U$. In particular, we deduce that $n = |U| \leq |V'|$. Symmetrically, we obtain some $k \in \mathbb{N}$ and a sequence of elements $\beta_1 := u_1, \beta_2, \ldots, \beta_k := u_6$ of U' such that $V'_{\beta_i} \subsetneq V'_{\beta_{i+1}}$ and $G \models \neg E(\beta_i, \beta_{i+1})$ for all $i \in [k-1]$. Hence, $V_{\beta_i} \subsetneq V_{\beta_{i+1}}$ and $H \models \neg E(\beta_i, \beta_{i+1})$ for all $i \in [k-1]$. Once again, since H satisfies ξ_2, ξ_{2^*}, and ξ_3 we obtain that $V_{\beta_1} = \{v_1\}$, $V_{\beta_n} = V$, and $|V_{\beta_{i+1}}| = |V_{\beta_i}| + 1$. It thus follows that $k = |V| \leq |U'|$. Putting the above together we have that

$$|U| \leq |V'| \leq |V| \leq |U'| \leq |U|.$$

Consequently, $n = k$ while $V = V' = \{\alpha_1, \ldots, \alpha_n\}$ and $U = U' = \{\beta_1, \ldots, \beta_n\}$ as needed. ◀

We now define the intended minimal induced models of our formula ϕ.

▶ **Definition 4.** *For $n \geq 7$ we define the graph $\mathcal{H}_n$ with vertex and edge set*

$$V(\mathcal{H}_n) := \{v_1, \ldots, v_n\} \cup \{u_1, \ldots, u_n\} \cup \{a\} \cup \{b\};$$

$$E(\mathcal{H}_n) := \{(v_i, u_j) : i \leq j\} \cup \{(v_i, v_j) : j = i + 1\} \cup \{(u_i, u_j) : j \neq i + 1\}$$
$$\cup \{(a, u_i) : i \geq 2\} \cup \{(b, u_i) : i \geq n - 1\} \cup \{(a, v_2), (b, v_{n-1})\},$$

respectively. We also write $\mathcal{I}_n$ for the subgraph of $\mathcal{H}_n$ induced on the set

$$V(\mathcal{I}_n) := \{v_1, v_2, v_3, v_{n-2}, v_{n-1}, v_n, u_1, u_2, u_3, u_{n-2}, u_{n-1}, u_n, a, b\} \subseteq V(\mathcal{H}_n).$$

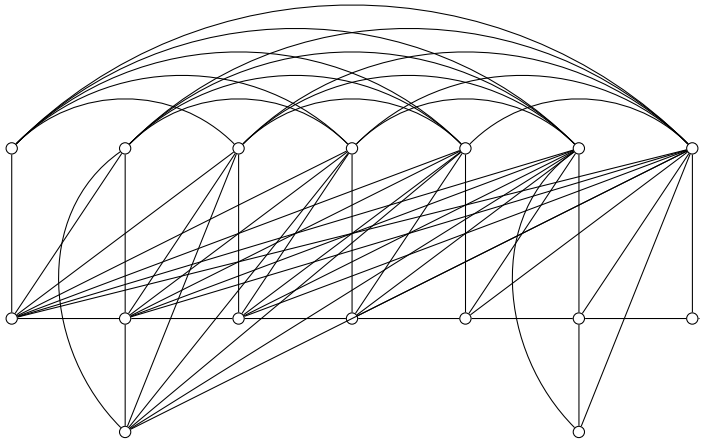


Figure 2 The graph $\mathcal{H}_7$.

We aim to establish that the graphs $\mathcal{H}_n$ are all minimal induced models of ϕ. Towards this, we first argue that the only embedding of $\mathcal{I}_n$ in $\mathcal{H}_n$ is the inclusion map. While this lemma is not conceptually difficult, it requires analysing and ruling out different cases corresponding to potential images of the gadget. Its proof may be found in Section A.

▶ **Lemma 5.** *Let $n \geq 7$ and $f : \mathcal{I}_n \to \mathcal{H}_n$ be an embedding. Then f is the inclusion map.*

▶ **Proposition 6.** *For each $n \geq 7$ the graphs $\mathcal{H}_n$ are minimal induced models of ϕ.*

Proof. We fix some $n \geq 7$. We first argue that $\mathcal{H}_n \models \phi$ for every $n \geq 7$. Indeed, we clearly have that

$$\mathcal{H}_n \models I(v_1, v_2, v_3, v_{n-2}, v_{n-1}, v_n, u_1, u_2, u_3, u_{n-2}, u_{n-1}, u_n, a, b).$$

Moreover, the set $U := \{u_1, \dots, u_n\} \subseteq V(\mathcal{H}_n)$ is precisely the set of neighbours of v_1 which are not v_2, while the set $V := \{v_1, \dots, v_n\} \subseteq V(\mathcal{H}_n)$ is precisely the set of non-neighbours of v_1 which are not a or b. Evidently, we then have that for every vertex $v_i \in V \setminus \{v_1\}$ the vertex $v_{i-1} \in V$ is adjacent to v_i and its neighbourhood over U strictly contains that of v_i. Consequently $\mathcal{H}_n \models \phi_2(\bar{v}, \bar{u}, a, b)$. Likewise, for every vertex $u_i \in U \setminus \{u_n\}$ the vertex $u_{i+1} \in U$ is non-adjacent to u_i and its neighbourhood over V strictly contains that of u_i. It follows that $\mathcal{H}_n \models \psi_2(\bar{v}, \bar{u}, a, b)$, and so $\mathcal{H}_n \models \phi$ as required.

Now, suppose that H is a proper induced subgraph of $\mathcal{H}_n$, and assume for a contradiction that $H \models \phi$, i.e. there are vertices $x_1, \dots, x_6, y_1, \dots, y_6, \alpha, \beta$ of H

$$H \models (I(\bar{x}, \bar{y}, \alpha, \beta) \wedge [\phi_1(\bar{x}, \bar{y}, \alpha, \beta) \wedge \psi_1(\bar{x}, \bar{y}, \alpha, \beta) \to \phi_2(\bar{x}, \bar{y}, \alpha, \beta) \wedge \psi_2(\bar{x}, \bar{y}, \alpha, \beta)]).$$

Since these vertices induce a copy of $\mathcal{I}_n$, it follows by Lemma 5 that

$$(x_1, x_2, x_3, x_4, x_5, x_6, y_1, y_2, y_3, y_4, y_5, y_6, \alpha, \beta) = \\ (v_1, v_2, v_3, v_{n-2}, v_{n-1}, v_n, u_1, u_2, u_3, u_{n-2}, u_{n-1}, u_n, a, b),$$

and so $\mathcal{I}_n \leq H \lneq \mathcal{H}_n$. Moreover, letting $U' := U \cap V(H)$ and $V' := V \cap V(H)$ we see that

- the elements in V' have pairwise comparable neighbourhoods over U', and the elements of U' have pairwise comparable neighbourhoods over V';
- the only neighbour of v_n in U' is u_n, and the only neighbour of u_1 in V' is v_1;

- u_n is adjacent to every element of V;
- if $x, y \in V'$ are adjacent and the U'-neighbours of y are strictly more than the U'-neighbours of x then there is some $i \in [n-1]$ such that $y = v_i$ and $x = v_{i+1}$, and there is a unique vertex in U' that is adjacent to y and not adjacent to x, namely u_i;
- if $x, y \in U'$ are adjacent and the V'-neighbours of y are strictly more than the V'-neighbours of x then there is some $i \in [n-1]$ such that $y = v_{i+1}$ and $y = v_i$, and there is a unique vertex in V' that is adjacent to y and not adjacent to x, namely v_i.

It follows that $H \models (\phi_1(\bar{v}, \bar{u}, a, b) \wedge \psi_1(\bar{v}, \bar{u}, a, b))$. Since $H \models \phi$ this implies that $H \models (\phi_2(\bar{v}, \bar{u}, a, b) \wedge \psi_2(\bar{v}, \bar{u}, a, b))$. However, since $V(H) \subsetneq V(\mathcal{H}_n)$, there is some $i \in [4, n-3]$ such that $v_i \notin V(H)$ or $u_i \notin V(H)$. Assume the former, and let $i \in [4, n-3]$ be maximal such that $v_i \notin V(H)$. It follows that there is no vertex in $x \in V'$ that is adjacent to v_{i+1} and its neighbourhood over U' is strictly greater than that of v_{i+1}, contradicting that $H \models \psi_1(\bar{v}, \bar{u}, a, b)$. By a symmetric argument we obtain a contradiction if $u_i \notin V(H)$, and thus follows that $H \not\models \phi$. ◀

▶ **Theorem 7.** *Extension preservation fails on any hereditary graph class containing the graphs $\mathcal{H}_n$ for arbitrarily large $n \in \mathbb{N}$.*

Proof. Let $\mathcal{C}$ be a class of graphs containing the graphs $\mathcal{H}_n$ for arbitrarily large n. Since the formula ϕ is preserved under extensions over the class of all finite graphs, it is in particular preserved under extensions over $\mathcal{C}$. Since $\mathcal{C}$ is hereditary and ϕ has infinitely many minimal induced models in $\mathcal{C}$, namely the graphs $\mathcal{H}_n$, it follows by Lemma 2 that ϕ is not equivalent to an existential formula over $\mathcal{C}$. ◀

Finally, we observe that the graphs $\mathcal{H}_n$ have bounded cliquewidth, which is easily seen to be at most 4. For this, we crucially use the fact that successive pairs are adjacent on one side and non-adjacent on the other. One could simplify the construction, e.g. by using adjacency to denote succession on both sides, but this would slightly increase the cliquewidth.

▶ **Observation 8.** *The graphs $\mathcal{H}_n$ have (linear) cliquewidth 4.*

▶ **Corollary 9.** *Extension preservation fails on $\mathcal{CW}_k$ for every $k \geq 4$.*

As witnessed by the above, orders appear to provide strong counterexamples to extension preservation. In the next section we explore preservation in certain monadically stable classes, where no such issues are expected to arise.

4 Extension preservation on strongly flip-flat classes

Local information on dense graphs can be as complicated as global information, as for instance is the case with cliques. This fact seemingly renders locality useless in the context of dense graph classes. Nonetheless, our understanding of tame classes indicates that it is still possible to recover meaningful local information, after possibly "sparsifying" our graphs in a controlled manner. The flip operation, which is central to the emerging theory of dense graph classes, plays precisely this role. We introduce it in the following definition.

▶ **Definition 10.** *Let G be a graph and $k \in \mathbb{N}$. A k-partition P of G is a partition of the vertex set into k labelled parts $P_1, \ldots, P_k$, i.e. $V(G) = \bigcup_{i \in [k]} P_i$ and $P_i \cap P_j = \emptyset$ for $i \neq j$. By a k-flip F we denote a symmetric subset of $[k]^2$, i.e. a set of tuples $F = \{(i,j) : i, j \in [k]\}$ such that $(i,j) \in F \iff (j,i) \in F$. Given a k-partition P of G and a k-flip F we define the graph $G \triangle_F P$ on the same vertex set as G and on the edge set*

$$E(G \triangle_F P) := E(G) \triangle \{(u,v) : u \neq v, u \in P_i, v \in P_j, \text{ and } (i,j) \in F\}.$$

where $\triangle$ denotes the symmetric difference operation.

We note that the notation for flips existing in the literature uses the notation $\oplus$ rather than $\triangle$ (e.g. in [17]); here we have opted for the latter as the symbol $\oplus$ was used in [3] and [14] to denote the amalgamation operation. Moreover, instead of partitioning our graph, we may define k-flips by applying a sequence of at most k atomic operations, each one switching the edges and non-edges between two arbitrary subsets A, B of our vertex set. Evidently, these definitions are equivalent up to blowing up the number of flips by a value that only depends on k, while we have opted for the partition definition here to simplify our construction in Definition 14 below.

▶ **Definition 11.** *We say that a hereditary class of graphs $\mathcal{C}$ is* flip-flat[1] *if for every $r \in \mathbb{N}$ there exist $k_r \in \mathbb{N}$ and a function $f_r : \mathbb{N} \to \mathbb{N}$ satisfying that for every $m \in \mathbb{N}$ and every $G \in \mathcal{C}$ of size at least $f_r(m)$ there is a k_r-partition P of G, a k_r-flip $F \subseteq [k_r]^2$, and a set $A \subseteq V(G)$ of size at least m which is r-independent in $G \triangle_F P$. If in the above $k_r := k \in \mathbb{N}$ does not depend on r, then we say that $\mathcal{C}$ is* strongly flip-flat.

It was established in [17, Theorem 1.3] that a hereditary class of graphs is flip-flat if, and only if, it is monadically stable. In particular, every transduction of a quasi-wide class is flip-flat. The qualitative difference between strong flip-flatness and flip-flatness is precisely the same as that of almost-wideness and quasi-wideness. We make this idea precise in the following straightforward proposition, which establishes that every transduction of a uniformly almost-wide class is strongly flip-flat. For this, we use the following lemma from [29, Lemma H.3], which follows easily from Gaifman's locality theorem.

▶ **Lemma 12** (Flip transfer lemma, [29])**.** *There exists a (computable) function $\Xi : \mathbb{N}^3 \to \mathbb{N}$ satisfying the following. Fix $k, c, q \geq 1$ and $\mathcal{T}_{\delta,\phi}$ a transduction involving c colours and formulas of quantifier rank at most q. Let G, H be graphs such that $H \in T_{\delta,\phi}(G)$. Then for every k-partition P of G and k-flip F there exists a $\Xi(k,c,q)$-partition P_H of H and a $\Xi(k,c,q)$-flip F_H such that for all $u, v \in V(H)$:*

$$\mathrm{dist}_{G \triangle_F P}(u, v) \leq 2^q \cdot \mathrm{dist}_{H \triangle_{F_H} P_H}(u, v).$$

▶ **Proposition 13.** *Every transduction of a uniformly almost-wide graph class is strongly flip-flat.*

Proof. Let $\mathcal{C}$ be a uniformly almost-wide graph class and fix $k_{\mathcal{C}} \in \mathbb{N}$ witnessing this, so that for every $r, m \in \mathbb{N}$ there is $f_r(m) \in \mathbb{N}$ satisfying that every G of size at least $f_r(m)$ in the hereditary closure of $\mathcal{C}$ contains an r-independent set of size m after removing at most $k_{\mathcal{C}}$ elements. Let $\mathcal{D}$ a class such that there is a transduction $T_{\delta,\phi}$ satisfying $\mathcal{D} \subseteq T_{\delta,\phi}(\mathcal{C})$. Let $c \in \mathbb{N}$ be the number of unary predicates used by T, and q be the maximum of the quantifier ranks of δ and ϕ. We argue that $\mathcal{D}$ is strongly flip-flat with $k := \Xi(2^{k_{\mathcal{C}}}, c, q)$.

Indeed, fix $r, m \in \mathbb{N}$ and a graph $H \in \mathcal{D}$ of size at least $f_{2^q \cdot r}(m)$. It follows that there exists some $G \in \mathcal{C}$ such that $H \in T_{\delta,\phi}(G)$, and since $f_{2^q \cdot r}(m) \leq |V(H)|$, we obtain by uniform almost-wideness that $G[V(H)]$ contains a $(2^q \cdot r)$-independent set of size m after removing a set of size at most $k_{\mathcal{C}}$. In particular, there is a $2^{k_{\mathcal{C}}}$-partition P of G and a $2^{k_{\mathcal{C}}}$-flip F such that $(G \triangle_F P)[V(H)]$ contains an $(2^q \cdot r)$-independent subset of size m; call this set A. Consequently, Lemma 12 implies that there is a k-partition P of H and a k-flip F such that

[1] The original definition of flip-flatness in [17] is the uniform variant of the definition we have provided here. A simple analysis of the obstructions to monadic stability from [16], reveals that these definitions are equivalent for hereditary classes of graphs.

for all $a, b \in A \subseteq V(H)$

$$r = \frac{2^q \cdot r}{2^q} \leq \frac{\mathrm{dist}_{G \triangle_F P}(a,b)}{2^q} \leq \mathrm{dist}_{H \triangle_{F_H} P_H}(a,b),$$

i.e. A is an r-independent set of size m in $H \triangle_{F_H} P_H$. It follows that $\mathcal{D}$ is strongly flip-flat. ◀

In particular, transductions of bounded degree classes, classes of bounded shrub-depth [20], and transductions of proper minor-closed classes [4, Theorem 5.3] are all strongly flip-flat. However, obtaining preservation via locality and wideness in the style of [3, 4, 13, 14] additionally requires subtle closure assumptions. The proofs of the above articles are essentially structured into two parts. The first part argues via locality that for every formula ϕ preserved by extensions (or homomorphisms in the case of [4, 13, 14]) over a class $\mathcal{C}$ closed under substructures and disjoint unions there exist $r, m \in \mathbb{N}$ such that no minimal induced model of ϕ in $\mathcal{C}$ can contain an r-independent set of size m. In the second part, wideness is used to bound the size of minimal models of ϕ, as large enough models would have to contain r-independent sets of size m, under the proviso that a bounded number of bottleneck points have been removed. To account for the removal of these points, we have to work with an adjusted formula ϕ' in an expanded vocabulary, together with suitably adjusted structures ([2] called these *plebian companions*) on which we apply the argument of the first part. Working with ϕ', however, has translated the requirement of closure under disjoint unions to closure under a more involved operation which depends on the choice of bottlenecks. Consequently, preservation can fail on natural tame classes which do not satisfy this closure condition, e.g. for planar graphs [14, Theorem 5.8].

In the context of vertex deletions, the corresponding operation was *amalgamation over bottlenecks* [14, Theorem 4.2]. Here, we must formulate a different operation to account for the fact that flips are required to witness wideness. This is precisely the construction below.

▶ **Definition 14.** *Given $k \in \mathbb{N}$, a graph G, a k-partition P of G, and a k-flip $F \subseteq [k^2]$ we write $G \star_{(F,P)} G$ for the graph whose vertex set $V(G \star_{(F,P)} G) := V(G + G)$ is the same as the disjoint union of two copies of G, and whose edge set is*

$$E(G \star_{(F,P)} G) := E(G+G) \cup \{(u,v) : u, v \text{ are in distinct copies of } G, u \in P_i, v \in P_j, (i,j) \in F\}$$

We call this the flip-sum of G over (F, P).

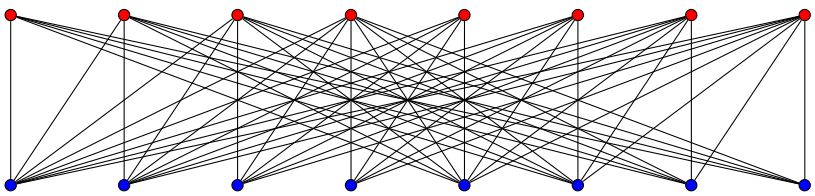


Figure 3 The graph $H \star_{(F,P)} H$, where H is the half-graph of order 4, P is the partition into red (top) and blue (bottom) vertices and $F = \{(1,2),(2,1)\}$.

We now introduce the relevant translation for the formulas.

▶ **Definition 15.** *Given a k-flip $F \subseteq [k]^2$, consider the formula*

$$E_F(x,y) := E(x,y) \triangle_{(i,j)\in F} (P_i(x) \wedge P_j(y)).$$

over the signature $\tau_E^k := \tau_E \cup \{P_1, \ldots, P_k\}$, where $\triangle_{(i,j)\in F}$ denotes the consecutive application of the XOR *operator over all tuples $(i,j) \in F$. Given a τ_E-formula ϕ, we define the τ_E^k-formula*

ϕ^k obtained from ϕ by replacing every atom $E(x,y)$ with the formula $E_F(x,y)$. Moreover, for every graph G and k-partition P we write $G_{(F,P)}$ for the $\{P_1,\dots,P_k\}$-expansion of $G\triangle_F P$ where each predicate is interpreted by the respective part of P. It is then clear from the definitions and the fact that the flip operation is involutive that

$$G \models \phi \iff G_{(F,P)} \models \phi^k.$$

Our goal in Theorem 18 is to start with a strongly flip-flat class and a formula ϕ and apply the argument of [3, Theorem 4.3] to the formula ϕ^k and the structures $G_{(F,P)}$. However, as previously explained, ϕ^k is not necessarily preserved under embeddings over $\mathcal{C}$. We can nonetheless use the following easy lemma in case that the class is closed under the desired flip-sums, which will be sufficient for our purposes.

▶ **Lemma 16.** *Let $\mathcal{C}$ be a hereditary class of graphs and ϕ a formula preserved under extensions over $\mathcal{C}$. Fix a graph $G \in \mathcal{C}$, a k-partition P of G, and a k-flip $F \subseteq [k]^2$. If $G \star_{(F,P)} G \in \mathcal{C}$ then*

$$G_{(F,P)} \models \phi^k \implies G_{(F,P)} + G_{(F,P)}[S] \models \phi^k$$

for any $S \subseteq V(G)$.

Proof. Fix $\mathcal{C}, \phi, G, P, F$ as in the statement above, and let $S \subseteq V(G)$. Write G^* for the subgraph of $G \star_{(F,P)} G$ induced on the vertex set of $G + G[S]$; it follows that $G^* \in \mathcal{C}$ by hereditariness. As $G_{(F,P)} \models \phi^k$ we obtain that $G \models \phi$, and since G^* contains an induced copy of G and ϕ is preserved by extensions over $\mathcal{C}$ it follows that $G^* \models \phi$. Let P^* be the natural k-partition of G^* inherited from G, i.e. for each $i \in [k]$ the i-th part P^*_i of P^* contains the union of the i-th parts of G and $G[S]$. It follows from the definitions that the structure $G^*_{(F,P^*)}$ is isomorphic to $G_{(F,P)} + G_{(F,P)}[S]$. Finally, since $G^* \models \phi$ we obtain that $G^*_{(F,P^*)} \models \phi^k$ and so $G_{(F,P)} + G_{(F,P)}[S] \models \phi^k$ as claimed. ◀

We shall also make use of the following observation, which simply says that the induced substructures of $G_{(F,P)}$ are the same as expansions of flips of induced substructures of G.

▶ **Observation 17.** *Let G be a graph, P a k-partition of G, and F a k-flip. Then for every $S \subseteq V(G)$ the structure $G_{(F,P)}[S]$ is equal to $G[S]_{(F,P_S)}$, where P_S is the k-partition of $G[S]$ obtained by restricting each part of P on S.*

We are now ready to state the main theorem of this section.

▶ **Theorem 18.** *Fix a hereditary class of graphs $\mathcal{C}$. Suppose that there is some $k \in \mathbb{N}$ such that for all $r \in \mathbb{N}$ there is a function $f_r : \mathbb{N} \to \mathbb{N}$ satisfying that for every $m \in \mathbb{N}$ and every $G \in \mathcal{C}$ of size at least $f(m)$ there is a k-partition P of $V(G)$, some k-flip F, and $A \subseteq V(G)$ such that*

1. *$|A| \geq m$;*
2. *A is r-independent in $G\triangle_F P$;*
3. *$G \star_{(F,P)} G \in \mathcal{C}$.*

Then extension preservation holds over $\mathcal{C}$.

The proof of Theorem 18 is an adaptation of the proof of [3, Theorem 4.3], which established that extension preservation holds over any class closed under weak substructures and disjoint unions which is *wide*, i.e. for every $r \in \mathbb{N}$ there exists $f_r : \mathbb{N} \to \mathbb{N}$ such that for every $m \in \mathbb{N}$ every structure with at least $f_r(m)$-many elements contains an r-independent

set of size m. Here, we replace wideness by strong flip-flatness by working with the formula ϕ^k of Definition 15. Moreover, as previously explained, addability is replaced by assumption 3 above. Preservation is then ensured by Lemma 22. Finally, going from closure under weak substructures to closure under induced substructures follows by analysing [3, Theorem 4.3]. The detailed proof can be found in Section B.

▶ **Example 19.** For $d \in \mathbb{N}$, write $\mathcal{D}_d$ be the class of all graphs G such that the maximum degree of G is at most d, or the maximum degree of $\overline{G}$, i.e. the complement graph of G, is at most d. Let $f_r(m) = (m-1)(d+1)^r + 1$ and consider a graph $G \in \mathcal{D}_d$ of size at least $f_r(m)$. If G has maximum degree d, then G must contain an r-independent set of size m. Consequently, letting $P = \{V(G)\}$ and $F = \emptyset$, we see that $G \star_{(F,P)} G$ is simply the disjoint union of two copies of G, which still has maximum degree d and is therefore in $\mathcal{D}_d$. On the other hand if $\overline{G}$ has maximum degree d, then for $P = \{V(G)\}$ and $F = \{(1,1)\}$, we see that $\bar{G} = G \triangle_F P$ has an r-independent set of size m. Since $G \star_{(F,P)} G$ is the complement of the disjoint union of two copies of $\overline{G}$ and so $\overline{G \star_{(F,P)} G}$ has maximum degree d, it follows that $G \star_{(F,P)} G \in \mathcal{D}_d$. Consequently, extension preservation holds over $\mathcal{D}_d$ by Theorem 18.

As mentioned above, Lemma 2 implies that any well-quasi-ordered class has the extension preservation property. In particular, this applies to classes of bounded *shrubdepth* [20, Corollary 3.9]. Still, in the following example we indirectly show that the class of all graphs of $\mathcal{SC}$*-depth* at most k has extension preservation by showing that it satisfies the requirements of Theorem 18, as an illustration that, although closure under flip-sums is a technical condition, it can be present in interesting tame dense classes.

▶ **Definition 20** ([20], Definition 3.5)**.** *We inductively define the class $\mathcal{SC}(k)$ as:*

- $\mathcal{SC}(0) = \{K_1\}$*;*
- *If $G_1, \dots, G_n \in \mathcal{SC}(k), H := G_1 + \dots + G_n$ and $X \subseteq V(H)$, then $\overline{H}^X := H \triangle_F P \in \mathcal{SC}(k+1)$ for $P_1 := X, P_2 := V(H) \setminus X$ and $F = \{(1,1)\}$, i.e. $\overline{H}^X$ is the graph obtained from H by flipping the edges within X.*

▶ **Example 21.** Fix $k \in \mathbb{N}$ and let $G \in \mathcal{SC}(k)$. Consider an $\mathcal{SC}$*-decomposition tree of* G, i.e. a labelled tree $\mathcal{T}$ of height $k+1$ whose leaves are labelled by the vertices of G, every non-leaf node is labelled by the graph $\overline{(G_1 + \dots G_n)}^X$ where G_i are the labels of its children, X is a subset of $\bigcup_{i \in [n]} V(G_i)$, and the root ρ is labelled by G. Let $f(m) = m^{k+1}$, and suppose that $|G| > f(m)$. Since $\mathcal{T}$ has height $k+1$ and its leaves correspond to the vertices of G, there must exist some vertex t of $\mathcal{T}$ with at least m children. Let $t_1 := \rho, t_2, \dots, t_\ell := t$ be the unique path from the root of $\mathcal{T}$ to t, and for each $i \in [\ell]$ let $X_i \subseteq V(G)$ be the set coming from the label of t_i. Letting P the partition of $V(G)$ into 2^ℓ parts depending on the membership of a vertex within each of $X_1, \dots, X_\ell$ and $F \subseteq [2^\ell]^2$ be the flip that corresponds to complementing each of $X_1, \dots, X_\ell$, it follows that $G \triangle_F P$ contains at least m distinct connected components. Let $\mathcal{T}'$ be the tree obtained from $\mathcal{T}$ by the following operation. We first create a copy of each subtree of $\mathcal{T}$ rooted at a child of t_ℓ and connect them to t_ℓ. The labels are naturally carried from each original subtree to the copy. If the label of t_ℓ in $\mathcal{T}$ was $\overline{(G_1 + \dots + G_m)}^X$, then its label in $\mathcal{T}'$ is $\overline{(G_1 + G'_1 + \dots + G_m + G'_m)}^{X \cup X'}$ where each G'_i corresponds to the copy of G_i, and X' corresponds to the set of copies of the vertices in X. From there, we perform the same operation for $i = \ell - 1, \dots, 1$, this time copying only the children of t_i that are not t_{i+1}. This completes the construction of $\mathcal{T}'$. It is easy to then see than the root of $\mathcal{T}'$ corresponds to the graph $G \star_{(F,P)} G$, thus witnessing that $G \star_{(F,P)} G \in \mathcal{SC}(k)$. It follows that the class $\mathcal{SC}(k)$ satisfies the requirements of Theorem 18, and thus extension preservation holds over this class.

5 Conclusion

We conclude with some questions and remarks. Firstly, it would be of independent interest to provide a characterisation of strongly flip-flat classes, akin to the characterisation of almost-wide classes via shallow minors given in [24, Theorem 3.21]. This could either be a characterisation via excluded induced subgraphs occurring in flips, in analogy to the one of monadic stability provided in [15], or in terms of *shallow vertex minors*, in analogy to the one in [8].

Moreover, it is unclear whether one can produce a formula preserved by extensions with minimal induced models of cliquewidth 3. The issue with interweaving definable orders is that one simultaneously requires for two sets to semi-induce a half-graph while (non-)adjacency is used to mark successors; this requires to keep track of at least four colours classes in a clique decomposition. We therefore leave the question of whether extension preservation holds over graphs of cliquewidth 3 open. It is also easily seen that the structures $\mathcal{H}_n$ have *twin-width* 2 (see [6] for definitions). The status of extension preservation on the class of all graphs of twin-width 1 is also unknown.

The role of orders was crucial in our construction in Section 3. In the context of undirected graphs, orders are instantiated through half-graphs. It natural to then inquire if, for every fixed $k, \ell \in \mathbb{N}$, the class of all graphs of cliquewidth at most k which omit semi-induced half-graphs of size larger than ℓ has the extension preservation property. Every such class is known to be equal to a transduction of a class of bounded treewidth by [26], and so by Proposition 13, it is strongly flip-flat. It would therefore be interesting to provide a direct combinatorial argument witnessing this, so as to be able to verify if such classes satisfy the closure requirements of Theorem 18.

References

1 Hans Adler and Isolde Adler. Interpreting nowhere dense graph classes as a classical notion of model theory. *European Journal of Combinatorics*, 36:322–330, 2014. doi:10.1016/j.ejc.2013.06.048.

2 Miklos Ajtai and Yuri Gurevich. Datalog vs first-order logic. *Journal of Computer and System Sciences*, 49(3):562–588, 1994. 30th IEEE Conference on Foundations of Computer Science. doi:10.1016/S0022-0000(05)80071-6.

3 Albert Atserias, Anuj Dawar, and Martin Grohe. Preservation under extensions on well-behaved finite structures. *SIAM Journal on Computing*, 38(4):1364–1381, 2008. doi:10.1137/060658709.

4 Albert Atserias, Anuj Dawar, and Phokion G Kolaitis. On preservation under homomorphisms and unions of conjunctive queries. *Journal of the ACM (JACM)*, 53(2):208–237, 2006. doi:10.1145/1131342.1131344.

5 John T Baldwin and Saharon Shelah. Second-order quantifiers and the complexity of theories. *Notre Dame Journal of Formal Logic*, 26(3):229–303, 1985. doi:10.1305/NDJFL/1093870870.

6 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable fo model checking. *ACM Journal of the ACM (JACM)*, 69(1):1–46, 2021. doi:10.1145/3486655.

7 Samuel Braunfeld, Anuj Dawar, Ioannis Eleftheriadis, and Aris Papadopoulos. Monadic nip in monotone classes of relational structures. In *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

8 Hector Buffière, Eun Jung Kim, and Patrice Ossona de Mendez. Shallow vertex minors, stability, and dependence. *arXiv preprint arXiv:2405.00408*, 2024.

9 Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1):77–114, 2000. doi:10.1016/S0166-218X(99)00184-5.

10 Konrad K. Dabrowski, Matthew Johnson, and Daniël Paulusma. *Clique-width for hereditary graph classes*, pages 1–56. London Mathematical Society Lecture Note Series. Cambridge University Press, 2019. doi:10.1017/9781108649094.002.

11 Peter Damaschke. Induced subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 14(4):427–435, 1990. doi:10.1002/JGT.3190140406.

12 Anuj Dawar. Finite model theory on tame classes of structures. In *International Symposium on Mathematical Foundations of Computer Science*, pages 2–12. Springer, 2007. doi:10.1007/978-3-540-74456-6_2.

13 Anuj Dawar. Homomorphism preservation on quasi-wide classes. *Journal of Computer and System Sciences*, 76(5):324–332, 2010. doi:10.1016/J.JCSS.2009.10.005.

14 Anuj Dawar and Ioannis Eleftheriadis. Preservation theorems on sparse classes revisited. *arXiv preprint arXiv:2405.10887*, 2024. doi:10.48550/arXiv.2405.10887.

15 Jan Dreier, Ioannis Eleftheriadis, Nikolas Mählmann, Rose McCarty, Michał Pilipczuk, and Szymon Toruńczyk. First-order model checking on monadically stable graph classes. *arXiv preprint arXiv:2311.18740*, 2023.

16 Jan Dreier, Nikolas Mählmann, and Sebastian Siebertz. First-order model checking on structurally sparse graph classes. In *STOC 2023*, pages 567–580. ACM, 2023. doi:10.1145/3564246.3585186.

17 Jan Dreier, Nikolas Mählmann, Sebastian Siebertz, and Szymon Toruńczyk. Indiscernibles and Flatness in Monadically Stable and Monadically NIP Classes. 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 125:1–125:18, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.125.

18 Jan Dreier, Nikolas Mählmann, and Szymon Toruńczyk. Flip-breakability: A combinatorial dichotomy for monadically dependent graph classes. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1550–1560, 2024. doi:10.1145/3618260.3649739.

19 H-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer, 2nd edition, 1999.

20 Robert Ganian, Petr Hliněný, Jaroslav Nešetřil, Jan Obdržálek, and Patrice Ossona De Mendez. Shrub-depth: Capturing height of dense graphs. *Logical Methods in Computer Science*, 15, 2019.

21 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. *J. ACM*, 64(3), June 2017. doi:10.1145/3051095.

22 Yuri Gurevich. Toward logic tailored for computational complexity. *Computation and Proof Theory*, pages 175–216, 1984.

23 Ulrich Knauer and Kolja Knauer. *Algebraic Graph Theory: Morphisms, Monoids and Matrices*. De Gruyter, Berlin, Boston, 2019. doi:doi:10.1515/9783110617368.

24 Jaroslav Nešetřil and Patrice Ossona De Mendez. First order properties on nowhere dense structures. *The Journal of Symbolic Logic*, 75(3):868–887, 2010. URL: http://www.jstor.org/stable/20799288, doi:10.2178/JSL/1278682204.

25 Jaroslav Nešetřil and Patrice Ossona De Mendez. *Sparsity: graphs, structures, and algorithms*, volume 28. Springer Science & Business Media, 2012.

26 Jaroslav Nešetřil, Patrice Ossona de Mendez, Michał Pilipczuk, Roman Rabinovich, and Sebastian Siebertz. Rankwidth meets stability. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2014–2033. SIAM, 2021.

27 Benjamin Rossman. Homomorphism preservation theorems. *Journal of the ACM (JACM)*, 55(3):1–53, 2008. doi:10.1145/1379759.1379763.

28 W. W. Tait. A counterexample to a conjecture of Scott and Suppes. *Journal of Symbolic Logic*, 24(1):15–16, 1959. doi:10.2307/2964569.

29 Szymon Toruńczyk. Flip-width: Cops and robber on dense graphs. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 663–700. IEEE, 2023.

A The proof of Lemma 5

Here we provide a proof of Lemma 5, which we now restate.

▶ **Lemma 5.** *Let $n \geq 7$ and $f : \mathcal{I}_n \to \mathcal{H}_n$ be an embedding. Then f is the inclusion map.*

This is achieved in two steps. First, we consider the subgraph $\mathcal{I}'$ of $\mathcal{H}_n$ induced on $\{v_1, v_2, v_3, u_1, u_2, u_3, a\}$. Evidently, the map $g_n : \mathcal{I}' \to \mathcal{H}_n$ sending

$$(v_1, v_2, v_3, u_1, u_2, u_3, a) \mapsto (v_{n-2}, v_{n-1}, v_n, u_{n-2}, u_{n-1}, u_n, b)$$

is an embedding. We argue that this is the only non-trivial embedding of $\mathcal{I}'$ in $\mathcal{H}_n$.

▶ **Lemma 22.** *Let $n \geq 7$ and $f : \mathcal{I}' \to \mathcal{H}_n$ be an embedding. Then f is either the inclusion map or equal to g_n.*

Proof. As before, we write $V := \{v_1, \ldots, v_n\} \subseteq V(\mathcal{H}_n)$ and $U := \{u_1, \ldots, u_n\} \subseteq V(\mathcal{H}_n)$. We shall consider the possible images of the vertex v_2. Suppose that $f(v_2) = u_i$ for some $i \in [n]$. Clearly, since the vertices v_1, v_3, a are pairwise non-adjacent, we cannot have $f[\{v_1, v_3, a\}] \subseteq U$. We hence distinguish cases.

1. Suppose that v_1, v_3, a are all mapped to vertices in V under f. Since these are non-adjacent, we must have $f[\{v_1, v_3, a\}] = \{v_m, v_r, v_\ell\}$ for some $m + 2 < r + 1 < \ell \leq i$. Now, consider $f(u_1)$; this must be some vertex in $\mathcal{H}_n$ which is adjacent to only one of v_m, v_r, v_ℓ and not adjacent to u_i. This necessarily implies that $f(u_1) = v_{i+1}$, $f(v_1) = v_\ell$ while $\ell = i$. Consider $f(u_2)$; this must be a vertex non-adjacent to v_{i+1}, and adjacent to u_i, v_i and exactly one of $\{v_m, v_r\}$. From this we deduce that $f(u_2) = v_{i-1}$, $f(a) = v_r$, and $r = i - 2$. Finally, the vertex $f(u_3)$ must be adjacent to $v_m, v_{i-2}, v_i, u_i, v_{i+1}$ and non-adjacent to v_{i-1}; obviously no such vertex exists in $\mathcal{H}_n$, and we thus obtain a contradiction.
2. Suppose that two of v_1, v_3, a are mapped to vertices in V and one is mapped to a vertex in U. In this case we must have that $f[\{v_1, v_3, a\}] = \{u_m, v_r, v_\ell\}$ for some $m + 1 < r + 1 < \ell \leq i$. Consider $f(u_1)$; this must be non-adjacent to v_i and adjacent to exactly one of u_m, v_r, v_ℓ. This further results in two distinct cases. If $f(u_1) = v_{i+1}$, then we have $f(v_1) = v_\ell$ and $\ell = i$, which leads to a contradiction with an analogous argument to the above. If $f(u_1) = u_{i-1}$, then necessarily $f(v_1) = v_r$ while $m = i - 2, r = i - 1, \ell = i$. Considering $f(u_2)$, we now see that this vertex must be non-adjacent to u_{i-1} and adjacent to u_i, v_{i-1} and exactly one of $\{u_{i-2}, v_i\}$; evidently there is no such vertex in $\mathcal{H}_n$ and we thus obtain a contradiction.
3. Suppose that exactly one of v_1, v_3, a is mapped to a vertex in V and two are mapped to vertices of U. This forces that $f[\{v_1, v_3, a\}] = \{u_{m-1}, u_m, v_\ell\}$ for some $m < \ell \leq i$ and $m + 1 < i$. Again, consider $f(u_1)$; this is non-adjacent to u_i and adjacent to exactly one of $\{u_m, u_{m+1}, v_\ell\}$. Once again this leads to two options. If $f(u_1) = v_{i+1}$, then we have $f(v_1) = v_\ell$ and $\ell = i$, which leads to a contradiction as in Case 1. On the other hand, if $f(u_1) = u_{i-1}$ then we necessarily obtain that $f(v_1) = u_{m-1}$ while $m = i - 2$ and $\ell = i$. The vertex $f(u_2) \in \mathcal{H}_n$ must then be non-adjacent to u_{i-1} and adjacent to u_{i-1}, u_i and exactly one of u_{i-2}, v_i; since there is no such vertex in $\mathcal{H}_n$ we once again obtain a contradiction.
4. Suppose that one of v_1, v_3, a is mapped to a or b under f. Since u_3 is adjacent to all of v_1, v_2, v_3, a it must necessarily be that $f(u_3) = u_m$ for some $m > i + 1$. The vertex $f(u_2)$ must then be non-adjacent to u_m, and adjacent to u_i and exactly two of $f(v_1), f(v_3), f(a)$. As no such vertex exists in this case, we obtain a contradiction.

Since the above cases lead to a contradiction, we see that $f(v_2) \notin U$. Since no v_i for $i \in [n] \setminus \{2, n-1\}$ has three neighbours which induce an independent set, this necessarily implies that $f(v_2)$ is equal to v_2 or v_{n-1}. Assume that $f(v_2) = v_2$. Again, since v_1, v_3, a share no edges, we must necessarily have $f[\{v_1, v_3, a\}] = \{v_1, v_3, a\}$. Since u_3 is adjacent to all of v_1, v_2, v_3, a we see that $f(u_3) = u_m$ for some $m \geq 3$. As u_2 is non-adjacent to u_3 and adjacent to v_2 to exactly two of v_1, v_3, a, we see that $f(u_3) = u_3$ and $f(u_2) = u_2$, which in turn ensure that f is the inclusion map. By similar reasoning, we deduce that if $f(v_2)$ is equal to v_{n-1} then $f = g_n$ as required. ◀

Proof of Lemma 5. Let $f : \mathcal{I}_n \to \mathcal{H}_n$ be an embedding. It follows by Lemma 22 that either f is the inclusion map, or it is the map given by swapping the two induced copies of $\mathcal{I}'$, i.e. the map

$$(v_1, v_2, v_3, u_1, u_2, u_3, a) \mapsto (v_{n-2}, v_{n-1}, v_n, u_{n-2}, u_{n-1}, u_n, b);$$

$$(v_{n-2}, v_{n-1}, v_n, u_{n-2}, u_{n-1}, u_n, b) \mapsto (v_1, v_2, v_3, u_1, u_2, u_3, a).$$

Since u_{n-2} is adjacent to v_2, the latter case would imply that u_1 is adjacent to v_{n-1}, which is a contradiction. Hence, f is the inclusion map as claimed. ◀

B The proof of Theorem 18

Before proceeding with Theorem 18 we introduce some relevant definitions. Fix a relational signature τ and $q, d \in \mathbb{N}$, and let A be a τ-structure. By the *(q,d)-type of some $a \in A$* we shall mean the set containing all the MSO formulas $\theta(x)$ of quantifier rank[2] at most q, up to logical equivalence, such that $N_d^A(a) \models \theta(a)$. When we speak of a (q,d)-type t over τ, without reference to a particular element in a structure, we shall mean a (q,d)-type of some element in some τ-structure. We say that an element $a \in A$ *realises a (q,d)-type t* whenever $N_d^A(a) \models \theta(a)$ for all $\theta(x) \in t$. Evidently, the number of (q,d)-types is bounded by some $p \in \mathbb{N}$ depending only on τ and q. Given a τ-structure A, a set $C \subseteq A$, and a (q,d)-type t, we say that t is *covered by C in A* if all $a \in A$ realising t satisfy $N_d^A(a) \subseteq C$. For $n \in \mathbb{N}$ we also say that t is *n-free over C in A* if there is a $2d$-independent set $S \subseteq A$ of size n such that each $a \in S$ realises t and $N_d^A(a) \cap C = \emptyset$.

▶ **Lemma 23.** *Fix a relational signature τ and $q, d \in \mathbb{N}$. Let p be the number of (q,d)-types over τ. Then for every τ-structure A and $n \in \mathbb{N}$, there exists a radius $e \leq 2dp$ and a set $D \subseteq A$ of at most $(n-1)p$ points such that each (q,d)-type is either covered by $N_e^A(D)$ or is n-free over over $N_e^A(D)$.*

Proof. Fix an enumeration $t_1, \ldots, t_p$ of all (q,d)-types over τ. We shall define D and e inductively starting at $D_0 = \emptyset$ and $e_0 = 0$. Assuming D_i and e_i have been defined, we let $C = N_{e_i}^A(D_i)$. If all types are covered by C or are n-free over C then we are done; otherwise, we let $j \in [p]$ be minimal such that t_j is neither covered by C nor n-free over C. We then define a set $E \subseteq A$ inductively, starting with $E_0 := \emptyset$ and at step $\ell + 1$ adding to E_ℓ a realisation $a \in A \setminus N_{2d}^A(C \cup E_\ell)$ of t_j if there exists one; this iteration must stop within $n-1$ steps, as otherwise t_j would be n-free over C. In particular, $|E| \leq n-1$ and t_j is covered by $N_{e_i+2d}^A(D_i \cup E)$. We subsequently let $D_{i+1} = D_i \cup E$ and $e_{i+1} = e_i + 2d$. It follows that the construction must stop within at most p steps, since at each step we cover a previously uncovered type, which in addition, remains covered for the rest of the construction. Consequently, $|D| \leq (n-1)p$ and $e \leq 2dp$ as claimed. ◀

[2] Here both first-order and second-order quantifiers contribute to the quantifier rank.

▶ **Theorem 18.** *Fix a hereditary class of graphs $\mathcal{C}$. Suppose that there is some $k \in \mathbb{N}$ such that for all $r \in \mathbb{N}$ there is a function $f_r : \mathbb{N} \to \mathbb{N}$ satisfying that for every $m \in \mathbb{N}$ and every $G \in \mathcal{C}$ of size at least $f(m)$ there is a k-partition P of $V(G)$, some k-flip F, and $A \subseteq V(G)$ such that*

1. *$|A| \geq m$;*
2. *A is r-independent in $G \triangle_F P$;*
3. *$G \star_{(F,P)} G \in \mathcal{C}$.*

Then extension preservation holds over $\mathcal{C}$.

Proof. Fix $\mathcal{C}$ as above, and let ϕ be a formula preserved by extensions overs $\mathcal{C}$. We shall obtain a bound on the size of the minimal induced models of ϕ, by arguing that any large enough model of ϕ contains a proper induced substructure which also models ϕ. We can then conclude that ϕ is equivalent to an existential formula over $\mathcal{C}$ using Lemma 2.

Letting $k \in \mathbb{N}$ be as the in the statement of Theorem 18, we consider the formula ϕ^k from Definition 15. Using Gaifman's locality theorem we rewrite ϕ^k into a boolean combination of basic local sentences, i.e. we may assume that there is some $\ell \in \mathbb{N}$ and τ_E^k-sentences ψ_i for $i \in [\ell]$ such that

$$\phi^k = \bigvee_{i \in \ell} \psi_i \text{ and } \psi_i = \bigwedge_{j \in A_i} \chi_{ij} \wedge \bigwedge_{j \in B_i} \neg \chi_{ij},$$

where each χ_{ij} is a basic local sentence. We henceforth fix the following constants:

- ρ is the maximum over all the locality radii of the χ_{ij};
- s is the sum of all widths of the χ_{ij};
- γ is the maximum over all the quantifier ranks of the χ_{ij};
- $q := \gamma + 3\rho + 3$;
- $d := 2(\rho + 1)(\ell + 1)s + 6\rho + 2$;
- p is the number of (q, d)-types over the signature τ_E^k;
- $n := (\ell + 2)s$;
- $m := (n - 1)q + s + \ell s + 1$;
- $r := 4dp + 2\rho + 1$.

Our goal is to establish that any minimal induced model of ϕ in $\mathcal{C}$ must have size less than $f_r(m)$, where f is as in the statement of Theorem 18. So, assume that some $G \models \phi$ has size at least $f_r(m)$. It follows by assumption that there is a k-partition P and a k-flip F such that $G \triangle_F P$ contains an r-independent set of size m. We henceforth work with the structure $G^* := G_{(F,P)}$, i.e. the expansion of $G \triangle_F P$ with unary predicates corresponding to the parts of P. By definition, we have that $G^* \models \phi^k$.

By Lemma 23 we obtain a radius $e \leq 2dp$ and a set $D \subseteq V(G^*)$ of at most $(n-1)p$ vertices such that each (q, d)-type in G^* is either covered by $N_e^{G^*}(D)$ or is n-free over $N_e^{G^*}(D)$; we henceforth refer to types of the former kind as *rare*, and to types of the latter kind as *frequent*.

We proceed to inductively construct increasing sequences of sets $S_0 \subseteq S_1 \subseteq \cdots \subseteq V(G^*)$, $C_0 \subseteq C_1 \subseteq \cdots \subseteq V(G^*)$, and $I_0 \subseteq I_1 \subseteq \cdots \subseteq I$ which satisfy the following conditions for every i:

1. $S_i \subseteq N_\rho^{G^*}(C_i)$;
2. $|C_i| \leq is$;
3. $|I_i| = i$;
4. no disjoint extension of $G^*[S_i]$ satisfies $\bigvee_{j \in I_i} \psi_j$;
5. $N_e^{G^*}(D)$ and $N_d^{G^*}(C_i)$ are disjoint.

Clearly, this construction must terminate within ℓ steps. Indeed, assume for a contradiction that we have constructed S_ℓ, C_ℓ, and I_ℓ satisfying conditions 1-5 above. If so, then $I_\ell = I$ while $G^* + G^*[S_\ell]$ is a disjoint extension of $G^*[S_\ell]$ which satisfies $\phi^k = \bigvee_{i\in I} \psi_i$ by Lemma 16, therefore contradicting condition 4. At the end of the construction we will obtain some $N < \ell$ and some $S_N \subsetneq V(G^*)$ satisfying $G^*[S_N] \models \phi^k$. Combining Definition 15 with Observation 17, this will imply that $G[S_N] \models \phi$, and hence that G cannot be a minimal model of ϕ as required.

Initially, we set $S_0 = C_0 = I_0 = \emptyset$. Assume that S_i, C_i, and I_i have been defined. Write $H^* := G^* + G^*[S_i]$ for the disjoint union of G^* with its substructure induced on S_i. By our closure assumptions on $\mathcal{C}$ and Lemma 16 we deduce that $H^* \models \phi^k$. In particular, there exists some $i' \in I$ such that $H^* \models \psi_{i'}$, while $i' \notin I_i$ due to property 4. We let $I_{i+1} = I_i \cup \{i'\}$ and henceforth drop the reference to the index i' as it will remain fixed for the remaining of the argument, e.g. by writing ψ and χ_j instead of $\psi_{i'}$ and $\chi_{i'j}$ respectively.

As H^* satisfies $\psi = (\bigwedge_{j\in A} \chi_j \wedge \bigwedge_{j\in B} \neg\chi_j)$, it satisfies the basic local sentences χ_j with $j \in A$. For each $j \in A$, we may thus choose a minimal set $W_j \subseteq V(H^*)$ of witnesses for the outermost existential quantifiers of the basic local sentence χ_j, and let $W := \bigcup_{j\in A} W_j$ be their union. As s is the sum of the widths of all the χ's it follows that $|W| \leq s$. We partition W into those witnesses that appear in the disjoint copy of G^*, and those that appear in the disjoint copy of $G^*[S_i]$, and write W_G and W_H for these respective parts.

Now, suppose that some $v \in W_G$ satisfies $N^{G^*}_{\rho+1}(C_i) \cap N^{G^*}_\rho(v) \neq \emptyset$; we argue that we may replace v with some witness $v' \in V(G^*)$ such that $N^{G^*}_{\rho+1}(C_i) \cap N^{G^*}_\rho(v') = \emptyset$. Indeed, we first choose some $u \in C_i$ such that $N^{G^*}_{\rho+1}(u) \cap N^{G^*}_\rho(v) \neq \emptyset$. Consequently, we have that $N^{G^*}_\rho(v) \subseteq N^{G^*}_{3\rho+1}(u) \subseteq N^{G^*}_d(u)$. Property 5 then ensures that the (q,d)-type t (in G^*) of u is frequent, and so it has $n > (\ell+1)s \geq |W \cup C_i|$ realisations whose d-neighbourhoods are pairwise disjoint and disjoint from $N^{G^*}_e(D)$. We may thus pick a realisation $u' \in V(G^*)$ of t such that $N^{G^*}_{\rho+1}(W \cup C_i) \cap N^{G^*}_{3\rho+1}(u') = \emptyset$. Let τ be the (γ,ρ)-type of v, and consider the formula

$$\theta(x) := \exists y[\forall z(\mathrm{dist}(y,z) \leq \rho \to \mathrm{dist}(x,z) \leq 3\rho+1) \wedge \bigwedge_{\eta\in\tau} \eta^{N_r(y)}(y)].$$

Clearly, the quantifier rank of θ is bounded by $3\rho + 3 + \gamma \leq q$, while $N^{G^*}_d(u) \models \theta(u)$ with v serving as the existential witness. Consequently $\theta(x)$ is in t, and as u and u' have the same (q,d)-type, it follows that $N^{G^*}_d(u') \models \theta(u')$. It follows that there is $v' \in V(G^*)$ such that $N^{G^*}_\rho(v') \subseteq N^{G^*}_{3\rho+1}(u') \subseteq N^{G^*}_d(u')$, while v and v' have the same (γ,ρ)-type. In particular, their ρ-neighbourhoods satisfy the same FO-formulas of quantifier rank $\leq \gamma$. Finally, observe that $N^{G^*}_{\rho+1}(W \cup C_i) \cap N^{G^*}_{3\rho+1}(v') = \emptyset$ and so $N^{G^*}_{\rho+1}(C_i) \cap N^{G^*}_\rho(v') = \emptyset$; we may thus replace v by v' in W_G as a witness.

After replacing all such witnesses in $_G$, we can ensure that

$$|\{v \in W_G : N^{G^*}_{\rho+1}(C_i) \cap N^{G^*}_\rho(v) \neq \emptyset\}| = 0 \qquad (\star)$$

Consider the induced substructure $U^* := G^*[N^{G^*}_e(D) \cup N^{G^*}_\rho(W_G) \cup S_i]$. We claim that U^* satisfies $\bigwedge_{j\in A} \chi_j$. Indeed, notice that $S_i \subseteq N^{G^*}_\rho(C_i)$, while $N^{G^*}_{\rho+1}(C_i)$ is disjoint from $N^{G^*}_e(D)$ by property 5 and disjoint from $N^{G^*}_\rho(W_G)$ by $(\star)$. It follows that U^* is the disjoint union of $G^*[N^{G^*}_e(D) \cup N^{G^*}_\rho(W_G)]$ and $G^*[S_i]$; thus all the witnesses from W and their ρ-neighbourhoods can be found in U^*, implying that $U^* \models \chi_j$ for all $j \in A$ as these are basic local sentences.

Now, observe that U^* is a proper induced substructure of G^*. This is because

$$|D \cup W_G \cup C_i| \leq (n-1)p + s + \ell s < m;$$

$$N_e^{G^*}(D) \cup N_\rho^{G^*}(W_G) \cup S_i \subseteq N_{2dp+\rho}^{G^*}(D \cup W_G \cup C_i) \subseteq N_{\lfloor r/2 \rfloor}^{G^*}(D \cup W_G \cup C_i),$$

and so, unlike G^*, U^* does not contain an r-independent set of size m. Consequently, if $U^* \models \phi^k$ then we set $S_N := N_e^{G^*}(D) \cup N_\rho^{G^*}(W_G) \cup S_i$ and our construction terminates.

We hereafter assume that $U^* \not\models \phi^k$, and proceed with the definition of S_{i+1} and C_{i+1}. Since $U^* \models \bigwedge_{j \in A} \chi_j$ it must be that $U^* \not\models \bigwedge_{j \in B} \neg\chi_j$. We can therefore fix some $j \in B$ such that $U^* \models \chi_j$. Suppose that

$$\chi_j = \exists x_1, \ldots, \exists x_{s'} [\bigwedge_{a \neq b} \mathrm{dist}(x_a, x_b) > 2\rho' \wedge \bigwedge_a \xi^{N_{\rho'}(x_a)}(x_a)]$$

for some $\rho' \leq \rho$, $s' \leq s$, and a formula ξ of quantifier rank $\gamma' \leq \gamma$. Fix a set $V = \{w_1, \ldots, w_{s'}\} \subseteq U^*$ of witnesses for the outermost existential quantifier of χ_j. Notice that if the (q,d)-type in G^* of every $w \in V$ was rare then $N_{\rho'}^{G^*}(V) \subseteq N_e^{G^*}(D) \subseteq V(G^*)$, implying that $G^* \models \chi_j$ and thus $H^* \models \chi_j$ as H^* is a disjoint extension of G^* and χ_j is a basic local sentence. We can thus fix some $w \in V$ whose (q,d)-type in G^*, say t_w, is frequent. As a result, there is a set $Z \subseteq V(G^*)$ of n realisations of t_w whose d-neighbourhoods are pairwise disjoint and disjoint from $N_e^{G^*}(D)$. Now, since $4\rho + 3 \leq d$, $n = (\ell+2)s$, and $|C_i| \leq \ell s$, there exists a subset $Z' \subseteq Z$ of at least s elements which additionally satisfies $N_{\rho+1}^{G^*}(C_i) \cap N_\rho^{G^*}(Z') = \emptyset$.

Consider $F := N_{\rho'}^{U^*}(w)$. Evidently, $U^*[F] = G^*[F]$ and so $G^*[F] \models \xi^{N_{\rho'}(x)}(w)$. For a set variable X consider the formula $\xi^{N_{\rho'}(x) \cap X}(x, X)$ obtained from ξ by simultaneously relativising the quantifiers of ξ to the r'-neighbourhoods of x and to the set X. Observe that the quantifier rank of $\xi^{N_{\rho'}(x) \cap X}(x, X)$ is at most $\gamma' + \rho' < q$, and moreover $G^* \models \xi^{N_{\rho'}(x) \cap X}(w, F)$. Since $\rho' < d$ it follows that the MSO formula $\exists X \xi^{N_{\rho'}(x) \cap X}(x, X)$ is in t_w. As every $\omega \in Z'$ has the same (q,d)-type in G^* as w, we may find sets $F_\omega \subseteq N_{\rho'}^{G^*}(\omega)$ for every $\omega \in Z'$ such that $G^* \models \xi^{N_{\rho'}(x) \cap X}(\omega, F_\omega)$. In particular, this implies that $G^*[F_\omega] \models \xi^{N_{\rho'}(x)}(\omega)$. We finally let:

$$C_{i+1} = C_i \cup Z'; \quad S_{i+1} = S_i \cup \bigcup_{\omega \in Z'} F_\omega.$$

We argue that these satisfy the properties 1-5. First, observe that $|C_{i+1}| = |C_i| + s \leq is + s = (i+1)s$. Moreover, as $F_\omega \subseteq N_{\rho'}^{G^*}(\omega)$, $\omega \in C_{i+1}$, and $\rho' \leq \rho$ we have $S_{i+1} \subseteq N_\rho^{G^*}(C_{i+1})$. By the fact that every $\omega \in Z'$ realises a frequent type we also have that $N_e^{G^*}(D) \cap N_d^{G^*}(C_{i+1}) = \emptyset$. It remains to argue that no disjoint extension of $G^*[S_{i+1}]$ satisfies $\bigvee_{j \in I_{i+1}} \psi_j$.

Towards this, we note that $G^*[S_{i+1}]$ is a disjoint extension of $G^*[S_i]$ by the fact that $S_i \subseteq N_\rho^{G^*}(C_i)$ and $N_{\rho+1}^{G^*}(C_i) \cap N_\rho^{G^*}(Z') = \emptyset$. Therefore, no disjoint extension of $G^*[S_{i+1}]$ satisfies ψ_j for $j \in I_i$. At the same time, every disjoint extension of $G^*[S_{i+1}]$ contains witnesses for the outermost existential quantifiers of $\chi_{i'j}$, namely the elements $\omega \in Z'$, which are pairwise at distance at least $2d > 2\rho'$ and satisfy $G^*[F_\omega] \models \xi^{N_{\rho'}(x)}(\omega)$ and $N_{\rho'}^{G^*[S_{i+1}]}(\omega) = F_\omega$, and thus $G^*[S_{i+1}] \models \xi^{N_{\rho'}(x)}(\omega)$. It follows that every disjoint extension of $G^*[S_{i+1}]$ satisfies $\chi_{i'j}$, and so it cannot satisfy $\psi_{i'}$ as needed. This complete our inductive construction of S_{i+1}, C_{i+1}, and I_{i+1}. ◀

The Parameterized Complexity of Learning Monadic Second-Order Logic

Steffen van Bergerem
Humboldt-Universität zu Berlin, Germany

Martin Grohe
RWTH Aachen University, Germany

Nina Runde
RWTH Aachen University, Germany

Abstract

Within the model-theoretic framework for supervised learning introduced by Grohe and Turán (TOCS 2004), we study the parameterized complexity of learning concepts definable in monadic second-order logic (MSO). We show that the problem of learning an MSO-definable concept from a training sequence of labeled examples is fixed-parameter tractable on graphs of bounded clique-width, and that it is hard for the parameterized complexity class para-NP on general graphs.

It turns out that an important distinction to be made is between 1-dimensional and higher-dimensional concepts, where the instances of a k-dimensional concept are k-tuples of vertices of a graph. For the higher-dimensional case, we give a learning algorithm that is fixed-parameter tractable in the size of the graph, but not in the size of the training sequence, and we give a hardness result showing that this is optimal. By comparison, in the 1-dimensional case, we obtain an algorithm that is fixed-parameter tractable in both.

2012 ACM Subject Classification Theory of computation → Logic; Theory of computation → Complexity theory and logic; Theory of computation → Fixed parameter tractability; Computing methodologies → Logical and relational learning; Computing methodologies → Supervised learning

Keywords and phrases monadic second-order definable concept learning, agnostic probably approximately correct learning, parameterized complexity, clique-width, fixed-parameter tractable, Boolean classification, supervised learning, monadic second-order logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.8

Related Version *Full Version*: `https://arxiv.org/abs/2309.10489` [10]

Funding *Steffen van Bergerem*: This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 431183758 (gefördert durch die Deutsche Forschungsgemeinschaft (DFG) – Projektnummer 431183758).
Martin Grohe: Funded by the European Union (ERC, SymSim, 101054974). 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.
Nina Runde: This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 453349072 (gefördert durch die Deutsche Forschungsgemeinschaft (DFG) – Projektnummer 453349072).

1 Introduction

We study abstract machine-learning problems in a logical framework with a declarative view on learning, where the (logical) specification of concepts is separated from the choice of specific machine-learning models and algorithms (such as neural networks). Here we are concerned with the computational complexity of learning problems in this logical learning framework, that is, the *descriptive complexity of learning* [8].

Specifically, we consider Boolean classification problems that can be specified in monadic second-order logic (MSO). The input elements for the classification task come from a set $\mathbb{X}$, the *instance space*. A *classifier* on $\mathbb{X}$ is a function $c\colon \mathbb{X} \to \{+,-\}$. Given a *training sequence* S of labeled examples $(x,\lambda) \in \mathbb{X} \times \{+,-\}$, we want to find a classifier, called a *hypothesis*, that explains the labels given in S and that can also be used to predict the labels of elements from $\mathbb{X}$ not given as examples. In the logical setting, the instance space $\mathbb{X}$ is a set of tuples from a (relational) structure, called the *background structure*, and classifiers are described by formulas of some logic, in our case MSO, using parameters from the background structure. This model-theoretic learning framework was introduced by Grohe and Turán [33] and further studied in [30, 32, 31, 7, 9, 11, 8].

We study these problems within the following well-known settings from computational learning theory. In the *consistent-learning* model, the examples are assumed to be generated using an unknown classifier, the *target concept*, from a known *concept class*. The task is to find a hypothesis that is consistent with the training sequence S, i.e. a function $h\colon \mathbb{X} \to \{+,-\}$ such that $h(x) = \lambda$ for all $(x,\lambda) \in S$. In Haussler's model of *agnostic probably approximately correct (PAC) learning* [35], a generalization of Valiant's *PAC learning* model [50], an (unknown) probability distribution $\mathscr{D}$ on $\mathbb{X} \times \{+,-\}$ is assumed, and training examples are drawn independently from this distribution. The goal is to find a hypothesis that generalizes well, i.e. one is interested in algorithms that return with high probability a hypothesis with a small expected error on new instances drawn from the same distribution. For more background on PAC learning, we refer to [37, 41, 46]. In both settings, we require our algorithms to return a hypothesis from a predefined *hypothesis class*.

Our Contributions

In this paper, we study the parameterized complexity of the consistent-learning problem MSO-Consistent-Learn and the PAC-learning problem MSO-PAC-Learn. In both problems, we are given a graph G (the background structure) and a sequence of labeled training examples of the form $(\bar{v},\lambda)$, where $\bar{v}$ is a k-tuple of vertices from G and $\lambda \in \{+,-\}$. The goal is to find a hypothesis of the form $h_{\varphi,\bar{w}}$ for an MSO formula $\varphi(\bar{x};\bar{y})$ and a tuple $\bar{w}$ with $h_{\varphi,\bar{w}}(\bar{v}) := +$ if $G \models \varphi(\bar{v};\bar{w})$ and $h_{\varphi,\bar{w}}(\bar{v}) := -$ otherwise. For MSO-Consistent-Learn, this hypothesis should be consistent with the given training examples. For MSO-PAC-Learn, the hypothesis should generalize well. We restrict the complexity of allowed hypotheses by giving a bound q on the quantifier rank of φ and a bound ℓ on the length of $\bar{w}$. Both q and ℓ as well as the dimension k of the problem, that is, the length of the tuples to classify, are part of the parameterization of the problems. A detailed description of MSO-Consistent-Learn is given in Section 3. The problem MSO-PAC-Learn is formally introduced in Section 5.

▶ **Example 1.1.** Assume we are given the graph G depicted in Figure 1, the training sequence $S = ((v_1,+),(v_3,+),(v_4,-),(v_5,-))$, and $k = 1$, $\ell = 1$, $q = 3$. Note that $k = 1$ indicates that the instances are vertices of the input graph G. Furthermore, $\ell = 1$ indicates that the specification may involve one vertex of the input graph as a parameter. Finally, $q = 3$ indicates that the formula specifying the hypothesis must have quantifier rank at most 3.

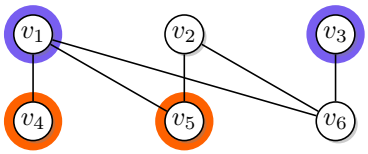


Figure 1 Graph G for Example 1.1. Positive examples are shown in purple, and negative examples are shown in orange.

Our choice of a hypothesis $h\colon V(G) \to \{+,-\}$ consistent with S says that "there is a bipartite partition of the graph such that all positive instances x are on the same side as v_2 and all negative examples are on the other side." This hypothesis can be formally specified in MSO as $h_{\varphi,\bar{w}}$ for the MSO formula $\varphi(x;y) = \exists Z\big(\psi_{\text{bipartite}}(Z) \wedge Z(x) \wedge Z(y)\big)$ and parameter setting $\bar{w} = (v_2)$, where $\psi_{\text{bipartite}}(Z) = \forall z_1 \forall z_2 \Big(E(z_1,z_2) \to \neg\big(Z(z_1) \leftrightarrow Z(z_2)\big)\Big)$.

For the 1-dimensional case of MSO-Consistent-Learn, called 1D-MSO-Consistent-Learn, [31, 30] gave algorithms that are sublinear in the background structures after a linear-time pre-processing stage for the case that the background structure is a string or a tree. This directly implies that 1D-MSO-Consistent-Learn can be solved in time $f(\ell,q)\cdot n$ for some function f, that is, in fixed-parameter linear time, if the background structure is a string or a tree. Here n is the size of the background structure and ℓ, q are the parameters of the learning problem described above. We generalize the results to labeled graphs of bounded clique-width. Graphs of clique-width c can be described by a *c-expression*, that is, an expression in a certain graph grammar that only uses c labels (see Section 2.1 for details). In our algorithmic results for graphs of bounded clique-width, we always assume that the graphs are given in the form of a c-expression. We treat c as just another parameter of our algorithms. By the results of Oum and Seymour [44], we can always compute a $2^{\mathcal{O}(c)}$-expression for a graph of clique-width c by a fixed-parameter tractable algorithm.

▶ **Theorem 1.2.** *Let $\mathscr{C}$ be a class of labeled graphs of bounded clique-width. Then* 1D-MSO-Consistent-Learn *is fixed-parameter linear on $\mathscr{C}$.*

Since graphs of bounded tree-width also have bounded clique-width, our result directly implies fixed-parameter linearity on graph classes of bounded tree-width.

Our proof for Theorem 1.2 relies on the model-checking techniques due to Courcelle, Makowsky, and Rotics for graph classes of bounded clique-width [23]. To make use of them, we encode the training examples into the graph as new labels. While this construction works for $k = 1$, it fails for higher dimensions if there are too many examples to encode.

As far as we are aware, all previous results for learning MSO formulas are restricted to the one-dimensional case of the problem. We give the first results for $k > 1$, presenting two different approaches that yield tractability results in higher dimensions.

As we discuss in Section 5, for the PAC-learning problem MSO-PAC-Learn in higher dimensions, we can restrict the number of examples to consider to a constant. In this way, we obtain fixed-parameter tractability results for learning MSO-definable concepts in higher dimensions, similar to results for first-order logic on nowhere dense classes [9, 8].

▶ **Theorem 1.3.** *Let $\mathscr{C}$ be a class of labeled graphs of bounded clique-width. Then* MSO-PAC-Learn *is fixed-parameter linear on $\mathscr{C}$.*

In the second approach to higher-dimensional tractability, and as the main result of this paper, we show in Section 6 that a consistent hypothesis can be learned on graphs of bounded clique-width with a quadratic running time in terms of the size of the graph.

▶ **Theorem 1.4.** *There is a function $g\colon \mathbb{N}^5 \to \mathbb{N}$ such that, for a Λ-labeled graph G of clique-width $\mathsf{cw}(G) \leq c$ and a training sequence S of size $|S| = m$, the problem* MSO-Consistent-Learn *can be solved in time*

$$\mathcal{O}\big((m+1)^{g(c,|\Lambda|,q,k,\ell)}|V(G)|^2\big).$$

While this is not strictly a fixed-parameter tractability result, since we usually do not consider m to be part of the parameterization, we show in Section 7 that this bound is optimal. Technically, this result is much more challenging than Theorems 1.3 and 1.2. While we still use an overall dynamic-programming strategy that involves computing MSO types, here we need to consider MSO types over sequences of tuples. The number of such sequence types is not constantly bounded, but exponential in the length of the sequence. The core of our argument is to prove that the number of relevant types can be polynomially bounded. This fundamentally distinguishes our approach from typical MSO/automata arguments, where types are from a bounded set (and they correspond to the states of a finite automaton).

Lastly, we study MSO-Consistent-Learn on arbitrary classes of labeled graphs. Analogously to the hardness of learning FO-definable concepts and the relation to the FO-model-checking problem discussed in [9], we are interested specifically in the relation of MSO-Consistent-Learn to the MSO-model-checking problem MSO-Mc. We show that MSO-Mc can already be reduced to the 1-dimensional case of MSO-Consistent-Learn, even with a training sequence of size two. This yields the following hardness result that we prove in Section 4.

▶ **Theorem 1.5.** 1D-MSO-Consistent-Learn *is para-NP-hard under fpt Turing reductions.*

Related Work

The model-theoretic learning framework studied in this paper was introduced in [33]. There, the authors give information-theoretic learnability results for hypothesis classes that can be defined using first-order and monadic second-order logic on restricted classes of structures.

Algorithmic aspects of the framework were first studied in [32], where it was proved that concepts definable in first-order logic can be learned in time polynomial in the degree of the background structure and the number of labeled examples the algorithm receives as input, independently of the size of the background structure. This was generalized to first-order logic with counting [7] and with weight aggregation [11]. On structures of polylogarithmic degree, the results yield learning algorithms running in time sublinear in the size of the background structure. It was shown in [31, 7] that sublinear-time learning is no longer possible if the degree is unrestricted. To address this issue, in [31], it was proposed to introduce a preprocessing phase where, before seeing any labeled examples, the background structure is converted to a data structure that supports sublinear-time learning later. This model was applied to monadic second-order logic on strings [31] and trees [30].

The parameterized complexity of learning first-order logic was first studied in [9]. Via a reduction from the model-checking problem, the authors show that on arbitrary relational structures, learning hypotheses definable in FO is AW[$*$]-hard. In contrast to this, they show that the problem is fixed-parameter tractable on nowhere dense graph classes. This result has been extended to nowhere dense structure classes in [8]. Although not stated as fpt results, the results in [31, 30] yield fixed-parameter tractability for learning MSO-definable concepts on strings and trees if the problem is restricted to the 1-dimensional case where the tuples to classify are single vertices.

The logical learning framework is related to, but different from the framework of *inductive logic programming* (see, e. g., [21, 42, 43]), which may be viewed as the classical logic-learning framework. In the database literature, there are various approaches to learning queries from examples [6, 5, 34, 36, 38, 13, 49, 1, 2, 14, 48, 17]. Many of these are concerned with active learning scenarios, whereas we are in a statistical learning setting. Moreover, most of the results are concerned with conjunctive queries or queries outside the relational database model, whereas we focus on monadic second-order logic. Another related subject in the database literature is the problem of learning schema mappings from examples [3, 15, 18, 19, 29]. In formal verification, related logical learning frameworks [20, 25, 28, 40, 52] have been studied as well. In algorithmic learning theory, related works study the parameterized complexity of several learning problems [4, 39] including, quite recently, learning propositional CNF and DNF formulas and learning solutions to graph problems in the PAC setting [16].

2 Preliminaries

We let $\mathbb{N}$ denote the set of non-negative integers. For $m, n \in \mathbb{N}$, we let $[m, n] \coloneqq \{\ell \in \mathbb{N} \mid m \leq \ell \leq n\}$ and $[n] \coloneqq [1, n]$. For a set V, we let $2^V \coloneqq \{V' \mid V' \subseteq V\}$.

2.1 Clique-Width

In this paper, graphs are always undirected and simple (no loops or parallel edges); we view them as $\{E\}$-structures for a binary relation symbol E, and we denote the set of vertices of a graph G by $V(G)$. A *label set* is a set Λ of unary relation symbols, and a Λ-*graph* or Λ-*labeled graph* is the expansion of a graph to the vocabulary $\{E\} \cup \Lambda$. A *labeled graph* is a Λ-graph for any label set Λ.

In the following, we define *expressions* to represent labeled graphs. A *base graph* is a labeled graph of order 1. For every base graph G, we introduce a *base expression* β that *represents* G. Moreover, we have the following operations.

Disjoint union: For disjoint Λ-graphs G_1, G_2, we define $G_1 \uplus G_2$ to be the union of G_1 and G_2. If G_1 and G_2 are not disjoint, then $G_1 \uplus G_2$ is undefined.

Adding edges: For a Λ-graph G and unary relation symbols $P, Q \in \Lambda$ with $P \neq Q$, we let $\eta_{P,Q}(G)$ be the Λ-graph obtained from G by adding an edge between every pair of distinct vertices $v \in P(G)$, $w \in Q(G)$. That is, $E\big(\eta_{P,Q}(G)\big) \coloneqq E(G) \cup \big\{(v, w), (w, v) \bigm| v \in P(G), w \in Q(G), v \neq w\big\}$.

Relabeling: For a Λ-graph G and unary relation symbols $P, Q \in \Lambda$ with $P \neq Q$, we let $\rho_{P,Q}(G)$ be the Λ-graph obtained from G by relabeling all vertices in P by Q, that is, $V(\rho_{P,Q}(G)) \coloneqq V(G)$, $P(\rho_{P,Q}(G)) \coloneqq \emptyset$, $Q(\rho_{P,Q}(G)) \coloneqq Q(G) \cup P(G)$, and $R(\rho_{P,Q}(G)) \coloneqq R(G)$ for all $R \in \Lambda \setminus \{P, Q\}$.

Deleting labels: For a Λ-graph G and a unary relation symbol $P \in \Lambda$, we let $\delta_P(G)$ be the restriction of G to $\Lambda \setminus \{P\}$, that is, the $(\Lambda \setminus \{P\})$-graph obtained from G by removing the relation $P(G)$.

We also introduce a modification of the disjoint-union operator, namely the *ordered-disjoint-union operator* $\uplus^<$, which is used in Section 6 to simplify notations.

Ordered disjoint union: To introduce this operator, we need two distinguished unary relation symbols $P_1^<$ and $P_2^<$. For disjoint Λ-graphs G_1, G_2, where we assume $P_1^<, P_2^< \notin \Lambda$, we let $G_1 \uplus^< G_2$ be the $(\Lambda \cup \{P_1^<, P_2^<\})$-expansion of the disjoint union $G_1 \uplus G_2$ with $P_1^<(G_1 \uplus^< G_2) \coloneqq V(G_1)$ and $P_2^<(G_1 \uplus^< G_2) \coloneqq V(G_2)$. By deleting the relations $P_1^<, P_2^<$ immediately after introducing them in an ordered disjoint union, we can simulate a standard disjoint-union by an ordered disjoint union, that is, $G_1 \uplus G_2 = \delta_{P_1^<}(\delta_{P_2^<}(G_1 \uplus^< G_2))$.

A Λ-*expression* is a term formed from base expressions β, whose label set is a subset of Λ, using unary operators $\eta_{P,Q}$, $\rho_{P,Q}$, δ_P for $P, Q \in \Lambda$ with $P \neq Q$, and the binary operator $\uplus$. We require Λ-expressions to be well-formed, that is, all base expressions represent base graphs with mutually distinct vertices, and the label sets fit the operators.

Every Λ-expression Ξ *describes* a Λ'-graph G_Ξ for some $\Lambda' \subseteq \Lambda$. Note that there is a one-to-one correspondence between the base expressions in Ξ and the vertices of G_Ξ. Actually, we may simply identify the vertices of G_Ξ with the base expressions in Ξ. We let $V_\Xi \coloneqq V(G_\Xi)$ be the set of these base expressions. We may then view an expression Ξ as a tree where V_Ξ is the set of leaves of this tree. We let $|\Xi|$ be the number of nodes of the tree. We have $|G_\Xi| = |V_\Xi| \leq |\Xi|$. In general, we cannot bound $|\Xi|$ in terms of $|G_\Xi|$, but for every Λ-expression Ξ, we can find a Λ-expression Ξ' such that $G_{\Xi'} = G_\Xi$ and $|\Xi'| \in \mathcal{O}(\left|\Lambda^2\right| \cdot |G_\Xi|)$.

Each subexpression Ξ' of Ξ describes a labeled graph $G_{\Xi'}$ on a subset $V_{\Xi'} \subseteq V_\Xi$ consisting of all base expressions in Ξ'. Note that, in general, $G_{\Xi'}$ is not a subgraph of G_Ξ.

For $c \in \mathbb{N}$, a *c-expression* is a Λ-expression for a label set Λ of size $|\Lambda| = c$. It is easy to see that every labeled graph of order n is described by an n-expression. The *clique-width* $\mathsf{cw}(G)$ of a (labeled) graph G is the least c such that G is described by a c-expression.

We remark that our notion of clique-width differs slightly from the one given by Courcelle and Olariu [24], since we allow vertices to have multiple labels, and we also allow the deletion of labels. Thus, our definition is similar to the definition of *multi-clique-width* [27]. However, for our algorithmic results, the definitions are equivalent, since we have $\mathsf{cw}(G) \leq \mathsf{cw}'(G)$ and $\mathsf{cw}'(G) \in 2^{\mathcal{O}(\mathsf{cw}(G))}$ for every (labeled) graph G, where cw' is the notion of clique-width from [24].

▶ **Lemma 2.1** ([44]). *For a graph G with n vertices and clique-width $c' \coloneqq \mathsf{cw}(G)$, there is an algorithm that outputs a c-expression for G where $c = 2^{3c'+2} - 1$. The algorithm has a running time of $\mathcal{O}(n^9 \log n)$.*

2.2 Monadic Second-Order Logic

We consider monadic second-order (MSO) logic, which is a fragment of second-order logic where we only quantify over unary relations (sets). In MSO, we consider two kinds of free variables, which we call set variables (uppercase X, Y, X_i) and individual variables (lowercase x, y, x_i). The *quantifier rank* $\mathrm{qr}(\varphi)$ of a formula φ is the nesting depth of its quantifiers.

Let τ be a relational vocabulary and $q \in \mathbb{N}$. By $\mathsf{MSO}(\tau, q)$, we denote the set of all MSO formulas of quantifier rank at most q using only relation symbols in τ, and we let $\mathsf{MSO}(\tau) \coloneqq \bigcup_q \mathsf{MSO}(\tau, q)$. By $\mathsf{MSO}(\tau, q, k, s)$, we denote the set of all $\mathsf{MSO}(\tau, q)$ formulas with free individual variables in $\{x_1, \ldots, x_k\}$ and free set variables in $\{X_1, \ldots, X_s\}$. In particular, $\mathsf{MSO}(\tau, q, 0, 0)$ denotes the set of *sentences*. Moreover, it will be convenient to separate the free individual variables into *instance variables* $(x_1, x_2, \ldots)$ and *parameter variables* $(y_1, y_2, \ldots)$. For this, we let $\mathsf{MSO}(\tau, q, k, \ell, s)$ denote the set of all $\mathsf{MSO}(\tau, q)$ formulas with free instance variables in $\{x_1, \ldots, x_k\}$, free parameter variables in $\{y_1, \ldots, y_\ell\}$, and free set variables in $\{X_1, \ldots, X_s\}$. Furthermore, we write $\varphi(\bar{x}, \bar{y}, \bar{X})$ to denote that the formula φ has its free instance variables among the entries of $\bar{x}$, its free parameter variables among the entries $\bar{y}$, and its free set variables among the entries $\bar{X}$.

We normalize formulas such that the set of normalized formulas in $\mathsf{MSO}(\tau, q, k, \ell, s)$ is finite, and there is an algorithm that, given an arbitrary formula in $\mathsf{MSO}(\tau, q, k, \ell, s)$, decides if the formula is normalized, and if not, computes an equivalent normalized formula. In the following, we assume that all formulas are normalized.

In this paper, all structures we consider will be labeled graphs for some label set Λ. In notations such as $\mathsf{MSO}(\tau, \dots)$, it will be convenient to write $\mathsf{MSO}(\Lambda, \dots)$ if $\tau = \{E\} \cup \Lambda$. For a Λ-labeled graph G and a tuple $\bar{v} \in (V(G))^k$, the *q-type of $\bar{v}$ in G* is the set $\mathsf{tp}_q^G(\bar{v})$ of all formulas $\varphi(\bar{x}) \in \mathsf{MSO}(\Lambda, q, k, 0)$ such that $G \models \varphi(\bar{v})$.

2.3 VC Dimension

For $q, k, \ell \in \mathbb{N}$, a formula $\varphi(\bar{x}, \bar{y}) \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$, a Λ-labeled graph G, and a tuple $\bar{w} \in (V(G))^\ell$, we let

$$\varphi(G, \bar{w}) \coloneqq \{\bar{v} \in (V(G))^k \mid G \models \varphi(\bar{v}, \bar{w})\}.$$

For a set $X \subseteq (V(G))^k$, we let

$$H_\varphi(G, X) \coloneqq \{X \cap \varphi(G, \bar{w}) \mid \bar{w} \in (V(G))^\ell\}.$$

We say that X is *shattered* by φ if $H_\varphi(G, X) = 2^X$. The *VC dimension* $\mathrm{VC}(\varphi, G)$ of φ on G is the maximum $d \in \mathbb{N}$ such that there is a set $X \subseteq V(G)^k$ of cardinality $|X| = d$ that is shattered by φ. In this paper, we are only interested in finite graphs, but for infinite G, we let $\mathrm{VC}(\varphi, G) \coloneqq \infty$ if the maximum does not exist. For a class $\mathscr{C}$ of Λ-labeled graphs, the VC dimension of φ over $\mathscr{C}$, $\mathrm{VC}(\varphi, \mathscr{C})$, is the least d such that $\mathrm{VC}(\varphi, G) \leq d$ for all $G \in \mathscr{C}$ if such a d exists, and ∞ otherwise.

▶ **Lemma 2.2** ([33, Theorem 17]). *There is a function $g\colon \mathbb{N}^5 \to \mathbb{N}$ such that the following holds. Let Λ be a label set, let $\mathscr{C}$ be the class of all Λ-graphs of clique-width at most c, and let $q, k, \ell \in \mathbb{N}$. Then $\mathrm{VC}(\varphi, \mathscr{C}) \leq g(c, |\Lambda|, q, k, \ell)$ for all $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$.*

2.4 Parameterized Complexity

A *parameterization* κ is a function mapping the input x of a problem to a natural number $\kappa(x) \in \mathbb{N}$. An algorithm $\mathbb{A}$ is an *fpt algorithm with respect to κ* if there is a computable function $f\colon \mathbb{N} \to \mathbb{N}$ and a polynomial p such that for every input x the running time of $\mathbb{A}$ is at most $f(\kappa(x)) \cdot p(|x|)$.

A *parameterized problem* is a tuple (Q, κ). We say $(Q, \kappa) \in \mathsf{FPT}$ or (Q, κ) is *fixed-parameter tractable* if there is an fpt algorithm with respect to κ for Q, and we say (Q, κ) is *fixed-parameter linear* if the polynomial in the running time of the fpt algorithm is linear. We say $(Q, \kappa) \in$ para-NP if there is a nondeterministic fpt algorithm with respect to κ for Q. If the parameterization is clear from the context, then we omit it.

For two parameterized problems $(Q, \kappa), (Q', \kappa')$, an *fpt Turing reduction* from (Q, κ) to (Q', κ') is an algorithm $\mathbb{A}$ with oracle access to Q' such that $\mathbb{A}$ decides Q, $\mathbb{A}$ is an fpt algorithm with respect to κ, and there is a computable function $g\colon \mathbb{N} \to \mathbb{N}$ such that on input x, $\kappa'(x') \leq g\big(\kappa(x)\big)$ for all oracle queries with oracle input x'.

For additional background on parameterized complexity, we refer to [26].

3 Tractability for One-Dimensional Training Data on Well-Behaved Classes

We start by formalizing the parameterized version of the problem MSO-Consistent-Learn described in the introduction. For a training sequence S, a graph G, and a hypothesis $h_{\varphi,\bar{w}}$, we say *$h_{\varphi,\bar{w}}$ is consistent with S on G* if for every positive example $(\bar{v}, +) \in S$, we have $G \models \varphi(\bar{v}, \bar{w})$, and for every negative example $(\bar{v}, -) \in S$, we have $G \not\models \varphi(\bar{v}, \bar{w})$.

MSO-Consistent-Learn

Instance: Λ-labeled graph G, $q, k, \ell \in \mathbb{N}$, training sequence $S \in (V(G)^k \times \{+,-\})^m$
Parameter: $\kappa := |\Lambda| + q + k + \ell$
Problem: Return a hypothesis $h_{\varphi,\bar{w}}$ consisting of
- a formula $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$ and
- a parameter setting $\bar{w} \in V(G)^\ell$

such that $h_{\varphi,\bar{w}}$ is consistent with the training sequence S on G, if such a hypothesis exists. Reject if there is no consistent hypothesis.

The problem 1D-MSO-Consistent-Learn refers to the 1-dimensional version of the problem MSO-Consistent-Learn where the arity k of the training examples is 1. The tractability results for 1D-MSO-Consistent-Learn are significantly more straightforward than those for the higher-dimensional problem. This is due to the fact that the full training sequence can be encoded into the graph by only adding two new labels, and a parameter setting can be encoded with ℓ more new labels.

As discussed in Section 2.2, there is a function $f\colon \mathbb{N}^4 \to \mathbb{N}$ such that $|\mathsf{MSO}(\Lambda, q, k, \ell, 0)| \leq f(|\Lambda|, q, k, \ell)$. Therefore, to solve 1D-MSO-Consistent-Learn, we can iterate over all formulas $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$ and focus on finding a parameter setting $\bar{w} \in V(G)^\ell$ such that $h_{\varphi,\bar{w}}$ is consistent with S on G. Moreover, if the model-checking problem on a graph class with additional labels is tractable, then finding a consistent parameter setting is tractable as well by performing model checking on the graph with the encoded training sequence.

▶ **Lemma 3.1.** *Let $\mathscr{C}$ be a class of labeled graphs, let $\mathscr{C}_i$ be the class of all extensions of graphs from $\mathscr{C}$ by i additional labels for all $i \in \mathbb{N}$, let $f\colon \mathbb{N} \to \mathbb{N}$ be a function, and let $c \in \mathbb{N}$ such that the* MSO*-model-checking problem on $\mathscr{C}_i$ can be solved in time $f(|\varphi|) \cdot |V(G)|^c$ for all $i \in \mathbb{N}$, where φ is the* MSO *sentence and $G \in \mathscr{C}_i$ is the labeled graph given as input. There is a function $g\colon \mathbb{N}^3 \to \mathbb{N}$ such that* 1D-MSO-Consistent-Learn *can be solved on $\mathscr{C}$ in time $g(|\Lambda|, q, \ell) \cdot |V(G)|^{c+1}$.*

The formal proof of this result can be found in the full version [10]. If the input graph is given in the form of a c-expression, then the MSO model-checking problem is fixed-parameter linear on classes of bounded clique-width [23]. Therefore, Lemma 3.1 implies that there is a function $g\colon \mathbb{N}^4 \to \mathbb{N}$ such that 1D-MSO-Consistent-Learn can be solved in time $g(\mathsf{cw}(G), |\Lambda|, q, \ell) \cdot |G|^2$.

Theorem 1.2 improves this bound for classes of graphs of bounded clique-width even further, showing that the problem 1D-MSO-Consistent-Learn can be solved in time linear in the size of the graph. This can be done by again encoding the training sequence into the graph, but then extracting a consistent parameter setting directly, following techniques similar to the ones used by Courcelle and Seese for the corresponding model-checking problem on graphs of bounded clique-width. The full proof of Theorem 1.2 can be found in [10].

Since graphs of tree-width c_t have a clique-width of at most $3 \cdot 2^{c_t - 1}$ [22], Theorem 1.2 implies that for classes of graphs of bounded tree-width, 1D-MSO-Consistent-Learn is fixed-parameter linear as well. Moreover, although all background structures we consider in this paper are labeled graphs, we remark that the result for classes of bounded tree-width also holds on arbitrary relational structures and a corresponding version of 1D-MSO-Consistent-Learn.

4 Hardness for One-Dimensional Training Data

Previously, we restricted the input graph of the MSO-learning problem to certain well-behaved classes. Now, we consider the problem MSO-Consistent-Learn without any restrictions. Van Bergerem, Grohe, and Ritzert showed in [9] that there is a close relation between first-order model checking (FO-Mc) and learning first-order formulas. The fpt-reduction in [9] from model checking to learning yields AW[∗]-hardness for learning first-order formulas on classes of structures that are not nowhere dense. It is simple to show (and not surprising) that MSO-Consistent-Learn is at least as hard as FO-Mc. The more interesting question is whether MSO-Consistent-Learn is at least as hard as the model-checking problem for MSO sentences (MSO-Mc), which is defined as follows.

MSO-Mc
Instance: Λ-labeled graph G, $\mathsf{MSO}(\Lambda)$ sentence φ **Parameter**: $\lvert\varphi\rvert$ **Problem**: Decide whether $G \models \varphi$ holds.

We give a positive answer, which even holds for the MSO-learning problem with only one-dimensional training data where we restrict the training sequence to contain at most two training examples.

▶ **Lemma 4.1.** *The model-checking problem* MSO-Mc *is fpt Turing reducible to* 1D-MSO-Consistent-Learn *where we restrict the training sequence S given as input to have length at most* 2*.*

MSO-Mc is para-NP-hard under fpt Turing reductions as even for some fixed sentence φ, the corresponding model-checking problem can be NP-hard (for example for a formula defining 3-Colorability, see [26] for details). Hence, Lemma 4.1 proves Theorem 1.5. We give a proof sketch for Lemma 4.1. The full proof can be found in [10].

Proof sketch of Lemma 4.1. We describe an fpt algorithm solving MSO-Mc using access to a 1D-MSO-Consistent-Learn oracle. Let G be a Λ-labeled graph, and let φ be an $\mathsf{MSO}(\Lambda)$ sentence. We decide whether $G \models \varphi$ holds recursively by decomposing the input formula. While handling negation and Boolean connectives is easy, the crucial part of the computation is handling quantification. Thus, we assume that $\varphi = \exists x\psi$ or $\varphi = \exists X\psi$ for some MSO formula ψ. For both types of quantifiers, we use the 1D-MSO-Consistent-Learn oracle to identify a small set of candidate vertices or sets such that ψ holds for any vertex or set if and only if it holds for any of the identified candidates. Then, since the number of candidates will only depend on $\lvert\psi\rvert$, we can check recursively whether ψ holds for any of them and thereby decide MSO-Mc with an fpt algorithm.

More specifically, using the 1D-MSO-Consistent-Learn oracle, we partition the vertices and sets based on their $\mathrm{qr}(\psi)$-type, and we only check one candidate for each class of the partition. Intuitively, for every pair of vertices, we call the oracle with one vertex as a positive example and the other vertex as a negative example. The oracle returns a hypothesis if and only if the types of the two vertices differ. When partitioning the sets of vertices, we encode the two sets to check into the graph before calling the oracle. Moreover, instead of calling the oracle for every pair of sets (which would lead to a running time that is exponential in the size of the graph), we group the sets based on their size, start by finding a small family of candidate sets of size 1, and we use these candidates iteratively to find a small family of sets with one additional vertex. With this technique, the number of oracle calls is only quadratic in the size of the graph. ◀

5 PAC Learning in Higher Dimensions

So far, we considered the consistent-learning setting, where the goal is to return a hypothesis that is consistent with the given examples. In this section, we study the MSO-learning problem in the agnostic PAC-learning setting. There, for an instance space $\mathbb{X}$, we assume an (unknown) probability distribution $\mathscr{D}$ on $\mathbb{X} \times \{+,-\}$. The learner's goal is to find a hypothesis $h\colon \mathbb{X} \to \{+,-\}$, using an oracle to draw training examples randomly from $\mathscr{D}$, such that h (approximately) minimizes the generalization error

$$\operatorname{err}_{\mathscr{D}}(h) \coloneqq \Pr_{(x,\lambda)\sim\mathscr{D}}\big(h(x) \neq \lambda\big).$$

For every Λ-labeled graph G and $q, k, \ell \in \mathbb{N}$, let $\mathscr{H}_{q,k,\ell}(G)$ be the hypothesis class

$$\mathscr{H}_{q,k,\ell}(G) \coloneqq \big\{h_{\varphi,\bar{w}} \;\big|\; \varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0), \bar{w} \in (V(G))^{\ell}\big\}.$$

Formally, we define the MSO PAC-learning problem as follows.

MSO-PAC-Learn

Instance: Λ-labeled graph G, numbers $k, \ell, q \in \mathbb{N}$, $\delta, \varepsilon \in (0,1)$, oracle access to probability distribution $\mathscr{D}$ on $(V(G))^k \times \{+,-\}$
Parameter: $\kappa \coloneqq |\Lambda| + k + \ell + q + 1/\delta + 1/\varepsilon$
Problem: Return a hypothesis $h_{\varphi,\bar{w}} \in \mathscr{H}_{q,k,\ell}(G)$ such that, with probability of at least $1-\delta$ over the choice of examples drawn i.i.d. from $\mathscr{D}$, it holds that

$$\operatorname{err}_{\mathscr{D}}(h_{\varphi,\bar{w}}) \leq \min_{h\in\mathscr{H}_{q,k,\ell}(G)} \operatorname{err}_{\mathscr{D}}(h) + \varepsilon.$$

The remainder of this section is dedicated to the proof of Theorem 1.3, that is, we want to show that MSO-PAC-Learn is fixed-parameter linear on classes of bounded clique-width when the input graph is given as a c-expression. To solve the problem algorithmically, we can follow the *Empirical Risk Minimization (ERM)* rule [51, 46], that is, our algorithm should minimize the *training error* (or *empirical risk*)

$$\operatorname{err}_S(h) \coloneqq \frac{1}{|S|} \cdot |\{(\bar{v},\lambda) \in S \mid h(\bar{v}) \neq \lambda\}|$$

on the training sequence S of queried examples. Roughly speaking, an algorithm can solve MSO-PAC-Learn by querying a sufficient number of examples and then following the ERM rule. To bound the number of needed examples, we combine a fundamental result of statistical learning [12, 46], which bounds the number of needed examples in terms of the VC dimension of a hypothesis class, with Lemma 2.2, a result due to Grohe and Turán [33], which bounds the VC dimension of MSO-definable hypothesis classes on graphs of bounded clique-width. Together, they imply the following result. See the full version [10] for details.

▶ **Lemma 5.1.** *There is a computable function $m\colon \mathbb{N}^5 \times (0,1)^2 \to \mathbb{N}$ such that any algorithm that proceeds as follows solves the problem* MSO-PAC-Learn. *Given a Λ-labeled graph G of clique-width at most c, numbers $k, \ell, q \in \mathbb{N}$, $\delta, \varepsilon \in (0,1)$, and oracle access to a probability distribution $\mathscr{D}$ on $(V(G))^k \times \{+,-\}$, the algorithm queries at least $m(c, |\Lambda|, q, k, \ell, \delta, \varepsilon)$ many examples from $\mathscr{D}$ and then follows the ERM rule.*

Using this lemma, we can now give a proof sketch for Theorem 1.3, showing that MSO-PAC-Learn is fixed-parameter linear on classes of bounded clique-width if the input graph is given as a c-expression, even for dimensions $k > 1$. The full proof can be found in [10].

Proof sketch of Theorem 1.3. Let G be a Λ-labeled graph, let $k, \ell, q \in \mathbb{N}$, $\delta, \varepsilon \in (0,1)$, and assume we are given oracle access to a probability distribution $\mathscr{D}$ on $(V(G))^k \times \{+,-\}$. Moreover, let $c \in \mathbb{N}$ and let Ξ be a c-expression given as input that describes G.

Let $s := m(c, |\Lambda|, q, k, \ell, \delta, \varepsilon)$, where m is the function from Lemma 5.1. We sample s examples from $\mathscr{D}$ and call the resulting sequence of training examples S. Then, for every subsequence S' of S, we make use of the techniques in Section 3 (adapted to higher-dimensional training data) and compute a hypothesis $h_{S'} \in \mathscr{H}_{q,k,\ell}$ that is consistent with S' if such a hypothesis exists. This can be computed by an fpt-algorithm with parameters $c, |\Lambda|, k, \ell, q, \delta$, and ε. Finally, from all subsequences with a consistent hypothesis, we choose a subsequence S^* of maximum length and return h_{S^*}. Note that this procedure minimizes the training error on the training sequence S, i. e., $\operatorname{err}_S(h_{S^*}) = \min_{h \in \mathscr{H}_{q,k,\ell}} \operatorname{err}_S(h)$. Hence, the procedure follows the ERM rule, and, by Lemma 5.1, it solves MSO-PAC-Learn. Since the number of subsequences to check can be bounded by 2^s, all in all, the described procedure is an fpt-algorithm that solves MSO-PAC-Learn. ◀

6 Consistent Learning in Higher Dimensions

In this section, we consider the problem MSO-Consistent-Learn with dimension $k > 1$ on labeled graphs of bounded clique-width. A brute-force attempt yields a solution in time $g(c, |\Lambda|, q, k, \ell) \cdot (V(G))^{\ell+1} \cdot m$, where m is the length of the training sequence, for some $g \colon \mathbb{N}^5 \to \mathbb{N}$. This is achieved by iterating over all formulas, then iterating over all parameter assignments, and then performing model checking for each training example. We assume that the graph G is considerably larger in scale than the sequence of training examples S. Therefore, Theorem 1.4 significantly improves the running time to $\mathcal{O}\big((m+1)^{g(c,|\Lambda|,q,k,\ell)}|V(G)|^2\big)$. While Theorem 1.4 is not a fixed-parameter tractability result in the classical sense, we show that this is optimal in Section 7. The present section is dedicated to the proof of Theorem 1.4.

Until now, we have viewed the training sequence as a sequence of tuples $S \in ((V(G))^k \times \{+,-\})^m$. In the following, it is useful to split the training sequence into two parts, a sequence of vertex tuples $\mathscr{a} \in ((V(G))^k)^m$ and a function $\sigma \colon [m] \to \{+,-\}$ which assigns the corresponding label to each tuple. Let G be a Λ-labeled graph, $\mathscr{a} = (\bar{v}_1, \dots, \bar{v}_m) \in ((V(G))^k)^m$, $\sigma \colon [m] \to \{+,-\}$, and $\varphi(\bar{x}, \bar{y}) \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$. We call $(G, \mathscr{a}, \sigma)$ *φ-consistent* if there is a parameter setting $\bar{w} \in (V(G))^\ell$ such that for all $i \in [m]$, $G \models \varphi(\bar{v}_i, \bar{w}) \iff \sigma(\bar{v}_i) = +$. We say that $\bar{w}$ is a *φ-witness* for $(G, \mathscr{a}, \sigma)$. This notation allows us to state the main technical ingredient for the proof of Theorem 1.4 as follows.

▶ **Theorem 6.1.** *There is a computable function $g \colon \mathbb{N}^5 \to \mathbb{N}$ and an algorithm that, given a Λ-graph G of clique-width $\mathsf{cw}(G) \leq c$, a sequence $\mathscr{a} = (\bar{v}_1, \dots, \bar{v}_m) \in ((V(G))^k)^m$, a function $\sigma \colon [m] \to \{+,-\}$, and a formula $\varphi(\bar{x}, \bar{y}) \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$, decides if $(G, \mathscr{a}, \sigma)$ is φ-consistent in time $(m+1)^{g(c,|\Lambda|,q,k,\ell)}|G|$.*

Using Theorem 6.1, we can now prove Theorem 1.4.

Proof of Theorem 1.4. Given a Λ-labeled graph G and a training sequence $S = ((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)) \in ((V(G))^k \times \{+,-\})^m$, we let $\mathscr{a} := (\bar{v}_1, \dots, \bar{v}_m) \in ((V(G))^k)^m$ and $\sigma \colon [m] \to \{+,-\}, i \mapsto \lambda_i$. We iterate over all formulas $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$ and use Theorem 6.1 to check whether $(G, \mathscr{a}, \sigma)$ is φ-consistent. If there is no φ such that $(G, \mathscr{a}, \sigma)$ is φ-consistent, then we reject the input. Otherwise, let $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$ be such that $(G, \mathscr{a}, \sigma)$ is φ-consistent. We compute a φ-witness following the same construction as in the proof of Lemma 3.1. That is, using a fresh label, we encode the parameter choice of a single variable into the graph, and then we check whether consistency still holds for the

corresponding formula φ' that enforces this parameter choice. In total, we perform up to $\ell \cdot |V(G)|$ such consistency checks to compute a φ-witness $\bar{w}$. The consistent formula φ together with the φ-witness $\bar{w}$ can then be returned as a hypothesis $h_{\varphi,\bar{w}}$ that is consistent with S on G and therefore a solution to MSO-Consistent-Learn. ◀

The remainder of this section is dedicated to the proof of Theorem 6.1. We start by introducing the formal definitions for types and sets of types over sequences of elements.

6.1 Type Definitions

Let G be a Λ-labeled graph and $\bar{v} \in (V(G))^k$. Recall that the *q-type of $\bar{v}$ in G* is the set $\mathsf{tp}_q^G(\bar{v})$ of all formulas $\varphi(\bar{x}) \in \mathsf{MSO}(\Lambda, q, k, 0)$ such that $G \models \varphi(\bar{v})$. A *$(\Lambda, q, k)$-type* is a set $\theta \subseteq \mathsf{MSO}(\Lambda, q, k, 0)$ such that, for each $\varphi \in \mathsf{MSO}(\Lambda, q, k, 0)$, either $\varphi \in \theta$ or $\neg\varphi \in \theta$. We denote the set of all (Λ, q, k)-types by $\mathsf{Tp}(\Lambda, q, k)$. Note that $\mathsf{tp}_q^G(\bar{v}) \in \mathsf{Tp}(\Lambda, q, k)$. For a type $\theta \in \mathsf{Tp}(\Lambda, q, k)$, we write $G \models \theta(\bar{v})$ if $G \models \varphi(\bar{v})$ for all $\varphi(\bar{x}) \in \theta$. Observe that $G \models \theta(\bar{v}) \iff \mathsf{tp}_q^G(\bar{v}) = \theta$. We say that a type $\theta \in \mathsf{Tp}(\Lambda, q, k)$ is *realizable* if there is some Λ-labeled graph G and tuple $\bar{v} \in (V(G))^k$ such that $\theta = \mathsf{tp}_q^G(\bar{v})$. We are not particularly interested in types that are not realizable, but it is undecidable if a type θ is realizable, whereas the sets $\mathsf{Tp}(\Lambda, q, k)$ are decidable. (More precisely, there is an algorithm that, given Λ, q, k and a set θ of formulas, decides if $\theta \in \mathsf{Tp}(\Lambda, q, k)$.) For a (Λ, q, k)-type θ and a Λ-labeled graph G, we let

$$\theta(G) \coloneqq \big\{\bar{v} \in (V(G))^k \;\big|\; G \models \theta(\bar{v})\big\}.$$

If $\theta(G) \neq \emptyset$, we say that θ is *realizable in G*.

As for formulas, we split the variables for types into two parts, so we consider (Λ, q, k, ℓ)-types $\theta \subseteq \mathsf{MSO}(\Lambda, q, k, \ell, 0)$, and we denote the set of all these types by $\mathsf{Tp}(\Lambda, q, k, \ell)$. For a Λ-labeled graph G and tuples $\bar{v} \in (V(G))^k, \bar{w} \in (V(G))^\ell$, we often think of $\mathsf{tp}_q^G(\bar{v}, \bar{w})$ as the *q-type of $\bar{w}$ over $\bar{v}$ in G*. Moreover, we let

$$\theta(\bar{v}, G) \coloneqq \{\bar{w} \in (V(G))^\ell \mid G \models \theta(\bar{v}, \bar{w})\}.$$

If $\theta(\bar{v}, G) \neq \emptyset$, we say that θ is *realizable over $\bar{v}$ in G*.

For a vector $\bar{k} = (k_1, \ldots, k_m) \in \mathbb{N}^m$ and a set V, we let $V^{\bar{k}}$ be the set of all sequences $\mathscr{a} = (\bar{v}_1, \ldots, \bar{v}_m)$ of tuples $\bar{v}_i \in V^{k_i}$. Let G be a labeled graph, $\mathscr{a} = (\bar{v}_1, \ldots, \bar{v}_m) \in V^{\bar{k}}$ for some $\bar{k} \in \mathbb{N}^m$, and $\bar{w} \in (V(A))^\ell$. We define the *q-type of $\bar{w}$ over $\mathscr{a}$ in G* to be the tuple

$$\mathsf{tp}_q^G(\mathscr{a}, \bar{w}) \coloneqq \big(\mathsf{tp}_q^G(\bar{v}_1, \bar{w}), \ldots, \mathsf{tp}_q^G(\bar{v}_m, \bar{w})\big).$$

Again, we need an "abstract" notion of type over a sequence. A *$(\Lambda, q, \bar{k}, \ell)$-type* for a tuple $\bar{k} = (k_1, \ldots, k_m) \in \mathbb{N}^m$ is an element of

$$\mathsf{Tp}(\Lambda, q, \bar{k}, \ell) \coloneqq \prod_{i=1}^{m} \mathsf{Tp}(\Lambda, q, k_i, \ell).$$

Let $\bar{\theta} = (\theta_1, \ldots, \theta_m) \in \mathsf{Tp}(\Lambda, q, \bar{k}, \ell)$. For a labeled graph G, a sequence $\mathscr{a} = (\bar{v}_1, \ldots, \bar{v}_m) \in (V(G))^{\bar{k}}$, and a tuple $\bar{w} \in (V(G))^\ell$, we write $G \models \bar{\theta}(\mathscr{a}, \bar{w})$ if $G \models \theta_i(\bar{v}_i, \bar{w})$ for all $i \in [m]$. Note that $G \models \bar{\theta}(\mathscr{a}, \bar{w}) \iff \mathsf{tp}_q^G(\mathscr{a}, \bar{w}) = \bar{\theta}$. For a type $\bar{\theta} \in \mathsf{Tp}(\Lambda, q, \bar{k}, \ell)$, a Λ-labeled graph G, and a sequence $\mathscr{a} \in (V(G))^{\bar{k}}$, we let

$$\bar{\theta}(\mathscr{a}, G) \coloneqq \big\{\bar{w} \in (V(G))^\ell \;\big|\; G \models \theta(\mathscr{a}, \bar{w})\big\}.$$

If $\bar{\theta}(\mathscr{a}, G) \neq \emptyset$, we say that $\bar{\theta}$ is *realizable over $\mathscr{a}$ in G*.

6.2 Computing the Realizable Types

For the proof of Theorem 6.1, we use the following result that allows us to compute the realizable types of an expression.

▶ **Lemma 6.2.** *There is a computable function $f\colon \mathbb{N}^4 \to \mathbb{N}$ and an algorithm that, given $c, q, k, \ell, m \in \mathbb{N}$, a vector $\bar{k} = (k_1, \dots, k_m) \in \mathbb{N}^m$ with $k_i \leq k$ for all $i \in [m]$, a Λ-expression Ξ with $|\Lambda| \leq c$, and a sequence $\mathscr{a} \in (V_\Xi)^{\bar{k}}$, computes the set of all $\bar{\theta} \in \mathsf{Tp}(\Lambda, q, \bar{k}, \ell)$ that are realizable over $\mathscr{a}$ in G_Ξ in time*

$$\mathcal{O}\Big((m+1)^{f(c,q,k,\ell)} \cdot |\Xi|\Big).$$

Before proving this result, we first show that it implies Theorem 6.1.

Proof of Theorem 6.1. We assume that the input graph G is given as a c-expression. To check whether $(G, \mathscr{a}, \sigma)$ is φ-consistent, we compute the set $\mathscr{R}$ of all $\bar{\theta} \in \mathsf{Tp}(\Lambda, q, \bar{k}, \ell)$ that are realizable over $\mathscr{a}$ in G, using Lemma 6.2. Then, for each $\bar{\theta} = (\theta_1, \dots, \theta_m) \in \mathscr{R}$ we check if $\varphi \in \theta_i \iff \sigma(i) = +$. If we find such a $\bar{\theta}$, then $(G, \mathscr{a}, \sigma)$ is φ-consistent; otherwise it is not. ◀

It remains to prove Lemma 6.2. In a bottom-up algorithm, starting at the leaves of Ξ, we compute the set of all tuples $\bar{\theta} = (\theta_1, \dots, \theta_m)$ of (Λ, q, k_i, ℓ)-types θ_i that are realizable over $\mathscr{a}$ in G. This is possible because the realizable tuples of types of an expression can be computed from the tuples of types of its subexpressions. We formally prove this for the operators $\eta_{P,Q}$, $\rho_{P,Q}$, δ_P, and $\uplus^<$ in the full version [10].

The difficulty with this approach is that we are talking about m-tuples of types. In general, the number of such tuples is exponential in m, and hence the size of the set we aim to compute could be exponentially large. Fortunately, this does not happen in graphs of bounded clique-width. By Lemma 2.2, we can bound the VC dimension of a first-order formula over classes of graphs of bounded clique-width. Further, we show in Lemma 6.3 that this suffices to give a polynomial bound for the number of realizable tuples.

▶ **Lemma 6.3.** *Let $d, q, k, \ell \in \mathbb{N}$, let $t \coloneqq |\mathsf{Tp}(\Lambda, q, k, \ell)|$, and let G be a Λ-labeled graph such that $\mathrm{VC}(\varphi, G) \leq d$ for all $\varphi \in \mathsf{MSO}(\Lambda, q, k, \ell, 0)$. Let $\mathscr{a} \in (V(G))^{\bar{k}}$ for some $\bar{k} \in \{0, \dots, k\}^m$. Then at most $(k+1) \cdot g(d, m)^t$ types in $\mathsf{Tp}(\Lambda, q, \bar{k}, \ell)$ are realizable over $\mathscr{a}$ in G.*

The proof of Lemma 6.3 is based on the Sauer–Shelah Lemma [45, 47]. See [10] for proof details. Based on this, we can now prove Lemma 6.2.

Proof of Lemma 6.2. As argued in Section 2, we may assume that Ξ only contains ordered-disjoint-union operators and no plain disjoint-union operators.

For every subexpression Ξ', we let $\Lambda_{\Xi'}$ be the set of labels of Ξ', that is, the set of unary relation symbols such that $G_{\Xi'}$ is a $\Lambda_{\Xi'}$-graph. Moreover, let $\mathscr{a}_{\Xi'} \coloneqq \mathscr{a} \cap V_{\Xi'}$, and let $\bar{k}_{\Xi'} \subseteq \mathbb{N}^m$ such that $\mathscr{a}_{\Xi'} \in (V_{\Xi'})^{\bar{k}_{\Xi'}}$.

We inductively construct, for every subexpression Ξ' of Ξ and $0 \leq \ell' \leq \ell$, the set $\mathscr{R}_{\ell'}(\Xi')$ of all types $\bar{\theta} \in \mathsf{Tp}(\Lambda_{\Xi'}, q, \bar{k}_{\Xi'}, \ell')$ that are realizable over $\mathscr{a}_{\Xi'}$ in $G_{\Xi'}$.

Case 1: Ξ' is a base expression. In this case, for each $\ell' \in [\ell]$, we can construct $\mathscr{R}_{\ell'}(\Xi')$ by brute force in time $f_1(c, q, k, \ell) \cdot m$ for a suitable (computable) function f_1. Let $(k'_1, \dots, k'_m) \coloneqq \bar{k}_{\Xi'}$. We compute θ_i by iterating over all formulas φ with $k'_i + \ell'$ free variables and evaluating φ on the single vertex graph $G_{\Xi'}$.

Case 2: $\Xi' = \eta_{P,Q}(\Xi'')$. Let $0 \leq \ell' \leq \ell$, and let $\bar{k}' = (k'_1, \dots, k'_m) \coloneqq \bar{k}_{\Xi'} = \bar{k}_{\Xi''}$. As we show in the full version [10], there is a computable mapping $T_{\eta,P,Q} \colon \mathsf{Tp}(\Lambda_{\Xi'}) \to 2^{\mathsf{Tp}(\Lambda_{\Xi''})}$ such that $\mathscr{R}_{\ell'}(\Xi')$ is the set of all $\bar{\theta} = (\theta_1, \dots, \theta_m) \in \mathsf{Tp}(\Lambda_{\Xi'}, q, \bar{k}', \ell')$ such that there is a $\bar{\theta}' = (\theta'_1, \dots, \theta'_m) \in \mathscr{R}(\Xi'')$ with $\theta'_i \in T_{\eta,P,Q}(\theta_i)$ for all $i \in [m]$. Moreover, for every realizable $\theta' \in \mathsf{Tp}(\Lambda_{\Xi'})$, we guarantee that there is at most one type $\theta \in \mathsf{Tp}(\Lambda_{\Xi''})$ such that $\theta' \in T_{\eta,P,Q}(\theta)$. To compute the set $\mathscr{R}_{\ell'}(\Xi')$, we step through all $\bar{\theta}' \in \mathscr{R}(\Xi'')$. For each such $\bar{\theta}' = (\theta'_1, \dots, \theta'_m)$, for all $i \in [m]$, we compute the unique $\theta_i \in \mathsf{Tp}(\Lambda_{\Xi'}, q, k'_i, \ell')$ such that $\theta'_i \in T_{\eta,P,Q}(\theta_i)$. If for some $i \in [m]$, no such θ_i exists, we move on to the next $\bar{\theta}'$. Otherwise, we add $\bar{\theta} = (\theta_1, \dots, \theta_m)$ to $\mathscr{R}(\Xi')$.

Case 3: $\Xi' = \rho_{P,Q}(\Xi'')$. Analogous to Case 2, again based on results from the full version [10].

Case 4: $\Xi' = \delta_P(\Xi'')$. Analogous to Case 2, based on results from [10].

Case 5: $\Xi' = \Xi_1 \uplus^{<} \Xi_2$. Let $\Lambda' \coloneqq \Lambda_{\Xi'}$, $V' \coloneqq V_{\Xi'}$, $\bar{k}' = (k'_1, \dots, k'_m) \coloneqq \bar{k}_{\Xi'}$, and $\mathscr{a}' \coloneqq (\bar{v}'_1, \dots, \bar{v}'_m) \coloneqq \mathscr{a}_{\Xi'} = \mathscr{a} \cap V'$. For $j = 1, 2$, let $\Lambda_j \coloneqq \Lambda_{\Xi_j}$, $V_j \coloneqq V_{\Xi_j}$, $\bar{k}_j \coloneqq (k_{j1}, \dots, k_{jm}) \coloneqq \bar{k}_{\Xi_j}$, and for all $i \in [m]$, let $K_{ji} \subseteq [k_{ji}]$ such that $\bar{v}'_i \cap V_j = (\bar{v}_i)_{K_{ji}}$. Let $0 \leq \ell' \leq \ell$.

For all $L_1, L_2 \subseteq [\ell']$ such that $L_2 = [\ell'] \setminus L_1$, we let $\mathscr{R}_{L_1,L_2}$ be the set of all $\bar{\theta} = (\theta_1, \dots, \theta_m) \in \mathsf{Tp}(\Lambda', q, \bar{k}', \ell')$ such that for $j = 1, 2$, we have

$$\bar{\theta}_j \coloneqq \big(T_{\uplus,j,L_j,K_{j1}}(\theta_1), \dots, T_{\uplus,j,L_j,K_{jm}}(\theta_m)\big) \in \mathscr{R}_{|L_j|}(\Xi_j),$$

where $T_{\uplus,j,L_j,K_{ji}} \colon \mathsf{Tp}(\Lambda') \to \mathsf{Tp}(\Lambda_j)$ for $i \in [m]$ is a computable mapping that we give in the full version [10]. Then, as we also show in [10],

$$\mathscr{R}_{\ell'}(\Xi') = \bigcup_{\substack{L_1 \subseteq [\ell'] \\ L_2 = [\ell'] \setminus L_1}} \mathscr{R}_{L_1,L_2}.$$

To compute $\mathscr{R}_{L_1,L_2}$, we iterate over all $\bar{\theta}_1 = (\theta_{11}, \dots, \theta_{1m}) \in \mathscr{R}_{|L_1|}(\Xi_1)$. For all $i \in [m]$ we compute the unique $\theta_i \in \mathsf{Tp}(\Lambda', q, k'_i, \ell')$ such that $T_{\uplus,1,L_1,K_{1i}}(\theta_i) = \theta_{1i}$. If, for some $i \in [m]$, no such θ_i exists, then we move on to the next $\bar{\theta}_1$. Otherwise, we compute

$$\bar{\theta}_2 = \big(T_{\uplus,2,L_2,K_{21}}(\theta_1), \dots, T_{\uplus,2,L_2,K_{2m}}(\theta_m)\big)$$

and check if $\bar{\theta}_2 \in \mathscr{R}_{|L_2|}(\Xi_2)$. If it is, we add $\bar{\theta}$ to $\mathscr{R}_{L_1,L_2}$. Otherwise, we move on to the next $\bar{\theta}_1$.

This completes the description of our algorithm. To analyze the running time, let

$$r \coloneqq \max_{\Xi'} |\mathscr{R}_{\Xi'}|,$$

where Ξ' ranges over all subexpressions of Ξ. By Lemmas 2.2 and 6.3, there is a computable function $f_2 \colon \mathbb{N}^4 \to \mathbb{N}$ such that

$$r \leq (m+1)^{f_2(c,q,k,\ell)}.$$

The running time of each step of the constructions can be bounded by $f_3(c, q, k, \ell) \cdot r$ for a suitable computable function f_3. We need to make $|\Xi|$ steps. Thus, overall, we obtain the desired running time. ◀

7 Hardness of Checking Consistency in Higher Dimensions

The following result shows, under the assumption FPT $\neq$ W[1], that Theorem 6.1 can not be improved to an fpt-result.

▶ **Theorem 7.1.** *There is a $q \in \mathbb{N}$ such that the following parameterized problem is* W[1]*-hard.*

Instance: graph G of clique-width at most 2, sequence $\mathscr{a} = (\bar{a}_1, \ldots, \bar{a}_m) \in \big((V(G))^2\big)^m$, function $\sigma\colon [m] \to \{+1, -1\}$, formula $\varphi(\bar{x}, \bar{y}) \in \mathsf{MSO}(\Lambda, q, 2, \ell, 0)$ **Parameter**: ℓ **Problem**: decide if $(G, \mathscr{a}, \sigma)$ is φ-consistent.

Proof sketch. We prove this by a reduction from the W[1]-complete *weighted satisfiability problem* for Boolean formulas in 2-conjunctive normal form [26]. The *weight* of an assignment to a set of Boolean variables is the number of variables set to 1.

WSat(2-CNF) **Instance**: Boolean formula Φ in 2-CNF **Parameter**: ℓ **Problem**: decide if Φ has a satisfying assignment of weight ℓ.

Given a 2-CNF formula $\Phi = \bigwedge_{i=1}^{m}(L_{i,1} \vee L_{i,2})$ in the variables $\{X_1, \ldots, X_n\}$ and $\ell \in \mathbb{N}$, we construct an instance $(G, \mathscr{a}, \sigma, \varphi)$ of the consistency problem from Theorem 7.1 where the graph G is a forest that encodes Φ. Moreover, $\mathscr{a}$, φ, and σ are chosen to verify that a φ-witness $\bar{w} \in (V(G))^\ell$ for $(G, \mathscr{a}, \sigma)$ encodes exactly ℓ variables among $X_1, \ldots, X_n$ that can be set to 1 to satisfy Φ. Hence, there is a φ-witness for $(G, \mathscr{a}, \sigma)$ if and only if Φ has a satisfying assignment of weight ℓ. Details on the construction can be found in the full version [10]. ◀

8 Conclusion

Just like model checking and the associated counting and enumeration problems, the learning problem we study here is a natural algorithmic problem for logics on finite structures. All these problems are related, but each has its own challenges requiring different techniques. Where model checking and enumeration are motivated by automated verification and database systems, we view our work as part of a descriptive complexity theory of machine learning [8].

The first problem we studied is 1D-MSO-Consistent-Learn, where the instances to classify consist of single vertices, and we extended the previous fixed-parameter tractability results for strings and trees [31, 30] to (labeled) graphs of bounded clique-width. Moreover, on general graphs, we showed that the problem is hard for the complexity class para-NP.

For MSO-learning problems in higher dimensions, we presented two different approaches that yield tractability results on graphs of bounded clique-width. For the agnostic PAC-learning problem MSO-PAC-Learn, we described a fixed-parameter tractable learning algorithm. Furthermore, in the consistent-learning setting for higher dimensions, we gave an algorithm that solves the learning problem and is fixed-parameter tractable in the size of the input graph. However, the algorithm is not fixed-parameter tractable in the size of the training sequence, and we showed that this is optimal.

In the learning problems considered so far, hypotheses are built using MSO formulas and tuples of vertices as parameters. We think that the algorithms presented in this paper for the 1-dimensional case could also be extended to hypothesis classes that allow tuples of sets

as parameters. Finally, utilizing the full power of MSO, one could also consider a learning problem where, instead of classifying tuples of vertices, we are interested in classifying sets of vertices. That is, for a graph G, we are given labeled subsets of $V(G)$ and want to find a hypothesis $h\colon 2^{V(G)} \to \{+,-\}$ that is consistent with the given examples. It is easy to see that the techniques used in our hardness result also apply to this modified problem, proving that it is para-NP-hard. However, it remains open whether our tractability results could also be lifted to this version of the problem.

References

1 Azza Abouzied, Dana Angluin, Christos H. Papadimitriou, Joseph M. Hellerstein, and Avi Silberschatz. Learning and verifying quantified boolean queries by example. In *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2013, New York, NY, USA, June 22–27, 2013*, pages 49–60. ACM, 2013. doi:10.1145/2463664.2465220.

2 Howard Aizenstein, Tibor Hegedüs, Lisa Hellerstein, and Leonard Pitt. Complexity theoretic hardness results for query learning. *Comput. Complex.*, 7(1):19–53, 1998. doi:10.1007/PL00001593.

3 Bogdan Alexe, Balder ten Cate, Phokion G. Kolaitis, and Wang Chiew Tan. Characterizing schema mappings via data examples. *ACM Trans. Database Syst.*, 36(4):23:1–23:48, 2011. doi:10.1145/2043652.2043656.

4 Vikraman Arvind, Johannes Köbler, and Wolfgang Lindner. Parameterized learnability of k-juntas and related problems. In *Algorithmic Learning Theory, 18th International Conference, ALT 2007, Sendai, Japan, October 1–4, 2007*, volume 4754 of *Lecture Notes in Computer Science*, pages 120–134. Springer, 2007. doi:10.1007/978-3-540-75225-7_13.

5 Pablo Barceló, Alexander Baumgartner, Víctor Dalmau, and Benny Kimelfeld. Regularizing conjunctive features for classification. *J. Comput. Syst. Sci.*, 119:97–124, 2021. doi:10.1016/j.jcss.2021.01.003.

6 Pablo Barceló and Miguel Romero. The complexity of reverse engineering problems for conjunctive queries. In *20th International Conference on Database Theory, ICDT 2017, Venice, Italy, March 21–24, 2017*, volume 68 of *LIPIcs*, pages 7:1–7:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICDT.2017.7.

7 Steffen van Bergerem. Learning concepts definable in first-order logic with counting. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24–27, 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785811.

8 Steffen van Bergerem. *Descriptive Complexity of Learning*. PhD thesis, RWTH Aachen University, Germany, 2023. doi:10.18154/RWTH-2023-02554.

9 Steffen van Bergerem, Martin Grohe, and Martin Ritzert. On the parameterized complexity of learning first-order logic. In *PODS 2022: International Conference on Management of Data, Philadelphia, PA, USA, June 12–17, 2022*, pages 337–346. ACM, 2022. doi:10.1145/3517804.3524151.

10 Steffen van Bergerem, Martin Grohe, and Nina Runde. The parameterized complexity of learning monadic second-order logic, 2023. doi:10.48550/arXiv.2309.10489.

11 Steffen van Bergerem and Nicole Schweikardt. Learning concepts described by weight aggregation logic. In *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, Ljubljana, Slovenia (Virtual Conference), January 25–28, 2021*, volume 183 of *LIPIcs*, pages 10:1–10:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CSL.2021.10.

12 Anselm Blumer, A. Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, October 1989. doi:10.1145/76359.76371.

13 Angela Bonifati, Radu Ciucanu, and Aurélien Lemay. Learning path queries on graph databases. In *Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015, Brussels, Belgium, March 23–27, 2015*, pages 109–120. OpenProceedings.org, 2015. doi:10.5441/002/edbt.2015.11.

14 Angela Bonifati, Radu Ciucanu, and Slawek Staworko. Learning join queries from user examples. *ACM Trans. Database Syst.*, 40(4):24:1–24:38, 2016. doi:10.1145/2818637.

15 Angela Bonifati, Ugo Comignani, Emmanuel Coquery, and Romuald Thion. Interactive mapping specification with exemplar tuples. *ACM Trans. Database Syst.*, 44(3):10:1–10:44, 2019. doi:10.1145/3321485.

16 Cornelius Brand, Robert Ganian, and Kirill Simonov. A parameterized theory of PAC learning. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7–14, 2023*, pages 6834–6841. AAAI Press, 2023. doi:10.1609/aaai.v37i6.25837.

17 Balder ten Cate and Víctor Dalmau. Conjunctive queries: Unique characterizations and exact learnability. In *24th International Conference on Database Theory, ICDT 2021, Nicosia, Cyprus, March 23–26, 2021*, volume 186 of *LIPIcs*, pages 9:1–9:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICDT.2021.9.

18 Balder ten Cate, Víctor Dalmau, and Phokion G. Kolaitis. Learning schema mappings. *ACM Trans. Database Syst.*, 38(4):28:1–28:31, 2013. doi:10.1145/2539032.2539035.

19 Balder ten Cate, Phokion G. Kolaitis, Kun Qian, and Wang-Chiew Tan. Active learning of GAV schema mappings. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10–15, 2018*, pages 355–368. ACM, 2018. doi:10.1145/3196959.3196974.

20 Adrien Champion, Tomoya Chiba, Naoki Kobayashi, and Ryosuke Sato. ICE-based refinement type discovery for higher-order functional programs. *J. Autom. Reason.*, 64(7):1393–1418, 2020. doi:10.1007/s10817-020-09571-y.

21 William W. Cohen and C. David Page Jr. Polynomial learnability and inductive logic programming: Methods and results. *New Gener. Comput.*, 13(3&4):369–409, December 1995. doi:10.1007/BF03037231.

22 Derek G. Corneil and Udi Rotics. On the relationship between clique-width and treewidth. *SIAM J. Comput.*, 34(4):825–847, 2005. doi:10.1137/S0097539701385351.

23 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

24 Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discret. Appl. Math.*, 101(1-3):77–114, 2000. doi:10.1016/S0166-218X(99)00184-5.

25 P. Ezudheen, Daniel Neider, Deepak D'Souza, Pranav Garg, and P. Madhusudan. Horn-ICE learning for synthesizing invariants and contracts. *Proc. ACM Program. Lang.*, 2(OOPSLA):131:1–131:25, 2018. doi:10.1145/3276501.

26 Jörg Flum and Martin Grohe. *Parameterized complexity theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

27 Martin Fürer. Multi-clique-width. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, Berkeley, CA, USA, January 9–11, 2017*, volume 67 of *LIPIcs*, pages 14:1–14:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ITCS.2017.14.

28 Pranav Garg, Christof Löding, P. Madhusudan, and Daniel Neider. ICE: A robust framework for learning invariants. In *Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18–22, 2014*, volume 8559 of *Lecture Notes in Computer Science*, pages 69–87. Springer, 2014. doi:10.1007/978-3-319-08867-9_5.

29 Georg Gottlob and Pierre Senellart. Schema mapping discovery from data instances. *J. ACM*, 57(2):6:1–6:37, 2010. doi:10.1145/1667053.1667055.

30 Émilie Grienenberger and Martin Ritzert. Learning definable hypotheses on trees. In *22nd International Conference on Database Theory, ICDT 2019, Lisbon, Portugal, March 26–28, 2019*, volume 127 of *LIPIcs*, pages 24:1–24:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICDT.2019.24.

31 Martin Grohe, Christof Löding, and Martin Ritzert. Learning MSO-definable hypotheses on strings. In *International Conference on Algorithmic Learning Theory, ALT 2017, Kyoto University, Kyoto, Japan, October 15–17, 2017*, volume 76 of *Proceedings of Machine Learning Research*, pages 434–451. PMLR, October 2017. ISSN: 2640-3498. URL: https://proceedings.mlr.press/v76/grohe17a.html.

32 Martin Grohe and Martin Ritzert. Learning first-order definable concepts over structures of small degree. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavík, Iceland, June 20–23, 2017*, pages 1–12. IEEE, June 2017. doi:10.1109/LICS.2017.8005080.

33 Martin Grohe and György Turán. Learnability and definability in trees and similar structures. *Theory Comput. Syst.*, 37(1):193–220, January 2004. doi:10.1007/s00224-003-1112-8.

34 David Haussler. Learning conjunctive concepts in structural domains. *Mach. Learn.*, 4:7–40, 1989. doi:10.1007/BF00114802.

35 David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inf. Comput.*, 100(1):78–150, 1992. doi:10.1016/0890-5401(92)90010-D.

36 Kouichi Hirata. On the hardness of learning acyclic conjunctive queries. In *Algorithmic Learning Theory, 11th International Conference, ALT 2000, Sydney, Australia, December 11–13, 2000*, volume 1968 of *Lecture Notes in Computer Science*, pages 238–251. Springer, 2000. doi:10.1007/3-540-40992-0_18.

37 Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994. URL: https://mitpress.mit.edu/books/introduction-computational-learning-theory.

38 Benny Kimelfeld and Christopher Ré. A relational framework for classifier engineering. *ACM Trans. Database Syst.*, 43(3):11:1–11:36, 2018. doi:10.1145/3268931.

39 Yuanzhi Li and Yingyu Liang. Learning mixtures of linear regressions with nearly optimal complexity. In *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, July 6–9, 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1125–1144. PMLR, 2018. URL: http://proceedings.mlr.press/v75/li18b.html.

40 Christof Löding, P. Madhusudan, and Daniel Neider. Abstract learning frameworks for synthesis. In *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2–8, 2016*, volume 9636 of *Lecture Notes in Computer Science*, pages 167–185. Springer, 2016. doi:10.1007/978-3-662-49674-9_10.

41 Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning. MIT Press, 2nd edition, 2018.

42 Stephen H. Muggleton. Inductive logic programming. *New Gener. Comput.*, 8(4):295–318, February 1991. doi:10.1007/BF03037089.

43 Stephen H. Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *J. Log. Program.*, 19/20:629–679, 1994. doi:10.1016/0743-1066(94)90035-3.

44 Sang-il Oum and Paul D. Seymour. Approximating clique-width and branch-width. *J. Comb. Theory, Ser. B*, 96(4):514–528, 2006. doi:10.1016/j.jctb.2005.10.006.

45 Norbert Sauer. On the density of families of sets. *J. Comb. Theory, Ser. A*, 13(1):145–147, 1972. doi:10.1016/0097-3165(72)90019-2.

46 Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, Cambridge, 2014. doi:10.1017/CBO9781107298019.

47 Saharon Shelah. A combinatorial problem: stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, 1972. doi:10.2140/pjm.1972.41.247.

48 Robert H. Sloan, Balázs Szörényi, and György Turán. Learning Boolean functions with queries. In *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, pages 221–256. Cambridge University Press, 2010. doi:10.1017/cbo9780511780448.010.

49 Slawek Staworko and Piotr Wieczorek. Learning twig and path queries. In *15th International Conference on Database Theory, ICDT 2012, Berlin, Germany, March 26–29, 2012*, pages 140–154. ACM, 2012. doi:10.1145/2274576.2274592.

50 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984. doi:10.1145/1968.1972.

51 Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2–5, 1991]*, pages 831–838, 1991. URL: https://proceedings.neurips.cc/paper_files/paper/1991/file/ff4d5fbbafdf976cfdc032e3bde78de5-Paper.pdf.

52 He Zhu, Stephen Magill, and Suresh Jagannathan. A data-driven CHC solver. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2018, Philadelphia, PA, USA, June 18–22, 2018*, pages 707–721. ACM, 2018. doi:10.1145/3192366.3192416.

On Homogeneous Models of Fluted Languages

Daumantas Kojelis
Department of Computer Science, University of Manchester, UK

Abstract

We study the fluted fragment of first-order logic which is often viewed as a multi-variable non-guarded extension to various systems of description logics lacking role-inverses. In this paper we show that satisfiable fluted sentences (even under reasonable extensions) admit special kinds of "nice" models which we call globally/locally homogeneous. Homogeneous models allow us to simplify methods for analysing fluted logics with counting quantifiers and establish a novel result for the decidability of the (finite) satisfiability problem for the fluted fragment with periodic counting. More specifically, we will show that the (finite) satisfiability problem for the language is TOWER-complete. If only two variable are used, computational complexity drops to NEXPTIME-completeness. We supplement our findings by showing that generalisations of fluted logics, such as the adjacent fragment, have finite and general satisfiability problems which are, respectively, Σ_1^0- and Π_1^0-complete. Additionally, satisfiability becomes Σ_1^1-complete if periodic counting quantifiers are permitted.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic

Keywords and phrases Fluted Fragment, Fluted Logic, Fluted Fragment with Periodic Counting, Adjacent Fragment, Adjacent Fragment with Counting, Adjacent Fragment with Periodic Counting, Counting Quantifiers, Periodic Counting Quantifiers, Decidable Fragments of First-Order Logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.9

Funding This work is supported by the NCN grant 2018/31/B/ST6/03662.

Acknowledgements The author would like to thank Dr. Ian Pratt-Hartmann and Dr. Bartosz Jan Bednarczyk for their valuable feedback.

1 Introduction

The *fluted fragment* (denoted $\mathcal{FL}$) is a fragment of first-order logic in which, roughly put, variables appear in predicates following the order in which they were quantified. For illustrative purposes, we translate the sentence "Every conductor nominates their favorite soloist to play at every concert" into this language as follows:

$$\forall x_1 \Big(cond(x_1) \to \exists x_2 \big(solo(x_2) \wedge fav(x_1, x_2) \wedge \forall x_3 (conc(x_3) \to nom(x_1, x_2, x_3)) \big) \Big). \quad (1)$$

As a non-example, the sentences axiomatising transitivity, symmetry and reflexivity of a relation are not in the fluted fragment.

The fluted fragment is a member of *argument-sequence logics* – a family of decidable (in terms of satisfiability) fragments of first-order logic which also includes the *ordered* [11, 13], *forward* [2] and *adjacent* [4] fragments. The fluted fragment in particular is decidable in terms of satisfiability even in the presence of counting quantifiers [19] or a distinguished transitive relation [22]. Surprisingly, the satisfiability problem for $\mathcal{FL}$ under a combination of the two not only retains decidability but also has the finite model property [24]. We refer the reader to [23] for a survey.

In this paper we will mostly be concerned with what we call the *fluted fragment with periodic counting* (denoted $\mathcal{FLPC}$). We remark that periodic counting quantifiers generalise standard (threshold) counting quantifiers which have been an object of intensive study as an extension for the fluted fragment in the past few years [19, 24]. Under this new formalism,

we are allowed to write formulas requesting an even number of existential witnesses. As an example, we can express sentences as "Every orchestra hires an even number of people to play first violin" in our language (see (2)).

The origins of *flutedness* trace back to the works of W. V. Quine [26]. It is, however, the definition given by W. C. Purdy (in [25]) that has become widespread and will be the one we use. The popularity of Purdy's idea of flutedness is not without cause, at least when keeping the field of description logics in mind. Indeed, after a routine translation, formulas of the description logic $\mathcal{ALC}$ are contained in the two-variable sub-fragment of $\mathcal{FLPC}$. This is even the case when $\mathcal{ALC}$ is augmented with role hierarchies, nominals and/or cardinality restrictions (possibly with modulo operations). We refer the reader to [12] for more details. In terms of expressive power, $\mathcal{FLPC}$ closely parallels $\mathcal{ALCSCC}$ – a new formalism with counting constrains expressible in quantifier-free Boolean algebra with Presburger arithmetic (see [16, 1]). Thus, noting that the guarded fragment with at least three variables becomes undecidable under counting extensions [8], and that the guarded fluted fragment has "nice" model theoretic properties such as *Craig interpolation* [3], fluted languages emerge as perfect candidates for generalising description logics in a multi-variable context.

In Sections 3 and 4 we establish that classes of models of satisfiable $\mathcal{FLPC}$-sentences always contain a "nice" structure in which elements behave (in a sense that we will make clear) *homogeneously*. Utilising this behaviour we will show that the fluted fragment extended with periodic counting quantifiers has a decidable satisfiability problem. Intriguingly, even though periodic counting quantifiers generalise standard counting quantifiers, our methodology allows us to avoid *Presburger quantification*, which was required to establish decidability of satisfiability for $\mathcal{FL}$ with standard counting [19].

In section 5 we show that the satisfiability problems for the fluted fragment with counting extensions become undecidable when minimal syntactic relaxations are allowed. More precisely, the section will culminate with a result showing that the finite satisfiability problem for the 3-variable adjacent fragment with counting is Σ^0_1-complete. Additionally, the general satisfiability problem will be shown to be Π^0_1-complete when 4 variables are used, and Σ^1_1-complete if periodic counting is allowed. Denoting the adjacent fragment as $\mathcal{AF}$, we provide a brief survey of complexity and undecidability standings in Table 1.

The work in this paper is closely related to [6] in which decidability of satisfiability is established for the two-variable fragment with periodic counting (denoted $\mathcal{FO}^2_{\mathrm{Pres}}$) but without a sharp complexity-theoretic bound. Our homogeneity conditions, which stem from lack of inverse relations in fluted logics, allow us to establish NExpTime-completeness for both the finite and general satisfiability problems of $\mathcal{FLPC}^2$.

Table 1 Complexity of finite (left-hand side of "/") and general (right-hand side of "/") satisfiability problems for languages (in the top row) under quantifier extensions (on the left-most column). All complexity classes are in regard to time. $\mathcal{C}$-c ($\mathcal{C}$-h) stands for complete (hard). Here $k \geq 4$ and $\ell \geq 3$.

	$\mathcal{FL}^2$	$\mathcal{FL}^\ell$	$\mathcal{AF}^3$	$\mathcal{AF}^k$
standard	NExp-c [9]	$(\ell-2)$-NExp [21]	NExp-c [4]	$(k-2)$-NExp [4]
counting	NExp-c [18]	$(\ell-1)$-NExp [19]	Σ^0_1-c/Δ^0_1 Th 11/claim	Σ^0_1-c/Π^0_1-c Th 11/15
periodic	NExp-c Th 5	$(\ell-1)$-NExp Th 9	Σ^0_1-c/Σ^0_1-h Th 11	Σ^0_1-c/Σ^1_1-c Th 11/15

2 Preliminaries

We use $\mathbb{N}$ to denote the set of integers $\{0, 1, 2, \dots\}$, and $\mathbb{N}^*$ to denote $\mathbb{N}$ along with the first infinite cardinal; i.e. $\mathbb{N}^* = \mathbb{N} \cup \{\aleph_0\}$. By picking some $n, p \in \mathbb{N}$ we write n^{+p} for the *linear set* $\{n + ip \mid i \in \mathbb{N}\}$. In the extended integers $\mathbb{N}^*$, the cardinal $\aleph_0$ is the maximum element under the canonical ordering "$<$" and

- $0 \cdot \aleph_0 = \aleph_0 \cdot 0 = 0$;
- $n + \aleph_0 = \aleph_0 + n = \aleph_0$ for all $n \in \mathbb{N}^*$; and
- $n \cdot \aleph_0 = \aleph_0 \cdot n = \aleph_0$ for all $n \in \mathbb{N}^* \setminus \{0\}$.

A *linear Diophantine inequation* is an expression of the form $a_1 v_1 + \cdots + a_n v_n + b \bowtie c_1 v_1 + \cdots + c_n v_n + d$, where $(a_i)_{i=1}^n, b, (c_i)_{i=1}^n, d$ are constant values taken form $\mathbb{N}^*$, $\bar{v} = v_1, \dots v_n$ is a vector of variables, and "$\bowtie$" is any of the relations "$=, \neq, \leq, <, \geq, >$" (each interpreted as one would assume). It is known that when the cardinal $\aleph_0$ is disallowed, a solution for a set of such inequations may be found in NPTime [17]. The picture does not change when $\aleph_0$ is permitted as a solution and/or constant. Indeed, we may reduce the problem of finding a solution over $\mathbb{N}^*$ to that of finding it over $\mathbb{N}$ as follows. First guess which variables should be mapped to $\aleph_0$ and which should have a finite value. Then, check that each inequation featuring a variable assigned $\aleph_0$ holds and discard them. What will be left is a system of inequations with constants in $\mathbb{N}$ and in variables assumed to be finite. See [20, Ch 7.4] for greater detail. We will allow systems of inequations to contain disjunctions.

By a word $\bar{a} \in A^n$ we mean a tuple $\bar{a} = a_1 \cdots a_n$, where $a_i \in A$ for each i, $1 \leq i \leq n$. In case $n = 1$, we often write a instead of $\bar{a}$. By $\bar{a}^{-1}$ we mean the reversal of $\bar{a}$; i.e. $\bar{a}^{-1} = a_n \cdots a_1$. If $\bar{a}$ and $\bar{b}$ are words we write $\bar{a}\bar{b}$ for the concatenation of the two. We use the terms "tuple" and "word" interchangeably.

Now, take some structure $\mathfrak{A}$ and an i-tuple $\bar{a}$ of elements from A. Suppose $B = \{b \in A \mid \mathfrak{A}, \bar{a}b \models \varphi\}$ for some first-order formula $\varphi(x_1, \dots, x_{i+1})$. Fixing $n, p \in \mathbb{N}$ we extend the syntax of first-order logic with *(threshold) counting quantifiers* $\exists_{[\geq n]}$ and *periodic counting quantifier* $\exists_{[n^{+p}]}$. Semantically, $\mathfrak{A}, \bar{a} \models \exists_{[\geq n]} x_{i+1} \varphi$ if and only if $|B| \geq n$. Similarly, $\mathfrak{A}, \bar{a} \models \exists_{[n^{+p}]} x_{i+1} \varphi$ if and only if $|B| \in n^{+p}$. We refrain from further generalisation to *ultimately periodic counting quantifier* $\exists_{[n_1^{+p_1} \cup \cdots \cup n_k^{+p_k}]}$ (as in [6]) as they can be expressed as a disjunction of formulas using periodic counting quantifiers $\exists_{[n_1^{+p_1}]} x_{i+1} \varphi \vee \cdots \vee \exists_{[n_k^{+p_k}]} x_{i+1} \varphi$. Thus, a sentence such as "Every orchestra hires an even number of people to play first violin" may be written in a language with periodic counting:

$$\forall x_1 \Big(orch(x_1) \to \exists_{[0^{+2}]} x_2 \big(pers(x_2) \wedge \exists x_3 (1st\_viol(x_3) \wedge hires\_to\_play(x_1, x_2, x_3)) \big) \Big). \quad (2)$$

Formally, the fluted fragment with periodic counting is the union of sets of formulas $\mathcal{FLPC}^{[\ell]}$ defined by simultaneous induction as follows:

(i) any atom $r(x_k, \dots, x_\ell)$, where $x_k, \dots, x_\ell$ is a contiguous subsequence of $x_1, x_2, \dots$ and r is a predicate of arity $\ell - k + 1$, is in $\mathcal{FLPC}^{[\ell]}$;

(ii) $\mathcal{FLPC}^{[\ell]}$ is closed under Boolean combinations;

(iii) if $\varphi \in \mathcal{FLPC}^{[\ell+1]}$, then $\exists_{[n^{+p}]} x_{\ell+1} \varphi$ is in $\mathcal{FLPC}^{[\ell]}$ for every $n, p \in \mathbb{N}$.

We write $\mathcal{FLPC} = \bigcup_{\ell \geq 0} \mathcal{FLPC}^{[\ell]}$ for the set of all fluted formulas with periodic counting and define the ℓ-variable fluted fragment with periodic counting to be the set $\mathcal{FLPC}^\ell := \mathcal{FLPC} \cap \mathcal{FO}^\ell$. (Here $\mathcal{FO}^\ell$ is the set of first-order formulas that do not use more than ℓ variables). We will implicitly restrict attention to signatures $\sigma \cup \{=\}$ which feature no function and/or constant symbols, and where "$=$" is always interpreted as the canonical equality relation. Lastly, we use $\forall x \varphi$ interchangeably with $\exists_{[0^{+0}]} x \, \neg \varphi$ whenever convenient.

Variables in fluted logics convey no meaningful information. Indeed, since the arity of every predicate in $\sigma \cup \{=\}$ is fixed and each atom features a suffix of the variable quantification order, we may employ what we call *variable-free notation*. As an example, formulas (1) and (2) may be (respectively) written as

$$\forall\Big(cond \to \exists\big(solo \wedge fav \wedge \forall(conc \to nom)\big)\Big), \tag{3}$$

$$\forall\Big(orch \to \exists_{[0^{+2}]}\big(person \wedge \exists(1st\_viol \wedge hires\_to\_play)\big)\Big) \tag{4}$$

without ambiguity (up to a shift of variable indices). Writing $\forall^\ell$ for $\forall x_1 \cdots \forall x_\ell$, we will employ variable-free notation extensively throughout subsequent sections.

Fix a first-order formula φ with periodic counting quantifiers. We assume that numeric values are encoded in binary and write $||\varphi||$ for the number of symbols used in φ. We point out that the signature of φ (which we write as $\mathrm{sig}(\varphi)$ for brevity) is no larger than $||\varphi||$.

Now, let φ be a formula in $\mathcal{FLPC}^{\ell+1}$. We say that φ is in *normal-form* if it takes the following shape

$$\bigwedge_{r\in R} \forall^\ell\Big(\alpha_r \to \exists_{[n_r^{+p_r}]}\gamma_r\Big) \wedge \bigwedge_{t\in T} \forall^\ell\Big(\beta_t \to \neg\exists_{[n_t^{+p_t}]}\delta_t\Big), \tag{5}$$

where R, T are sets of indices, each α_r, β_t is a quantifier-free formula in ℓ variables, each γ_r, δ_t is a quantifier-free formula in $\ell{+}1$ variables, and each $n_r^{+p_r}, n_t^{+p_t}$ is a linear set. Using standard rewriting techniques we prove the following in Appendix A.

▶ **Lemma 1.** *Suppose φ is an $\mathcal{FLPC}^{\ell+1}$-sentence. Then, we may compute, in polynomial time, an equisatisfiable normal-form $\mathcal{FLPC}^{\ell+1}$-sentence ψ.*

Notice that the negation before the periodic counting quantifier in the second conjunct of (5) is not moved-inwards. This is done deliberately so as to avoid computing complements of linear sets, which may be of exponential size as a function of $||\varphi||$.

Now, let $\mathfrak{A}$ be a structure over a finite signature $\sigma \cup \{=\}$. We say that an atom is $(\ell{+}1)$-fluted if it features a suffix of $x_1, \ldots, x_{\ell+1}$ as arguments. The fluted atomic $(\ell{+}1)$-type τ of $a\bar{b}c \in A^{\ell+1}$ (denoted $\mathsf{ftp}^{\mathfrak{A}}_{\ell+1}(a\bar{b}c)$) is then the set of $(\ell{+}1)$-fluted (possibly negated) atoms over $\sigma \cup \{=\}$ with arity no greater than $\ell{+}1$ that are satisfied by $a\bar{b}c$ in $\mathfrak{A}$. We invite the reader to view the tuple $a\bar{b}$ as *emitting* τ, and $\bar{b}c$ as *absorbing* τ. Write $\tau \restriction_{[2,\ell+1]}$ for the fluted ℓ-type π obtained by deleting entries in τ of arity greater than ℓ and decrementing variable indices by 1. We say that a fluted ℓ-type π is an endpoint of a fluted $(\ell{+}1)$-type τ if $\tau \restriction_{[2,\ell+1]} = \pi$. We define $\mathsf{FTP}^\sigma_{\ell+1}$ to be the set of all fluted $(\ell{+}1)$-types over $\sigma \cup \{=\}$.

The fluted ℓ-profile of $a\bar{b} \in A^\ell$ (denoted $\mathsf{fpr}^{\mathfrak{A}}_\ell(a\bar{b})$) is a function ρ mapping $\tau \in \mathsf{FTP}^\sigma_{\ell+1}$ to the number of times $a\bar{b}$ emits τ. Formally, $\rho(\tau) = |\{c \in A \mid \mathsf{ftp}^{\mathfrak{A}}_{i+1}(a\bar{b}c) = \tau\}|$. If φ is a quantifier-free $\mathcal{FLPC}^{\ell+1}$-formula, we write $\rho \models \exists_{[n^{+p}]}\varphi$ just in case $\sum_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}^{\tau\models\varphi} \rho(\tau) \in n^{+p}$. Clearly, $\rho \models \exists_{[n^{+p}]}\varphi$ if and only if $\mathfrak{A}, a\bar{b} \models \exists_{[n^{+p}]}\varphi$.

In the sequel we will assume that all structures are countable. This comes with no loss of generality as $\mathcal{FLPC}$ is subsumed by the countable fragment of infinitary logic $\mathcal{L}_{\omega_1,\omega}$; a language for which the downward Löwenheim-Skolem Theorem holds (folklore, see [15, p. 69]). Note that, in $\mathcal{FLPC}$, the finite model property fails as $\neg\exists_{[0^{+1}]}\top$ is an axiom of infinity.

3 The Two-Variable Sub-fragment

In this section we restrict attention to the two-variable fluted fragment with periodic counting. To achieve decidability of (finite) satisfiability we first specify what kind of "nice" models we will be looking for. Take any σ-structure $\mathfrak{A}$ and $\pi \in \mathsf{FTP}_1^\sigma$. For convenience, we write A_π for the set of all elements $a \in A$ with $\mathsf{ftp}^{\mathfrak{A}}(a) = \pi$. We say that π is *globally homogeneous* in $\mathfrak{A}$ if $\mathsf{fpr}^{\mathfrak{A}}(a) = \mathsf{fpr}^{\mathfrak{A}}(b)$ for each $a, b \in A_\pi$. That is to say, π is globally homogeneous in $\mathfrak{A}$ if, for each $\tau \in \mathsf{FTP}_2^\sigma$, the number of fluted 2-types τ emitted is the same for each element in A_π. The structure $\mathfrak{A}$ is *globally homogeneous* if each $\pi \in \mathsf{FTP}_1^\sigma$ is globally homogeneous in $\mathfrak{A}$.

For the remainder of the section fix some normal-form $\mathcal{FLPC}^2$-sentence φ. We claim that if φ is a satisfiable, then it is satisfiable in a globally homogeneous model. To see this, take some structure $\mathfrak{A} \models \varphi$ and a pair of elements $cd \in A^2$ that realised the fluted 2-type τ in $\mathfrak{A}$. Since τ consists of fluted formulas, it does not feature atoms of the form $p(x_1)$ and $p(x_2, x_1)$. Referencing τ only, we lack information to deduce what formulas are satisfied by the pair dc in $\mathfrak{A}$. On the other hand, $\mathfrak{A}, cd \models \psi$ if and only if $\tau \models \psi$ for each quantifier-free $\mathcal{FLPC}^2$-formula ψ. Thus, if we were to in any way alter the fluted 2-type of dc in $\mathfrak{A}$, the set of quantifier-free $\mathcal{FLPC}^2$-formulas satisfied by cd in $\mathfrak{A}$ would not change.

Taking a step back, take any $\pi \in \mathsf{FTP}_1^\sigma$ and recall that $A_\pi \subseteq A$ is the set of all elements realising the 1-type π in $\mathfrak{A}$. We redefine 2-types emitted by elements of A_π in $\mathfrak{A}$ in such a way that makes π globally homogeneous in the rewiring of $\mathfrak{A}$. Let us fix any $a \in A_\pi$ and write $\rho = \mathsf{fpr}^{\mathfrak{A}}(a)$. The element a and profile ρ picked will serve as an "example" of how the rest of A_π should form fluted 2-types with other elements of the model. Taking any $b \in A_\pi \setminus \{a\}$ and $c \in A \setminus \{a, b\}$ we see that it is impossible to prohibit b from emitting the fluted 2-type $\mathsf{ftp}^{\mathfrak{A}}(ac)$ to c via any normal-form $\mathcal{FLPC}^2$-formula. We thus allow b to impersonate a in $\mathfrak{A}$ by rewiring the fluted 2-type $\mathsf{ftp}^{\mathfrak{A}}(bc)$ to be $\mathsf{ftp}^{\mathfrak{A}}(ac)$ for each $c \in A \setminus \{a, b\}$ and, additionally, by resetting $\mathsf{ftp}^{\mathfrak{A}}(ba)$ to be the 2-type $\mathsf{ftp}^{\mathfrak{A}}(ab)$ and $\mathsf{ftp}^{\mathfrak{A}}(bb)$ to be $\mathsf{ftp}^{\mathfrak{A}}(aa)$. Clearly, only fluted 2-types emitted by b were reconsidered in this procedure, thus pairs in $(A \setminus \{b\}) \times A$ satisfy the same quantifier-free $\mathcal{FLPC}^2$-formulas as before. To see that $\mathfrak{A}$ still models φ we need only show that b does not violate $\alpha_r \to \exists_{[n_r^{+p_r}]}\gamma_r$ and $\beta_t \to \neg\exists_{[n_t^{+p_t}]}\delta_t$ for each $r \in R$ and $t \in T$. By our rewiring procedure, we have that $\mathsf{fpr}^{\mathfrak{A}}(a) = \rho = \mathsf{fpr}^{\mathfrak{A}}(b)$. Thus,

$$\mathfrak{A}, a \models \exists_{[n^{+p}]}\psi \iff \rho \models \exists_{[n^{+p}]}\psi \iff \mathfrak{A}, b \models \exists_{[n^{+p}]}\psi \qquad \text{for quantifier-free } \psi \in \mathcal{FLPC}^2.$$

Having already argued that $\mathfrak{A}, a \models \alpha_r \to \exists_{[n_r^{+p_r}]}\gamma_r$ and $\mathfrak{A}, a \models \beta_t \to \neg\exists_{[n_t^{+p_t}]}\delta_t$ for each $r \in R$ and $t \in T$ we therefore conclude that $\mathfrak{A} \models \varphi$.

Since the only 2-types redefined are those emitted by b, we can run this construction in parallel for each element in $A_\pi \setminus \{a\}$. Clearly, this renders π globally homogeneous in the rewired model. Since elements in $A \setminus A_\pi$ are left untouched by our rewiring, we can repeat this procedure for each $\pi \in \mathsf{FTP}_1^\sigma$ thus proving the following:

▶ **Lemma 2.** *Suppose φ is a satisfiable normal-form $\mathcal{FLPC}^2$-sentence. Then, φ is satisfiable in a globally homogeneous model.*

In globally homogeneous structures elements realising the same fluted 1-type are, in a sense, stripped away of their individuality as they all realise the same fluted 1-profile. When the globally homogeneous structure $\mathfrak{A}$ is clear from context, we can unambiguously write ρ_π for the fluted 1-profile realised by each element of A_π (for $\pi \in \mathsf{FTP}_1^\sigma$).

When considering the (finite) satisfiability problem for normal-form $\mathcal{FLPC}^2$-sentences such as φ, we will confine ourselves to the search of globally homogeneous models. More precisely, we will produce a system of linear Diophantine inequations Ψ that has a solution

over $\mathbb{N}^*$ if and only if φ is satisfiable in a globally homogeneous model. For this purpose, let $(x_\pi)_{\pi\in\mathsf{FTP}_1^\sigma}$, $(y_{\pi,\tau})_{\pi\in\mathsf{FTP}_1^\sigma}^{\tau\in\mathsf{FTP}_2^\sigma}$, $(i_{\pi,r})_{\pi\in\mathsf{FTP}_1^\sigma}^{r\in R}$, and $(j_{\pi,t})_{\pi\in\mathsf{FTP}_1^\sigma}^{t\in T}$ be sequences of variables. Intuitively, the value assigned to x_π will represent the number of elements realising the fluted 1-type π; $y_{\pi,\tau}$ – the number times the 2-type τ is emitted by an element realising π; and with $i_{\pi,r}$ and $j_{\pi,t}$ acting as periodic counters for elements realising π when considering linear sets $n_r^{+p_r}$ and $n_t^{+p_t}$. To be more precise, when given a satisfying assignment for Ψ, we will build a globally homogeneous $\mathfrak{A} \models \varphi$ with

$$|A_\pi| = x_\pi \text{ and } \rho_\pi(\tau) = y_{\pi,\tau} \qquad \text{for each } \pi \in \mathsf{FTP}_1^\sigma \text{ and } \tau \in \mathsf{FTP}_2^\sigma. \tag{6}$$

On the other hand, we will have that any globally homogeneous model $\mathfrak{A} \models \varphi$ gives rise to a solution of Ψ with the following assignments for all $\pi \in \mathsf{FTP}_1^\sigma$, $\tau \in \mathsf{FTP}_2^\sigma$, $r \in R$ and $t \in T$:[1]

$$x_\pi{:=}|A_\pi|;\ y_{\pi,\tau}{:=}\rho_\pi(\tau);\ i_{\pi,r}{:=}\Big(\sum_{\tau'\in\mathsf{FTP}_2^\sigma}^{\tau'\models\gamma_r}\rho_\pi(\tau'){-}n_r\Big)/p_r;\ j_{\pi,t}{:=}\Big\lfloor\Big(\sum_{\tau'\in\mathsf{FTP}_2^\sigma}^{\tau'\models\delta_t}\rho_\pi(\tau'){-}n_t\Big)/p_t\Big\rfloor. \tag{7}$$

We will proceed first by showing that the latter assignment satisfies our (yet to be defined) system of inequations $\Psi := \Psi_1 \cup \cdots \cup \Psi_6$.

When considering any model $\mathfrak{A} \models \varphi$ we have that the domain $A = \bigcup_{\pi\in\mathsf{FTP}_1^\sigma} A_\pi$ is non-empty. The following is thus trivially satisfied by the assignment $x_\pi := A_\pi$:

$$\Big\{\sum_{\pi\in\mathsf{FTP}_1} x_\pi \geq 1\Big\}. \tag{Ψ_1}$$

Additionally, picking any element $a \in A_\pi$ (for any $\pi \in \mathsf{FTP}_1^\sigma$) and picking any $\pi' \in \mathsf{FTP}_{\pi'}^\sigma$, we have that the number of fluted 2-types emitted by a to $A_{\pi'}$ must be $|A_{\pi'}| = x_{\pi'}$. Assuming that $\mathfrak{A}$ is globally homogeneous, we may fixate on the fact that the shared profile ρ_π of π has exactly $|A_{\pi'}|$ witnesses for fluted 2-types with the endpoint π', or, more formally, $\sum_{\tau\in\mathsf{FTP}_2^\sigma}^{\tau\restriction_{[2,2]}=\pi'}\rho(\tau) = |A_{\pi'}|$. Thus, the assignment $y_{\pi,\tau} := \rho_\pi(\tau)$ satisfies the following set of inequations:

$$\Big\{x_\pi \neq 0 \to \sum_{\tau\in\mathsf{FTP}_2^\sigma}^{\tau\restriction_{[2,2]}=\pi'} y_{\pi,\tau} = x_{\pi'} \;\Big|\; \pi,\pi' \in \mathsf{FTP}_1^\sigma\Big\}. \tag{Ψ_2}$$

Of course, under the supposition that $\pi \models \alpha_r$ for some $r \in R$ and $\pi \in \mathsf{FTP}_1^\sigma$ such that $|A_\pi| \geq 1$, we have that $\rho_\pi \models \exists_{[n_r^{+p_r}]}\gamma_r$. Clearly, $k := \sum_{\tau\in\mathsf{FTP}_2^\sigma}^{\tau\models\gamma_r}\rho_\pi(\tau)$ must be a member of the linear set $n_r^{+p_r}$. Thus, there is a period counter $i_{\pi,r} \in \mathbb{N}$ such that $k = n_r{+}i_{\pi,r}p_r$. Turning the equation around we get that $i_{\pi,r} = (k{-}n_r)/p_r$. Recalling that $\rho_\pi(\tau) = y_{\pi,\tau}$ for all $\tau \in \mathsf{FTP}_2^\sigma$, we have that the following is satisfied by our assignments:

$$\Big\{x_\pi \neq 0 \to \sum_{\tau\in\mathsf{FTP}_2^\sigma}^{\tau\models\gamma_r} y_{\pi,\tau} = n_r{+}i_{\pi,r}p_r \;\Big|\; r \in R, \pi \in \mathsf{FTP}_1^\sigma \text{ s.t. } \pi \models \alpha_r\Big\}. \tag{Ψ_3}$$

On the other hand, supposing $\pi \models \beta_t$ for some $t \in T$ and with $\pi \in \mathsf{FTP}_1^\sigma$ such that $|A_\pi| \geq 1$, we have that $\rho_\pi \not\models \exists_{[n_t^{+p_t}]}\delta_r$, thus leaving $k := \sum_{\tau\in\mathsf{FTP}_2^\sigma}^{\tau\models\gamma_r}\rho_\pi(\tau)$ outside of the linear set $n_t^{+p_t}$. Notice that this happens when the following conditions are met:

[1] In case p_r (resp. p_t) is 0, we allow $i_{\pi,r}$ (resp. $j_{\pi,t}$) to take any integer value.

1. $k < n_t$; or
2. $p_t \neq 0$ and $k > m$ for each $m \in n_t^{+p_t}$ (which only happens when $k = \aleph_0$); or
3. $p_t = 0$ and $k > n_t$; or
4. for some $m \in n_t^{+p_t}$ we have $m < k < m + p_t$.

Note that the listed conditions are exhaustive. We translate the requirements 1–4 into four functions $\Theta_1, \dots, \Theta_4$ which map fluted 1-types paired with indices in T to linear equations. The functions are then defined as follows:

- $\Theta_1(\pi, t) := \sum_{\tau \in \mathsf{FTP}_2^\sigma}^{\tau \models \delta_t} y_{\pi,\tau} < n_t$,
- $\Theta_2(\pi, t) := \sum_{\tau \in \mathsf{FTP}_2^\sigma}^{\tau \models \delta_t} y_{\pi,\tau} = \aleph_0$,
- $\Theta_3(\pi, t) := (p_t = 0) \wedge (n_t < \sum_{\tau \in \mathsf{FTP}_2^\sigma}^{\tau \models \delta_t} y_{\pi,\tau})$,
- $\Theta_4(\pi, t) := n_t + j_{\pi,t} p_t < \sum_{\tau \in \mathsf{FTP}_2^\sigma}^{\tau \models \delta_t} y_{\pi,\tau} < n_t + (j_{\pi,t} + 1) p_t$.

Since k adheres to at least one of the four conditions, we write the following clauses for eligible fluted 1-types:

$$\left\{ x_\pi \neq 0 \to \bigvee_{i \in [1,4]} \Theta_i(\pi, t) \;\middle|\; t \in T \text{ and } \pi \in \mathsf{FTP}_1^\sigma \text{ such that } \pi \models \beta_t \right\}. \tag{Ψ_4}$$

To verify that $j_{\pi,\tau} = \lfloor (k - n_t)/p_t \rfloor$ is indeed a satisfying assignment for Ψ_4, we need only consider case 4. For this assume that $m < k < m + p_t$ for some $m \in n_t^{+p_t}$. We can thus write $m = n_t + j p_t$. Since $j_{\pi,\tau} = \lfloor (k - n_t)/p_t \rfloor$, we conclude that $j_{\pi,\tau} = j$ thus satisfying $\Theta_4(\pi, t)$.

Reflecting on the semantics of the equality predicate, we see that for any given $\pi \in \mathsf{FTP}_1^\sigma$ in any globally homogeneous model $\mathfrak{A} \models \varphi$ there is exactly one $\tau \in \mathsf{FTP}_2^\sigma$ such that τ features the non-negated equality predicate and $\rho_\pi(\tau) \neq 0$. More precisely, $\rho_\pi(\tau) = 1$ and the endpoint of τ is π. We have that our assignment respects this condition and thus satisfies the following inequations:

$$\left\{ y_{\pi,\tau} = 0 \;\middle|\; \pi \in \mathsf{FTP}_1^\sigma,\ \tau \in \mathsf{FTP}_2^\sigma \text{ s.t. } = \in \tau, \tau \restriction_{[2,2]} \neq \pi \right\} \cup \left\{ \sum_{\tau \in \mathsf{FTP}_2^\sigma}^{= \in \tau} y_{\pi,\tau} = 1 \;\middle|\; \pi \in \mathsf{FTP}_1^\sigma \right\}. \tag{Ψ_5}$$

Finally, we forbid the periodic counters $(i_{\pi,r})_{\pi \in \mathsf{FTP}_1^\sigma}^{r \in R}$ and $(j_{\pi,t})_{\pi \in \mathsf{FTP}_1^\sigma}^{t \in R}$ from taking the value $\aleph_0$:

$$\left\{ i_{\pi,r}, j_{\pi,t} < \aleph_0 \;\middle|\; \pi \in \mathsf{FTP}_1^\sigma \text{ and } r \in R, t \in T \right\}. \tag{Ψ_6}$$

Putting everything together, we have engineered a system of equations $\Psi = \Psi_1 \cup \dots \cup \Psi_6$ that is satisfied by extracting relevant cardinalities (see (7)) from homogeneous models of φ.

▶ **Lemma 3.** *Suppose $\mathfrak{A} \models \varphi$ is a globally homogeneous model. Then, Ψ is satisfiable.*

We now move on to the converse direction:

▶ **Lemma 4.** *Suppose Ψ is satisfiable. Then, there is a globally homogeneous model $\mathfrak{A} \models \varphi$.*

Proof. Suppose that Ψ has a satisfying assignment. To avoid notational clutter, we identify the solution vectors of variables in Ψ as themselves. We will build a globally homogeneous model $\mathfrak{A}$ over the domain $A = \bigcup_{\pi \in \mathsf{FTP}_1^\sigma} A'_\pi$, where $A'_\pi = \{a_{\pi,i} \mid i \in [1, x_\pi]\}$. Intuitively, we wish that elements of A'_π realise the fluted 1-type π. We thus assign $\mathsf{ftp}_1^{\mathfrak{A}}(a) := \pi$ for all $a \in A'_\pi$. Recalling that A_π is the set of all elements that realise the 1-type π in $\mathfrak{A}$, we have that $A_\pi = A'_\pi$ is of cardinality x_π as required by (6). By Ψ_1, the domain is non-empty.

Picking any $\pi \in \mathsf{FTP}_1^\sigma$ we now move on to the assignment of fluted 2-types. Take any $\pi' \in \mathsf{FTP}_1^\sigma$ and let $S = \{\tau \in \mathsf{FTP}_2^\sigma \mid \tau \restriction_{[2,2]} = \pi'\}$, i.e. S is the set of all fluted 2-types containing the fluted 1-type π' as an endpoint. By Ψ_2, we have that $\sum_{\tau \in S} y_{\pi,\tau} = x_{\pi'}$, thus $\sum_{\tau \in S} y_{\pi,\tau} = |A_{\pi'}|$. Now, pick some element $a \in A_\pi$. In case $\pi \neq \pi'$, equation Ψ_5 prohibits fluted 2-types that feature the (non-negated) equality literal. We set fluted 2-types between a and elements of $A_{\pi'}$ in any way that results in $|\{b \in A_{\pi'} \mid \mathsf{ftp}_2^{\mathfrak{A}}(ab) = \tau\}| = y_{\pi,\tau}$ for each $\tau \in S$ (n.b. the exact configuration of fluted 2-types between a and elements of $A_{\pi'}$ is irrelevant as the fluted 2-type of ba for any $b \in A_{\pi'}$ is not set in this process). In case $\pi = \pi'$ notice that by Ψ_5 there is exactly one $\tau^= \in S$ such that (i) $= \in \tau^=$, (ii) $y_{\pi,\tau^=} \geq 0$, and with (iii) $\tau^= \restriction_{[2,2]} = \pi$. By Ψ_5 again, we have that $y_{\pi,\tau^=} = 1$. We therefore set the fluted 2-types between a and $A_\pi \setminus \{a\}$ for each $\tau \in S \setminus \{\tau^=\}$ as in the case before and, additionally, specify that $\mathsf{ftp}_2^{\mathfrak{A}}(aa) := \tau^=$.

By repeating the fluted 2-type assignment for each element $a \in A$ and fluted 1-type $\pi' \in \mathsf{FTP}_1^\sigma$ we are guaranteed that elements in A_π (where $\pi = \mathsf{ftp}^{\mathfrak{A}}(a)$) realise the fluted 1-profile $\rho_\pi := \{\tau \mapsto y_{\pi,\tau} \mid \tau \in \mathsf{FTP}_2^\sigma\}$ as required by (6). The resulting structure is clearly globally homogeneous.

We now claim that the resulting structure is a model of φ. Indeed, take any $a \in A$ with $\pi = \mathsf{ftp}_1^{\mathfrak{A}}(a)$ and suppose $\pi \models \alpha_r$ for some $r \in R$. Let $S = \{\tau \in \mathsf{FTP}_2^\sigma \mid \tau \models \gamma_r\}$. By equations Ψ_3 and Ψ_6, the sum $\sum_{\tau \in S} y_{\pi,\tau}$ is a member of the linear set $n_r^{+p_r}$. Since the element a is of fluted 1-type π, we have that it realises the profile ρ_π in $\mathfrak{A}$. By our construction, $\rho_\pi(\tau) = y_{\pi,\tau}$ for each $\tau \in \mathsf{FTP}_2^\sigma$. Thus, $\rho_\pi \models \exists_{[n_r^{+p_r}]}\gamma_r$ which is equivalent to $\mathfrak{A}, a \models \exists_{[n_r^{+p_r}]}\gamma_r$ as required.

On the other hand, suppose $\pi \models \beta_t$ for some $t \in T$. We claim that $\mathfrak{A}, a \not\models \exists_{[n_t^{+p_t}]}\delta_t$. To see this, let S be the set $\{\tau \in \mathsf{FTP}_2^\sigma \mid \tau \models \delta_t\}$. Writing $k = \sum_{\tau \in S} y_{\pi,\tau}$ we take note of equations Ψ_4 and Ψ_6, and conclude that one of the following conditions must be true:

1. k is smaller than the minimal element of $n_t^{+p_t}$; or
2. $n_t^{+p_t} \subseteq \mathbb{N}$ and $k = \aleph_0$; or
3. $p_t = 0$ and $k > n_t$; or
4. k is in between two consecutive elements of $n_t^{+p_t}$.

Whichever case it may be, we have that $k \notin n_t^{+p_t}$. Again, recalling that $\rho_\pi(\tau) = y_{\pi,\tau}$ for each $\tau \in \mathsf{FTP}_2^\sigma$, we conclude that $\rho_\pi \not\models \exists_{[n_t^{+p_t}]}\delta_t$ which is equivalent to saying $\mathfrak{A}, a \not\models \exists_{[n_t^{+p_t}]}\delta_t$. ◀

Given an $\mathcal{FLPC}^2$-sentence φ we present a decision procedure for the (finite) satisfiability problem. Compute a normal-form formula ψ from φ and write the linear Diophantine equations Ψ. Now, guess a solution vector $\bar{z}$ which can be done in non-deterministic polynomial time as a function of $||\Psi||$. If $\bar{z}$ is indeed a solution for Ψ, accept, otherwise, reject. In the case of the finite satisfiability problem, prohibit $\aleph_0$ from being a solution in Ψ. Correctness of the procedure follows from the fact that, by Lemma 2, ψ if satisfiable then it is satisfiable in a globally homogeneous model. Thus, by Lemma 3, if ψ is satisfiable, then Ψ has a solution. On the other hand, by Lemma 4, if Ψ has a solution, then ψ is satisfiable.

Noting that the satisfiability problem for $\mathcal{FL}^2$ is NExpTime-hard [21], and that $||\Psi||$ is bounded by a polynomial function on the number of different fluted 1- and 2-types (of which there are $2^{||\varphi||}$ many), we conclude the following:

▶ **Theorem 5.** *The (finite) satisfiability problem of $\mathcal{FLPC}^2$ is NExpTime-complete.*

4 More Than Two Variables

We now generalise our results on homogeneity and decidability of satisfiability for higher-arity formulas of $\mathcal{FLPC}$. Thus, throughout this section, we will be working in the $(\ell{+}1)$-variable sub-fragment of $\mathcal{FLPC}$, where $\ell \geq 2$ is fixed.

Firstly, we lift our homogeneity conditions to the multivariable setting. Suppose $\mathfrak{A}$ is a σ-structure and take any $(\ell{-}1)$-tuple $\bar{b}$ from A and $\pi \in \mathsf{FTP}^{\sigma}_{\ell}$. Let $A_{\pi \leftarrow \bar{b}}$ be the set $\{a \in A \mid \mathsf{ftp}^{\mathfrak{A}}(a\bar{b}) = \pi\}$. We say that π is $\bar{b}$-*homogeneous* in $\mathfrak{A}$ if for each $a, a' \in A_{\pi \leftarrow \bar{b}}$ and all $c \in A$ we have that $\mathsf{ftp}^{\mathfrak{A}}(a\bar{b}c) = \mathsf{ftp}^{\mathfrak{A}}(a'\bar{b}c)$. That is to say, $\bar{b}c$ absorbs the same fluted $(\ell{+}1)$-type from each ℓ-tuple $a\bar{b}$ that realises the fluted ℓ-type π. If each $\pi \in \mathsf{FTP}^{\sigma}_{\ell}$ is $\bar{b}$-homogeneous in $\mathfrak{A}$, then we say that the $(\ell{-}1)$-tuple $\bar{b}$ is *homogeneous* in $\mathfrak{A}$. Finally, if each $(\ell{-}1)$-tuple $\bar{b}$ is homogeneous in $\mathfrak{A}$, then we say that $\mathfrak{A}$ is *locally* ℓ*-homogeneous*.

When considering satisfiable normal-form $\mathcal{FLPC}^{\ell+1}$-sentences we can, without loss of generality, confine ourselves to locally ℓ-homogeneous structures. To see this fix some normal-form $\mathcal{FLPC}^{\ell+1}$-sentence φ and take $\bar{b} \in A^{\ell-1}$ and $a, a' \in A_{\pi \leftarrow \bar{b}}$. Proceeding similarly as before Lemma 2 we see that the fluted $(\ell{+}1)$-type of $a'\bar{b}c$ does not impact the satisfaction of quantifier-free $\mathcal{FLPC}^{\ell+1}$-formulas by $c\bar{b}^{-1}a'$. Thus, redefining the fluted $(\ell{+}1)$-types emitted by $a'\bar{b}$ will not alter the satisfaction of quantifier-free $\mathcal{FLPC}^{\ell+1}$-formulas by other tuples. Notice again that it is impossible to prohibit $\mathsf{ftp}^{\mathfrak{A}}(a\bar{b}c) \neq \mathsf{ftp}^{\mathfrak{A}}(a'\bar{b}c)$ by a normal-form formula. Thus, by setting $\mathsf{ftp}^{\mathfrak{A}}(a'\bar{b}c) := \mathsf{ftp}^{\mathfrak{A}}(a\bar{b}c)$ for each $c \in A$ and repeating the procedure for each $a' \in A_{\pi \leftarrow \bar{b}} \setminus \{a\}$, we will have that π is $\bar{b}$-*homogeneous* in $\mathfrak{A}$. Clearly $\mathfrak{A} \models \varphi$ as the tuples rewired by this procedure (i.e. $a'\bar{b}$ with $a' \in A_{\pi \leftarrow \bar{b}} \setminus \{a\}$) now emit $\tau \in \mathsf{FTP}^{\sigma}_{\ell+1}$ to a given witness if and only if $a\bar{b}$ does. The rewired tuples thus satisfy the same exact $\mathcal{FLPC}^{\ell+1}$-formulas with at most 1 quantifier as $a\bar{b}$ does. Repeating the procedure for all $\pi \in \mathsf{FTP}^{\sigma}_{\ell}$ and $\bar{b} \in A^{\ell-1}$ we will have the following:

▶ **Lemma 6.** *Suppose φ is a satisfiable normal-form $\mathcal{FLPC}^{\ell+1}$-sentence. Then, φ is satisfiable in a locally ℓ-homogeneous model.*

Using local ℓ-homogeneity coupled with variable reduction techniques prevalent in studies of fluted logics (see [21]), we will establish a decidability result for the (finite) satisfiability problem of $\mathcal{FLPC}^{\ell+1}$. More specifically, fixing a normal-form $\mathcal{FLPC}^{\ell+1}$-sentence φ we will compute a normal-form $\mathcal{FLPC}^{\ell}$-sentence ψ that is satisfiable in structures holding just enough information to build locally ℓ-homogeneous models for φ. To aid motivation, we fix $\mathfrak{A}$ to be any locally ℓ-homogeneous model of φ. We shall construct ψ whilst also expanding $\mathfrak{A}$ into $\mathfrak{A}' \models \psi$. Note that the construction depends exclusively on the syntactic properties of φ.

First, set $\mathfrak{A}' := \mathfrak{A}$. Take $(q_\pi)_{\pi \in \mathsf{FTP}^{\sigma}_{\ell}}$ to be a sequence of fresh $(\ell{-}1)$-ary predicate symbols. In $\mathfrak{A}'$ we decorate $(\ell{-}1)$-tuples $\bar{b}$ over A with q_π just in case we have that $A_{\pi \leftarrow \bar{b}} \neq \emptyset$. That is to say, $q_\pi^{\mathfrak{A}'}$ remembers which $(\ell{-}1)$-tuples can be extended (by appending an element to the left) to realise the fluted ℓ-type π. It is clear that $\mathfrak{A}'$ models the following:

$$\bigwedge_{\pi \in \mathsf{FTP}^{\sigma}_{\ell}} \forall^{\ell}\Big(\pi \to q_\pi\Big). \tag{ψ_1}$$

Proceeding similarly, let $(s_{\pi,\tau})^{\tau \in \mathsf{FTP}^{\sigma}_{\ell+1}}_{\pi \in \mathsf{FTP}^{\sigma}_{\ell}}$ be a sequence of new ℓ-ary predicates. Intuitively, we will have $\bar{b}c \in s^{\mathfrak{A}'}_{\pi,\tau}$ if in $\mathfrak{A}$ it is the case that $\bar{b}c$ absorbs the fluted $(\ell{+}1)$-type τ emitted from $a\bar{b}$ for some $a \in A_{\pi \leftarrow \bar{b}}$. Notice that, by local ℓ-homogeneity, if $\bar{b}c$ absorbs τ from some $a\bar{b}$ with $a \in A_{\pi \leftarrow \bar{b}}$, then it absorbs τ from $a'\bar{b}$ for all $a' \in A_{\pi \leftarrow \bar{b}}$. Thus, by our construction, $s_{\pi,\tau}$ is the unique predicate amongst $(s_{\pi,\tau'})^{\tau' \in \mathsf{FTP}^{\sigma}_{\ell+1}}$ satisfied by $\bar{b}c$ in $\mathfrak{A}'$. Clearly, $\mathfrak{A}'$ models:

$$\bigwedge_{\pi\in\mathsf{FTP}^\sigma_\ell}\bigwedge_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}\forall^\ell\quad\Big(s_{\pi,\tau}\to\tau\restriction_{[2,\ell+1]}\Big)\wedge$$

$$\bigwedge_{\pi\in\mathsf{FTP}^\sigma_\ell}\forall^{\ell-1}\Big(q_\pi\to\forall\big(\bigvee_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}s_{\pi,\tau}\wedge\bigwedge_{\tau,\tau'\in\mathsf{FTP}^\sigma_{\ell+1}}^{\tau\neq\tau'}(\neg s_{\pi,\tau}\vee\neg s_{\pi,\tau'})\big)\Big).\tag{ψ_2}$$

Again taking $\bar{b}\in A^{\ell-1}$ and any $\pi\in\mathsf{FTP}^\sigma_\ell$ suppose $\pi\models\alpha_r$ for some $r\in R$. In case $A_{\pi\leftarrow\bar{b}}$ is non-empty (thus guaranteeing $\bar{b}\in q^{\mathfrak{A}'}_\pi$), we pick any $a\in A_{\pi\leftarrow\bar{b}}$ and write $S=\{c\in A\mid\mathfrak{A},a\bar{b}c\models\gamma_r\}$. Since π is $\bar{b}$-homogeneous in $\mathfrak{A}$, the exact element in $A_{\pi\leftarrow\bar{b}}$ we pick has no effect on S. By our construction, S is then exactly the set of element $c\in A$ such that $\mathfrak{A}',\bar{b}c\models\bigvee^{\tau\models\gamma_r}_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}s_{\pi,\tau}$. Since $|S|\in n_r^{+p_r}$ it is then immediate that $\mathfrak{A}'$ models the following:

$$\bigwedge_{r\in R}\bigwedge^{\pi\models\alpha_r}_{\pi\in\mathsf{FTP}^\sigma_\ell}\forall^{\ell-1}\Big(q_\pi\to\exists_{[n_r^{+p_r}]}\bigvee^{\tau\models\gamma_r}_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}s_{\pi,\tau}\Big).\tag{ψ_3}$$

Similar observations follow whenever $\pi\models\beta_t$ for some $t\in T$. This time, however, the cardinality of $S=\{c\in A\mid\mathfrak{A},a\bar{b}c\models\delta_r\}$ must be outside the set $n_r^{+p_r}$. Clearly, $\mathfrak{A}'$ models:

$$\bigwedge_{t\in T}\bigwedge^{\pi\models\beta_t}_{\pi\in\mathsf{FTP}^\sigma_\ell}\forall^{\ell-1}\Big(q_\pi\to\neg\exists_{[n_t^{+p_t}]}\bigvee^{\tau\models\delta_t}_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}s_{\pi,\tau}\Big).\tag{ψ_4}$$

Writing $\psi:=\psi_1\wedge\cdots\wedge\psi_4$ we have shown the following:

▶ **Lemma 7.** *Suppose $\mathfrak{A}\models\varphi$ is a locally ℓ-homogeneous model. Then, $\mathfrak{A}$ can be extended to a model $\mathfrak{A}'$ of ψ.*

▶ **Lemma 8.** *Suppose $\mathfrak{A}'\models\psi$. Then, we can construct a locally ℓ-homogeneous model $\mathfrak{A}^+$ of φ over the same domain.*

Proof. Supposing ψ is satisfiable we take any model $\mathfrak{A}'$. Now, let $\mathfrak{A}^-$ be the model $\mathfrak{A}'$ but with the predicates in $(q_\pi)_{\pi\in\mathsf{FTP}^\sigma_\ell}$ and $(s_{\pi,\tau})^{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}_{\pi\in\mathsf{FTP}^\sigma_\ell}$ removed from the signature. We proceed by expanding $\mathfrak{A}^-$ into a locally ℓ-homogeneous model $\mathfrak{A}^+$ of the original sentence φ.

Fix $\bar{b}\in A^{\ell-1}$ and take some $a\in A$. Supposing that $\mathsf{ftp}^{\mathfrak{A}^-}_\ell(a\bar{b})=\pi$, by ψ_1 we have that $\mathfrak{A}',\bar{b}\models q_\pi$. Taking any $c\in A$ we observe that the conjuncts of ψ_2 enforce the following:

- if $\mathfrak{A}',\bar{b}c\models s_{\pi,\tau}$ for some $\tau\in\mathsf{FTP}^\sigma_{\ell+1}$, then $\bar{b}c$ can absorb the fluted $(\ell{+}1)$-type τ,
- $\bar{b}c$ satisfies at least one of the predicates $(s_{\pi,\tau})_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}$, and
- $\bar{b}c$ satisfies at most one of the predicates $(s_{\pi,\tau})_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}$.

We can then safely set $\mathsf{ftp}^{\mathfrak{A}^+}_{\ell+1}(a\bar{b}c):=\tau$ for each $c\in A$, where τ is taken from the subscript of the unique $s_{\pi,\tau}\in(s_{\pi,\tau})_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}$ that $\bar{b}c$ satisfies in $\mathfrak{A}'$. By repeating the above procedure for all $a\in A$ and tuples $\bar{b}\in A^{\ell-1}$ we will obtain the desired structure $\mathfrak{A}^+$.

To verify that $\mathfrak{A}^+$ is a model of φ we first claim that $\mathfrak{A}^+\models\bigwedge_{r\in R}\forall^\ell(\alpha_r\to\exists_{[n_r^{+p_r}]}\gamma_r)$. For this purpose, fix some $r\in R$ and $a\bar{b}\in A^\ell$, and suppose $\pi\models\alpha_r$, where $\mathsf{ftp}^{\mathfrak{A}^+}_\ell(a\bar{b})=\pi$. Recall that $\bar{b}\in q^{\mathfrak{A}'}_\pi$ by ψ_1. Then, ψ_3 gives us $\mathfrak{A}',\bar{b}\models\exists_{[n_r^{+p_r}]}\bigvee^{\tau\models\gamma_r}_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}s_{\pi,\tau}$. Taking any $c\in A$ we have, by our construction, that $\mathfrak{A}^+,a\bar{b}c\models\tau$ if and only if $\mathfrak{A}',\bar{b}c\models s_{\pi,\tau}$. Thus, $\mathfrak{A}^+,a\bar{b}\models\exists_{[n_r^{+p_r}]}\bigvee^{\tau\models\gamma_r}_{\tau\in\mathsf{FTP}^\sigma_{\ell+1}}\tau$, which is equivalent to saying $\mathfrak{A}^+,a\bar{b}\models\exists_{[n_r^{+p_r}]}\gamma_r$. Repeating the argument for each $r\in R$ and $a\bar{b}\in A^\ell$ will yield the required result.

To show $\mathfrak{A}^+\models\bigwedge_{t\in T}\forall^\ell(\beta_t\to\neg\exists_{[n_t^{+p_t}]}\delta_t)$ we proceed analogously with ψ_4 in place of ψ_3. ◀

Let us take stock of the previous three lemmas. Take φ to be an $\mathcal{FLPC}^{\ell+1}$-sentence. Without loss of generality, assume that it is in normal-form (Lemma 1). By Lemma 6, φ is satisfiable if it is satisfiable in a locally ℓ-homogeneous model. By computing the formula ψ we have, by Lemmas 7 and 8, that ψ is (finitely) satisfiable if and only if φ is. Noting that the (finite) satisfiability problem for $\mathcal{FLPC}^2$ is in NExpTime (Theorem 5) and that ψ can be constructed from φ in polynomial time in regards to the number of different fluted ℓ- and $(\ell{+}1)$-types (of which there are $2^{O(||\varphi||)}$), we conclude the following:

▶ **Theorem 9.** *The (finite) satisfiability problem for $\mathcal{FLPC}^{\ell+1}$ is in ℓ-NExpTime.*

Noting that the (finite) satisfiability problem for $\mathcal{FL}^{\ell+1}$ is $\lfloor(\ell{+}1)/2\rfloor$-NExpTime-hard [21], we see that no elementary function can encapsulate the complexity of (finite) satisfiability for $\mathcal{FLPC}$. We thus conclude our section having reached our initial goal:

▶ **Theorem 10.** *The (finite) satisfiability problem for $\mathcal{FLPC}$ is Tower-complete.*

5 Counting With Reversed Relations

In this section we will show that relaxing the syntactic restrictions of the fluted fragment with counting yields undecidability of satisfiability. We define the language $\mathcal{FL}_{\text{rev}}$ to be $\mathcal{FL}$ but with the addition of atoms with reversed variable sequences. More formally, if $r(x_k, \dots, x_\ell)$ is an $\mathcal{FL}$-atom, then $r(x_k, \dots, x_\ell)$ and $r(x_\ell, \dots, x_k)$ are $\mathcal{FL}_{\text{rev}}$-atoms. The language $\mathcal{FLC}_{\text{rev}}$ ($\mathcal{FLPC}_{\text{rev}}$) is then the obvious extension of $\mathcal{FL}_{\text{rev}}$ with (periodic) counting quantifiers. Clearly, the languages $\mathcal{FL}_{\text{rev}}$, $\mathcal{FLC}_{\text{rev}}$, and $\mathcal{FLPC}_{\text{rev}}$ are subfragments of the adjacent fragment with the appropriate counting extensions. We will use counting quantifiers $\exists_{[=1]}$ and $\exists_{[\leq 1]}$ with the meanings "there is exactly one element s.t. ..." and "there is at most one element s.t. ..." along side periodic counting qunatifiers. For simplicity, we do away with variable-free notation and use variable sequences of x, y, z and z, y, x in place of x_1, x_2, x_3.

We proceed by reducing *Hilbert's 10th problem* to the finite satisfiability problem of $\mathcal{FLC}^3_{\text{rev}}$. Let $\mathcal{E}$ be a system of Diophantine equations. We assume that each equation $e \in \mathcal{E}$ is of one of the following (simple) forms: (i) $u = 1$, (ii) $u{+}v = w$, or (iii) $u \cdot v = w$, where u, v, w are mutually disjoint variables. Clearly, no loss of generality occurs as any (non-simple) Diophantine equation can be rewritten into the simpler form by introducing new variables. For each $e \in \mathcal{E}$ we will define a formula φ_e depending on the form that e takes. Then, $\varphi := \bigwedge_{e\in\mathcal{E}} \varphi_e \wedge \psi$ will be the advertised formula that is finitely satisfiable if and only if $\mathcal{E}$ has a solution over $\mathbb{N}$. We specify that the signature of φ includes (1) unary predicates A_u for each variable of u in $\mathcal{E}$, (2) binary predicates R_e for each $e \in \mathcal{E}$ of the form (ii), and (3) ternary predicates P_e for each $e \in \mathcal{E}$ of the form (iii). We will not assume that the equality predicate is available. In the sequel we will argue that if $\mathfrak{A} \models \varphi$, then $\mathcal{E}$ has a solution with $u \mapsto |A^{\mathfrak{A}}_u|$ for each variable u. (And, of course, the converse as well). For technical reasons we wish for the sets $A^{\mathfrak{A}}_u$ and $A^{\mathfrak{A}}_v$ with $u \neq v$ to be disjoint. Denoting $\text{vars}(\mathcal{E})$ for the set of variables in $\mathcal{E}$, we first define $\psi := \bigwedge^{u\neq v}_{u,v\in\text{vars}(\mathcal{E})} \forall x\big(\neg A_u(x) \vee \neg A_v(x)\big)$, which clearly has the required effect. We proceed by taking $e \in \mathcal{E}$ in turn.

Suppose first that e is of the form (i) $u = 1$. We ensure that every model $\mathfrak{A}$ of φ will have $|A^{\mathfrak{A}}_u| = 1$ by defining φ_e to be $\exists_{[=1]}x\ A_u(x)$.

Now, supposing that e is of the form (ii) $u{+}v = w$, we define φ_e with the intent that models $\mathfrak{A}$ of φ will have $R^{\mathfrak{A}}_e$ being a bijection between $A^{\mathfrak{A}}_u \cup A^{\mathfrak{A}}_v$ and $A^{\mathfrak{A}}_w$ (i.e. $|A^{\mathfrak{A}}_u| + |A^{\mathfrak{A}}_v| = |A^{\mathfrak{A}}_w|$):

$$\forall x\Big(\big(A_u(x) \vee A_v(x)\big) \to \exists_{[=1]}y\big(A_w(y) \wedge R_e(xy)\big)\Big) \wedge$$
$$\forall y\Big(A_w(y) \to \exists_{[=1]}x\big((A_u(x) \vee A_v(x)) \wedge R_e(xy)\big)\Big).$$

Lastly, if e is of the form (iii) $u \cdot v = w$, then φ_e (defined just below) will guarantee that $\mathfrak{A} \models \varphi$ forces $P_e^{\mathfrak{A}}$ to be a bijection between $A_u^{\mathfrak{A}} \times A_v^{\mathfrak{A}}$ and $A_w^{\mathfrak{A}}$ (i.e. $|A_u^{\mathfrak{A}}| \cdot |A_v^{\mathfrak{A}}| = |A_w^{\mathfrak{A}}|$):

$$\forall x\Big(A_u(x) \to \forall y\big(A_v(y) \to \exists_{[=1]} z(A_w(z) \wedge P_e(xyz))\big)\Big) \wedge$$
$$\forall z\Big(A_w(z) \to \exists_{[=1]} y\big(A_v(y) \wedge \exists x(A_u(x) \wedge P_e(xyz))\big)\Big) \wedge$$
$$\forall z\Big(A_w(z) \to \exists y\big(A_v(y) \wedge \exists_{[=1]} x(A_u(x) \wedge P_e(xyz))\big)\Big).$$

It is then straightforward to show (see Appendix B) that φ is finitely satisfiable iff $\mathcal{E}$ has a solution over $\mathbb{N}$. Noting that the problem of finding solutions to Diophantine equations over $\mathbb{N}$ is Σ_1^0-complete, and that $\varphi \wedge \exists_{[0^{+1}]} x\ \top$ is an $\mathcal{FLPC}^3_{\text{rev}}$-sentence that is satisfiable if and only if φ is finitely satisfiable, we conclude the following:

▶ **Theorem 11.** *The finite satisfiability problem for $\mathcal{FLC}^3_{rev}$ is Σ_1^0-hard. If periodic counting is permitted, then so is the general satisfiability problem.*

We note that one can use the same type of argument as above when reducing from the problem of solving Diophantine equations $\mathcal{E}$ over $\mathbb{N}^* = \{1, 2, \dots\} \cup \{\aleph_0\}$ to the general satisfiability problem of $\mathcal{FLC}+\cdot^{-1}$. Such an approach, however, is not fruitful for determining undecidability as the problem of finding solutions to $\mathcal{E}$ over $\mathbb{N}^*$ is in NPTime [14]. Thus, to show undecidability of general satisfiability, we resort to the tiling problem. Take $\Phi = \langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle$ with $\mathcal{H}, \mathcal{V} \subseteq \mathcal{T} \times \mathcal{T}$. We will produce and $\mathcal{FLC}^4_{\text{rev}}$-sentence φ that is satisfiable if and only if Φ tiles the infinite $\mathbb{N}\times\mathbb{N}$ plane in accordance to the horizontal ($\mathcal{H}$) and vertical ($\mathcal{V}$) constraints. Additionally, we will argue that one can append additional $\mathcal{FLPC}^2_{\text{rev}}$ conjuncts to φ and thus obtain a reduction from the tiling problem with a designated tile recurring infinitely often on the first column (see [10] for details about the problem). All-in-all, such reductions will guarantee that $\mathcal{FLC}^4_{\text{rev}}$ is Π_1^0-hard, whilst making $\mathcal{FLPC}^4_{\text{rev}}$ hard for Σ_1^1.

Take G to be unary and H, V to be binary predicate symbols. We define the *canonical* ($\mathbb{N}\times\mathbb{N}$)*-grid* to be a $\{G, H, V\}$-structure $\mathfrak{G}$ over the domain $\mathbb{N}\times\mathbb{N}$ with the following extensions:

- $G^{\mathfrak{A}} := \mathbb{N}\times\mathbb{N}$,
- $H^{\mathfrak{A}} := \{\langle (i,j), (i,j{+}1)\rangle \mid i, j \in \mathbb{N}\}$, and
- $V^{\mathfrak{A}} := \{\langle (i,j), (i{+}1,j)\rangle \mid i, j \in \mathbb{N}\}$.

We say that a structure $\mathfrak{A}$ is a ($\mathbb{N}\times\mathbb{N}$)*-grid* if $\mathfrak{A}$ restricted to elements $G^{\mathfrak{A}}$ and the signature $\{G, H, V\}$ is isomorphic to the canonical ($\mathbb{N}\times\mathbb{N}$)-grid. More leniently, $\mathfrak{A}$ is *grid-like* if it contains a homomorphic embedding of $\mathfrak{G}$. It is well known that the satisfiability problem posed over subclasses of grid-like structures is undecidable for even inexpressive logics such as $\mathcal{FL}^2$. The following is almost immediate:

▶ **Lemma 12.** *The satisfiability problem for $\mathcal{FLC}^2_{rev}$ posed over subclasses of grid-like structures is Π_1^0-hard. The satisfiability problem for $\mathcal{FLPC}^2_{rev}$ posed over subclasses of ($\mathbb{N}\times\mathbb{N}$)-grids is Σ_1^1-hard.*

The lemma above is, of course, the "easy" part of a much larger reduction. The axiomatisation of grid-like structures and ($\mathbb{N}\times\mathbb{N}$)-grids is where the expressive power of $\mathcal{FLC}^4_{\text{rev}}$ and $\mathcal{FLPC}^4_{\text{rev}}$ is needed. Before writing the advertised formulas, we build the motivating structure we will be looking for in three steps. Suppose $\mathfrak{G}$ is the canonical ($\mathbb{N}\times\mathbb{N}$)-grid. Letting E_H, E_V be binary and O be unary predicate symbols we define the *graphed expansion of* $\mathfrak{G}$ to be the structure $\mathfrak{G}^+$ over the domain $(\mathbb{N}\times\mathbb{N}) \cup \mathbb{N}$ with the following extensions:

- $\mathfrak{G}^+{\restriction_{\mathbb{N}\times\mathbb{N}}} := \mathfrak{G}$,
- $k \notin G^{\mathfrak{G}^+}$ for each $k \in \mathbb{N}$,
- $k \in O^{\mathfrak{G}^+}$ if and only if $k = 0$,
- $E_H^{\mathfrak{G}^+} := \bigcup_{i,j\in\mathbb{N}}\{\langle (i,j), k\rangle \mid 1 \leq k \leq i\}$, and
- $E_V^{\mathfrak{G}^+} := \bigcup_{i,j\in\mathbb{N}}\{\langle (i,j), k\rangle \mid 1 \leq k \leq j\}$.

Intuitively, $\mathfrak{G}^+$, when restricted to $\mathbb{N}\times\mathbb{N} = G^{\mathfrak{G}^+}$, is the canonical $(\mathbb{N}\times\mathbb{N})$-grid. Notice that each $(i,j) \in \mathbb{N}\times\mathbb{N}$ has i elements in $\mathbb{N}$ that are E_H-successors and j elements that are E_V-successors. In other words, the coordinates of (i,j) are explicitly encoded in $\mathfrak{G}^+$ as the out-degrees of E_H and E_V respectively. (In the future, we will simply speak of E_H- and E_V-degree with "out" being left implicit). We invite the reader to regard $\mathbb{N}$ as the set of extra elements which help encode positions of grid elements. Notice that the singleton $0 \in O^{\mathfrak{G}^+}$ is not featured in any binary relations (most notably, E_H and E_V). This is deliberate as it will act as a *spare part* in the constructions to come.

We now define *the mapped expansion* $\mathfrak{G}^\star$ *of* $\mathfrak{G}^+$, where $\mathfrak{G}^+$ itself is the graphed expansion of $\mathfrak{G}$. For this, we introduce quaternary R_H, R_V, S_H, S_V and ternary predicates C_H, C_V, whilst setting the following extensions:

- $R_H^{\mathfrak{G}^\star} := \bigcup_{i,j,i',j'\in\mathbb{N}}^{i\leq i'}\{\langle k, (i,j), (i',j'), k\rangle \mid 1 \leq k \leq i\}$,
- $R_V^{\mathfrak{G}^\star} := \bigcup_{i,j,i',j'\in\mathbb{N}}^{j\leq j'}\{\langle k, (i,j), (i',j'), k\rangle \mid 1 \leq k \leq j\}$,
- $S_H^{\mathfrak{G}^\star} := \bigcup_{i,j,j'\in\mathbb{N}}\{\langle k-1, (i,j), (i{+}1,j'), k\rangle \mid 1 \leq k \leq i+1\}$,
- $S_V^{\mathfrak{G}^\star} := \bigcup_{i,j,i'\in\mathbb{N}}\{\langle k-1, (i,j), (i',j{+}1), k\rangle \mid 1 \leq k \leq j+1\}$,
- $C_H^{\mathfrak{G}^\star} := \bigcup_{i,j,i',j'\in\mathbb{N}}^{i\leq i'}\{\langle k, (i',j'), (i,j)\rangle \mid 1 \leq k \leq i\}$, and
- $C_V^{\mathfrak{G}^\star} := \bigcup_{i,j,i',j'\in\mathbb{N}}^{j\leq j'}\{\langle k, (i',j'), (i,j)\rangle \mid 1 \leq k \leq j\}$.

Recall that each $(i,j) \in \mathbb{N}\times\mathbb{N}$ sends E_H-edges to each $k \in [1,i]$. Fixing some $(i',j') \in \mathbb{N}\times\mathbb{N}$ with $i \leq i'$, we have that $R_H^{\mathfrak{G}^\star}$ injectively maps E_H-edges originating from (i,j) to E_H-edges of (i',j'). On the other hand, $S_H^{\mathfrak{G}^\star}$ is a bijection between the E_H-edges of (i,j) with the spare part 0 and E_H-edges of $(i{+}1,j')$. The relation $C_H^{\mathfrak{G}^\star}$ simply remembers which E_H-edges of (i',j') are mapped to E_H-edges of (i,j) via $R_H^{\mathfrak{A}}$. Relations $R_V^{\mathfrak{G}^\star}$, $S_V^{\mathfrak{G}^\star}$ and $C_V^{\mathfrak{G}^\star}$ act similarly.

Lastly, We say that $\mathfrak{G}^\#$ *is the ordered expansion* of $\mathfrak{G}^\star$, where $\mathfrak{G}^\star$ itself is the graphed and mapped expansion of $\mathfrak{G}$, if the signature contains two additional relations $\preceq_H$ and $\preceq_V$ which we will define to be total orders over $\mathbb{N}\times\mathbb{N}$. For motivational purposes, we will forget that grid elements a and b are pairs of natural numbers and instead focus on the E_H- and E_V-degrees of the elements. In $\mathfrak{G}^\#$ we will have that

- $a \preceq_H^{\mathfrak{G}^\#} b$ if and only if the E_H-degree of a is no more than that of b, and
- $a \preceq_V^{\mathfrak{G}^\#} b$ if and only if the E_V-degree of a is no more than that of b.

(Again, the E_H- and E_V-degrees encode the horizontal and vertical positions of the element).

We will now write the sentence $\varphi := \varphi_1 \wedge \cdots \wedge \varphi_{13}$ one conjunct at a time. At a high level, the conjuncts simply state facts about the graphed mapped and ordered expansion $\mathfrak{G}^\#$ of $\mathfrak{G}$. In the sequel we will argue that the satisfaction of φ by $\mathfrak{A}$ is sufficient to deduce that the structure in question is grid-like. For readability, we will be using variable sequences x,y,z,w and w,z,y,x instead of x_1,x_2,x_3,x_4. Strictly speaking, the formulas to be defined are not (reverse) fluted, but can be made such by moving quantifiers inwards.

Fix $\mathfrak{G}^\#$ to be as described above. We first capture some graphed properties. Recall that in $\mathfrak{G}^\#$ there is a single spare part element $0 \in O^{\mathfrak{G}^\#}$. Noting that this element is not part of the grid $(0 \notin G^{\mathfrak{G}^\#})$ we have that $\mathfrak{G}^\#$ models:

$$\exists_{[=1]}x\, O(x) \wedge \forall x\big(O(x) \to \neg G(x)\big) \tag{φ_1}$$

Additionally, recall that the spare part 0 has no incoming edges E_H- or E_V-edges in $\mathfrak{G}^{\#}$. Thus, $\mathfrak{G}^{\#}$ also models:

$$\forall x\Big(O(x) \to \forall y\big(\neg E_H(yx) \wedge \neg E_V(yx)\big)\Big) \tag{φ_2}$$

Moving to grid elements, we see that there is a single element in $G^{\mathfrak{A}^{\#}}$ with no E_H- or E_V-degree. Thus, $\mathfrak{G}^{\#}$ is a model of:

$$\exists_{[=1]}x\Big(G(x) \wedge \forall y\big(\neg E_H(xy) \wedge \neg E_V(xy)\big)\Big). \tag{φ_3}$$

Additionally, each element in $G^{\mathfrak{G}^{\#}}$ has a single H- and V-successor. Thus, $\mathfrak{G}^{\#}$ models:

$$\forall x\Big(G(x) \to \Big(\exists_{[=1]}y\big(H(xy) \wedge G(y)\big) \wedge \exists_{[=1]}y\ \big(V(xy) \wedge G(y)\big)\Big)\Big). \tag{φ_4}$$

For the next two conjuncts fix $(i,j) \in G^{\mathfrak{G}^{\#}}$. Notice that the H-successor $(i{+}1,j)$ has an E_H-degree that is larger by 1 when compared to its predecessor (i,j). We can thus map the E_H-edges from $(i{+}1,j)$ to the set of E_H-edges from (i,j) taken together with the spare part element bijectively. This is exactly how the extension to S_H in $\mathfrak{G}^{\#}$ is set up. Noting that V-successors have analogous properties we conclude that $\mathfrak{G}^{\#}$ models the following sentences:

$$\bigwedge_{X\in\{H,V\}} \forall xyz\Big(\big((E_X(yx) \vee O(x)) \wedge X(yz)\big) \to \exists_{[=1]}w\big(E_X(zw) \wedge S_X(xyzw)\big)\Big), \tag{φ_5}$$

$$\bigwedge_{X\in\{H,V\}} \forall wzy\Big(\big(E_X(zw) \wedge X(yz)\big) \to \exists_{[=1]}x\big((E_X(yx) \vee O(x)) \wedge S_X(xyzw)\big)\Big). \tag{φ_6}$$

Recall that for grid elements $(i,j),(i',j') \in G^{\mathfrak{G}^{\#}}$ we have $\mathfrak{G}^{\#} \models (i,j) \preceq_V (i',j')$ if and only if the E_V-degree of (i,j) is no more than that of (i',j'). Fixing (i,j) and its H-successor $(i{+}1,j)$ we see that $\mathfrak{G}^{\#} \models (i,j) \preceq_V (i{+}1,j) \wedge (i{+}1,j) \preceq_V (i,j)$. That is, (i,j) and its H-successor have the same E_V-degrees. Thus, $\mathfrak{G}^{\#}$ models:

$$\bigwedge_{X,Y\in\{H,V\}}^{X\neq Y} \forall xy\Big(X(xy) \to \big(x \preceq_Y y \wedge y \preceq_Y x\big)\Big). \tag{φ_7}$$

Now, recall that the ordering $\preceq_X$ $(X \in \{H,V\})$ is total on $G^{\mathfrak{G}^{\#}}$. Thus, $\mathfrak{G}^{\#}$ models:

$$\bigwedge_{X\in\{H,V\}} \forall xy\ \Big(\big(G(x) \wedge G(y)\big) \to \big(x \preceq_X y \vee y \preceq_X x\big)\Big). \tag{φ_8}$$

Taking $(i,j),(i',j') \in G^{\mathfrak{G}^{\#}}$ with $i \leq i'$ recall that the elements have E_H-edges mapped injectively by $R_H^{\mathfrak{G}^{\#}}$. Thus, $\mathfrak{G}^{\#} \models (i,j) \preceq_H (i',j')$ if and only if $R_H^{\mathfrak{G}^{\#}}$ is an injection between the E_H-edges of (i,j) and that of (i',j'). We capture the "only-if" direction of the dependency with the sentences φ_9 and φ_{10}, whilst the "if" direction is handled by φ_{11} and φ_{12}.

Still holding the supposition that $\mathfrak{G}^{\#} \models (i,j) \preceq_H (i',j')$, we write a sentence ensuring that each E_H-edge from (i,j) is mapped to some single E_H-edge from (i',j'):

$$\bigwedge_{X\in\{H,V\}} \forall xyz\Big(\big(y \preceq_X z \wedge E_X(yx)\big) \to \exists_{[=1]}w\big(E_X(zw) \wedge R_X(xyzw)\big)\Big). \tag{φ_9}$$

With the next sentence we require that each E_H-edge from (i',j') is a witness (in regard to $R_H^{\mathfrak{G}^{\#}}$) to at most a single E_H-edge from (i,j):

$$\bigwedge_{X\in\{H,V\}} \forall wzy\Big(\big(y \preceq_X z \wedge E_X(zw)\big) \to \exists_{[\leq 1]}x\big(E_X(yx) \wedge R_X(xyzw)\big)\Big). \tag{φ_{10}}$$

It is easy to verify that this is indeed how $R_H^{\mathfrak{G}^\#}$ is set up. Noting that $\preceq_V$ and R_V behave symmetrically we conclude $\mathfrak{G}^\# \models \varphi_9 \wedge \varphi_{10}$.

For the converse direction of the implication take any $(i,j),(i',j') \in G^{\mathfrak{G}^\#}$ and recall that $\langle k,(i',j'),(i,j)\rangle \in C_H^{\mathfrak{G}^\#}$ if and only if there is some $k' \in \mathbb{N}$ for which $\langle k',(i,j),(i',j'),k\rangle \in R_H^{\mathfrak{G}^\#}$. In other words, $C_H^{\mathfrak{G}^\#}$ remembers which E_H-edges from (i',j') are featured in a mapping (by $R_H^{\mathfrak{G}^\#}$) with E_H-edges from (i,j). We axiomatise this relationship as follows:

$$\bigwedge_{X\in\{H,V\}} \forall wzy\Big(C_X(wzy) \leftrightarrow \exists_{[=1]}x\big(E_X(yx) \wedge R_X(xyzw)\big)\Big). \tag{φ_{11}}$$

Utilising C_H we can then test if each E_H-edge of (i',j') is mapped to some E_X-edge of (i,j) and, if that is indeed the case, require that the grid elements be related via $\preceq_H$ accordingly. We do just that with φ_{12}:

$$\bigwedge_{X=H,V} \forall yz\Big(\big(G(y)\wedge G(z)\wedge \forall w(E_X(zw)\to C_X(wzy))\big) \to z \preceq_X y\Big). \tag{φ_{12}}$$

Noting that $\preceq_V$, R_V and C_V behave similarly, we have that $\mathfrak{G}^\# \models \varphi_{11}\wedge\varphi_{12}$.

Lastly, notice that there are no two grid elements that have the same E_H- and E_V-degrees. Thus, $\mathfrak{G}^\#$ models the uniqueness requirement as given by φ_{13}:

$$\forall y\,\Big(G(y)\to \exists_{[=1]}z\,\big(\bigwedge_{X=H,V}(y\preceq_X z\wedge z\preceq_X y)\big)\Big). \tag{φ_{13}}$$

Notice that by stepping inside the realm of periodic counting, we may capture the fact that in $\mathfrak{G}^\#$ there are no transfinite positions by defining the sentence χ limiting the E_H- and E_V-degrees of elements to finite values:

$$\bigwedge_{X=H,V} \forall x\exists_{[0^{+1}]}y\; E_X(xy). \tag{χ}$$

Recalling that $\varphi := \varphi_1\wedge\cdots\wedge\varphi_{13}$ we have showed the following:

▶ **Lemma 13.** *The graphed, mapped and ordered expansion of the canonical* $(\mathbb{N}\times\mathbb{N})$*-grid is a model of* $\varphi\wedge\chi$*.*

We proceed with the other direction as follows:

▶ **Lemma 14.** *Suppose* $\mathfrak{A}\models\varphi$*. Then* $\mathfrak{A}$ *is a grid-like structure. In addition, if* $\mathfrak{A}\models\chi$*, then* $\mathfrak{A}$ *is an* $(\mathbb{N}\times\mathbb{N})$*-grid.*

Proof. Suppose first that $\mathfrak{A}\models\varphi$. Notice that by φ_1 there is exactly one element that satisfies O in $\mathfrak{A}$ and, by φ_2, has no incoming E_H- and E_V-edges. This will be our spare part element in the argument to come. Now, take any element $a_{0,0}\in A$ such that $a_{0,0}\in G^{\mathfrak{A}}$ (i.e. $a_{0,0}$ is a grid element) with finite E_H- and E_V-degree. Such an element is guaranteed to exist by φ_3. Then, φ_4 gives us that $a_{0,0}$ has an H-successor $a_{1,0}$ and a V-successor $a_{0,1}$. Notice that, by φ_5 each E_H-edge originating from $a_{0,0}$ along with the spare part is paired with exactly one edge E_H-edge from $a_{1,0}$ in $S_E^{\mathfrak{A}}$. That is to say, writing $U=\{b\in A\mid a_{0,0}b\in E_H^{\mathfrak{A}} \text{ or } b\in O^{\mathfrak{A}}\}$ and $U'=\{c\in A\mid a_{1,0}c\in E_H^{\mathfrak{A}}\}$, we have that for each $b\in U$ there is exactly one $c\in U'$ such that $ba_{0,0}a_{1,0}c\in S_E^{\mathfrak{A}}$. The reverse is established by φ_6. Clearly, there is a bijection between U and U' thus making the E_H-degree of $a_{1,0}$ one greater than that of $a_{0,0}$. By φ_7 we have that $a_{0,0}\preceq_V^{\mathfrak{A}} a_{1,0}$ and $a_{1,0}\preceq_V^{\mathfrak{A}} a_{0,0}$. We first fixate on the fact that $a_{0,0}\preceq_V^{\mathfrak{A}} a_{1,0}$. Writing $U=\{b\in A\mid a_{0,0}b\in E_V^{\mathfrak{A}}\}$ and $U'=\{c\in A\mid a_{1,0}c\in E_V^{\mathfrak{A}}\}$ we have, by φ_9, that for each

$b \in U$ there is exactly one $c \in U'$ such that $ba_{0,0}a_{1,0}c \in R_V^{\mathfrak{A}}$. By φ_{10}, for each $c \in U'$ there is at most a single $b \in U$ such that $ba_{0,0}a_{1,0}c \in R_V^{\mathfrak{A}}$. We may thus regard $R_V^{\mathfrak{A}}$ as being an injection between E_V-edges of $a_{0,0}$ and that of $a_{1,0}$. Then, again by φ_9, φ_{10} and the fact that $a_{1,0} \preceq_V^{\mathfrak{A}} a_{0,0}$, we have that $R_V^{\mathfrak{A}}$ is an injection between the E_V-edges from $a_{1,0}$ and that of $a_{0,0}$. By the Cantor-Schröder-Bernstein Theorem, their E_V-degrees are thus equal. A symmetric argument holds for the E_V- and E_H-degree of $a_{0,1}$.

Now, let $a_{1,1}$ and $a'_{1,1}$ be, respectively, the V-sucessor of $a_{1,0}$ and the H-sucessor of $a_{0,1}$ promised by φ_4. Using the same arguments as in the paragraph above, it is easy to see that the E_H-degrees of $a_{1,1}$ and $a'_{1,1}$ coincide; and so do the E_V-degrees. We claim that $a_{1,1} = a'_{1,1}$. By φ_{13} we need only show that $a_{1,1}$ is equal to $a'_{1,1}$ with respect to the orderings $\preceq_H^{\mathfrak{A}}$ and $\preceq_V^{\mathfrak{A}}$ as we already have that $a_{1,1} \preceq_E a_{1,1}$ and $a_{1,1} \preceq_V a_{1,1}$ by φ_8. Fixating on E_H-edges first, we have, by φ_8, that $a_{1,1}$ and $a'_{1,1}$ are comparable by $\preceq_E^{\mathfrak{A}}$ in some way. Suppose, without loss of generality, that $a_{1,1} \preceq_E^{\mathfrak{A}} a'_{1,1}$. Writing $U = \{b \in A \mid a_{1,1}b \in E_H^{\mathfrak{A}}\}$ and $U' = \{c \in A \mid a'_{1,1}c \in E_H^{\mathfrak{A}}\}$ we have, by φ_9, that for each $b \in U$ there is exactly one $c \in U'$ such that $ba_{1,1}a'_{1,1}c \in R_E^{\mathfrak{A}}$, and, by φ_{10}, for each $c \in U'$ there is at most one $b \in U$ such that the same holds. That is to say, $R_E^{\mathfrak{A}}$ is an injection between E_H-edges originating from $a_{1,1}$ and E_H-edges from $a'_{1,1}$. Notice that since $a_{1,1}$ and $a'_{1,1}$ both have an equal and finite E_H-degree, we can conclude that $R_E^{\mathfrak{A}}$ is a bijection between the edges. Using this, we have that $ca'_{1,1}a_{1,1} \in C_H^{\mathfrak{A}}$ for each $c \in U'$ by φ_{11}. Clearly, the antecedents of φ_{12} are met and thus $a'_{1,1} \preceq_E^{\mathfrak{A}} a_{1,1}$ as required. Repeating the argument for $\preceq_V^{\mathfrak{A}}$ we indeed have (by φ_{13}) that $a_{1,1} = a'_{1,1}$ thus closing the grid.

By repeating the argument above on element in $G^{\mathfrak{A}}$ with finite E_H- and E_V-degree we conclude that $\mathfrak{A}$ contains a homomorphic embedding of the canonical ($\mathbb{N}\times\mathbb{N}$)-grid thus making it grid-like.

Supposing, in addition, that $\mathfrak{A} \models \chi$ we have that each element in $G^{\mathfrak{A}}$ has a finite E_H- and E_V-degree. We may thus unambiguously identify these elements as the pair of their E_H-degree $i \in \mathbb{N}$ and E_V-degree $j \in \mathbb{N}$. Hence, the structure $\mathfrak{A}$ restricted to elements in $G^{\mathfrak{A}}$ and signature $\{G, H, V\}$ is isomorphic to the canonical ($\mathbb{N}\times\mathbb{N}$)-grid thus making $\mathfrak{A}$ an ($\mathbb{N}\times\mathbb{N}$)-grid as required. ◀

Combining Lemmas 13 and 14 we have that φ is a satisfiable $\mathcal{FLC}^4_{\text{rev}}$-sentence modeled exclusively by (some non-empty subclass of) grid-like structures, whilst $\varphi \wedge \chi$ is a satisfiable $\mathcal{FLPC}^4_{\text{rev}}$-sentence that is modeled only by (some non-empty subclass of) ($\mathbb{N}\times\mathbb{N}$)-grids. Combining the observation above with Lemma 12 we have the following:

▶ **Theorem 15.** *The satisfiability problem for $\mathcal{FLC}^4_{rev}$ is Π^0_1-hard. The same problem for $\mathcal{FLPC}^4_{rev}$ is Σ^1_1-hard.*

Note that the adjacent fragment with periodic counting is a fragment of the constructive fragment of $\mathcal{L}_{\omega_1,\omega}$, which has a Σ^1_1-complete satisfiability problem [10, 15]. We conclude the section by reformulating results of Theorems 11 and 15 in terms of the adjacent fragment:

▶ **Corollary 16.** *The finite and general satisfiability problems for the adjacent fragment with counting are, respectively, Σ^0_1- and Π^0_1-complete. If periodic counting is permitted, then the general satisfiability problem turns to be Σ^1_1-complete.*

6 Discussion

In this paper we utilised the homogeneity property of satisfiable $\mathcal{FLPC}$-sentences to establish a decision procedure for the (finite) satisfiability problem of the new language. With this methodology we not only gained a better understanding of models of fluted formulas, but also managed to establish decidability of (finite) satisfiability using simpler methods when compared to Presburger quantifiers discussed in previous literature [19, 24].

Reflecting on global homogeneity we see that, as opposed to local homogeneity, the rewiring in Lemma 2 is impacted by the presence of the (in)equality atom. More precisely, the semantics of predicates of arity at most ℓ do not interfere in the rewiring for local ℓ-homogeneity. Because of this we may establish a more general result for local homogeneity of fluted sentences with semantic extensions. Consider a signature that is split into symbols σ^* with a fixed interpretation (e.g. transitive relation, reversed relation, etc.) and standard predicate symbols σ with no fixed meaning. Furthermore, suppose that the maximum arity of any symbol in σ^* is at most k. Then, Lemma 6 implies the following:

▶ **Corollary 17.** *Suppose φ is a fluted, normal-form, $(\ell{+}1)$-variable sentence (possibly with periodic counting) over the signature $\sigma \cup \sigma^*$, and where $\ell \geq k$. Then if φ is satisfiable it is satisfiable in a locally ℓ-homogeneous model.*

The same cannot be said about $(\ell{+}1)$-variable sentences when $\ell < k$, and thus establishing an analogue to global homogeneity (as was done for $\mathcal{FLPC}^2$ with equality in Lemma 2) requires case-by-case consideration. Nonetheless, we believe our approach could not only be used to simplify existing decidability procedures for satisfiability (e.g. for $\mathcal{FL}$ with a transitive relation and counting [24]) but to also expand on expressiveness of fluted languages.

We make use of Corollary 17 when (briefly) analysing the language $\mathcal{L}$ formed by combining $\mathcal{FLPC}$ and $\mathcal{FO}^2_{\text{Pres}}$. That is to say, $\mathcal{L}$ is $\mathcal{FLPC}$ with $(\ell{+}1)$-atoms $R(x_{\ell+1}, R_\ell)$ and $R(x_{\ell+1}, x_{\ell+1})$ allowed. The following is almost immediate. (See Appendix C for the proof).

▶ **Theorem 18.** *The (finite) satisfiability problem for $\mathcal{L}$ is decidable.*

Combining the result above with Corollary 16, and noting that more expressive counting quantifiers render the two-variable fragment undecidable [7, 5], one can argue that the language $\mathcal{L}$ is on the edge of decidability (for satisfiability). There are, however, other maximal fragments with counting that have a decidable satisfiability problem. The reader might have noticed that, in Section 5, we did not consider the general satisfiability problem for the 3-variable adjacent fragment with counting (whilst noting that the finite variant is undecidable in Theorem 11). Surprisingly, the problem was recently shown to be in Δ^0_1 by the current author. The details, however, are beyond the scope of this article. We conclude the paper by outlining the following problems which, to the best of our knowledge, are open.

1. What is the complexity of satisfiability for the 3-variable adjacent fragment with counting?
2. Is the satisfiability problem for the adjacent fragment with periodic counting Σ^1_1-hard?
3. Is the satisfiability problem for the guarded adjacent fragment with counting decidable?

References

1 Franz Baader. A new description logic with set constraints and cardinality constraints on role successors. In *Frontiers of Combining Systems*, pages 43–59. Springer International Publishing, 2017. doi:10.1007/978-3-319-66167-4_3.

2 Bartosz Bednarczyk. Exploiting forwardness: Satisfiability and query-entailment in forward guarded fragment. In *Logics in Artificial Intelligence - 17th European Conference, JELIA 2021, Virtual Event, May 17-20, 2021, Proceedings*, volume 12678 of *Lecture Notes in Computer Science*, pages 179–193. Springer, 2021. doi:10.1007/978-3-030-75775-5_13.

3 Bartosz Bednarczyk and Reijo Jaakkola. Towards a model theory of ordered logics: Expressivity and interpolation. In *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 15:1–15:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.MFCS.2022.15.

4 Bartosz Bednarczyk, Daumantas Kojelis, and Ian Pratt-Hartmann. On the Limits of Decision: the Adjacent Fragment of First-Order Logic. In *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 111:1–111:21, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPICS.ICALP.2023.111.

5 Bartosz Bednarczyk, Maja Orłowska, Anna Pacanowska, and Tony Tan. On Classical Decidable Logics Extended with Percentage Quantifiers and Arithmetics. In *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2021)*, volume 213 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:15, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPICS.FSTTCS.2021.36.

6 Michael Benedikt, Egor V. Kostylev, and Tony Tan. Two variable logic with ultimately periodic counting. *SIAM Journal on Computing*, 53(4):884–968, 2024. doi:10.1137/22M1504792.

7 Erich Grädel, Martin Otto, and Eric Rosen. Undecidability results on two-variable logics. *Archive for Mathematical Logic*, 38(4):313–354, 1999. doi:10.1007/S001530050130.

8 Erich Grädel. On the restraining power of guards. *The Journal of Symbolic Logic*, 64(4):1719–1742, 1999. doi:10.2307/2586808.

9 Erich Grädel, Phokion G. Kolaitis, and Moshe Y. Vardi. On the decision problem for two-variable first-order logic. *The Bulletin of Symbolic Logic*, 3(1):53–69, 1997. doi:10.2307/421196.

10 David Harel. Recurring dominoes: Making the highly undecidable highly understandable. In *Topics in the Theory of Computation*, volume 102 of *North-Holland Mathematics Studies*, pages 51–71. North-Holland, 1985.

11 Andreas Herzig. A new decidable fragment of first order logic. In *Abstracts of the 3rd Logical Biennial Summer School and Conference in honour of S. C. Kleene*, Varna, Bulgaria, 1990.

12 Ullrich Hustadt, Renate A Schmidt, and Lilia Georgieva. A survey of decidable first-order fragments and description logics. *Journal of Relational Methods in Computer Science*, 1(3):251–276, 2004.

13 Reijo Jaakkola. Ordered fragments of first-order logic. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, volume 202 of *LIPIcs*, pages 62:1–62:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.MFCS.2021.62.

14 Emil Jeřábek. Division by zero. *Archive for Mathematical Logic*, 55(7):997–1013, 2016. doi:10.1007/S00153-016-0508-5.

15 H. Jerome. Keisler. *Model theory for infinitary logic : logic with countable conjunctions and finite quantifiers*. Studies in logic and the foundations of mathematics ; v. 62. North-Holland Pub. Co., Amsterdam, 1971.

16 Viktor Kuncak and Martin Rinard. Towards efficient satisfiability checking for boolean algebra with presburger arithmetic. In *Automated Deduction – CADE-21*, pages 215–230, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-73595-3_15.

17 Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981. doi:10.1145/322276.322287.

18 Ian Pratt-Hartmann. The two-variable fragment with counting revisited. In *Logic, Language, Information and Computation, 17th International Workshop, WoLLIC 2010, Brasilia, Brazil, July 6-9, 2010. Proceedings*, volume 6188 of *Lecture Notes in Computer Science*, pages 42–54. Springer, 2010. doi:10.1007/978-3-642-13824-9_4.

19 Ian Pratt-Hartmann. Fluted logic with counting. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 141:1–141:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ICALP.2021.141.

20 Ian Pratt-Hartmann. *Fragments of First-Order Logic*. Oxford University Press, 2023.

21 Ian Pratt-Hartmann, Wieslaw Szwast, and Lidia Tendera. The fluted fragment revisited. *Journal of Symbolic Logic*, 84(3):1020–1048, 2019. doi:10.1017/JSL.2019.33.

22 Ian Pratt-Hartmann and Lidia Tendera. The Fluted Fragment with Transitivity. In *44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019)*, volume 138 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPICS.MFCS.2019.18.

23 Ian Pratt-Hartmann and Lidia Tendera. The fluted fragment with transitive relations. *Annals of Pure and Applied Logic*, 173(1):103042, 2022. doi:10.1016/J.APAL.2021.103042.

24 Ian Pratt-Hartmann and Lidia Tendera. Adding Transitivity and Counting to the Fluted Fragment. In *31st EACSL Annual Conference on Computer Science Logic (CSL 2023)*, volume 252 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:22, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPICS.CSL.2023.32.

25 William C. Purdy. Fluted formulas and the limits of decidability. *The Journal of Symbolic Logic*, 61(2):608–620, 1996. doi:10.2307/2275678.

26 Willard Van Orman Quine. On the limits of decision. In *Proceedings of the 14th International Congress of Philosophy*, volume III, pages 57–62. University of Vienna, 1969.

A Preliminaries

▶ **Lemma 1.** *Suppose φ is an $\mathcal{FLPC}^{\ell+1}$-sentence. Then, we may compute, in polynomial time, an equisatisfiable normal-form $\mathcal{FLPC}^{\ell+1}$-sentence ψ.*

Proof. We start by assuming that φ contains no universal quantifiers. No loss of generality follows this supposition as every formula of the form $\forall\theta$ is equivalent to $\exists_{[0]}\neg\theta$. Writing $\varphi_0 := \varphi$, take any subformula $\theta := \exists_{[n+p]}\chi$ of φ_0, where χ is quantifier-free. Supposing there are k free variables in θ, let q be a fresh predicate of arity k. We write ψ_1 as $\forall^\ell(q \to \exists_{[n+p]}\chi) \wedge \forall^\ell(\neg q \to \neg\exists_{[n+p]}\chi)$ and define φ_1 to be φ_0 but with θ replaced by q. Clearly, $\varphi_1 \wedge \psi_1 \models \varphi_0$. Conversely, if $\mathfrak{A} \models \varphi_0$, we may expand $\mathfrak{A}$ to $\mathfrak{A}'$ by setting $\bar{a} \in q^{\mathfrak{A}'}$ if $\mathfrak{A}, \bar{a} \models \theta$ for each $\bar{a} \in A^k$. Then, $\mathfrak{A}' \models \varphi_1 \wedge \psi_1$ as required. Processing φ_1 and subsequent sentences in the same way, we are left with a sentence φ_m composed solely of proposition letters and sentences $\psi_1, \ldots, \psi_m$. The conjunction of the aforementioned sentences is then (after rearrangement) of the required form. ◀

B Counting With Reversed Relations

▷ Claim. Suppose $\mathcal{E}$ is an instance of Hilbert's 10th problem and φ is computed as described as above Theorem 11. Then, φ is finitely satisfiable if and only if $\mathcal{E}$ has a solution over $\mathbb{N}$.

Proof. Suppose $\mathfrak{A} \models \varphi$. We claim that $\{u \mapsto |A_u^{\mathfrak{A}}| \mid u \in \mathrm{vars}(\mathcal{E})\}$ is a satisfying assignment for $\mathcal{E}$. Thus, again taking $e \in \mathcal{E}$ in turn, we have that if e is of the form (i) $u = 1$, then $\mathfrak{A} \models \varphi_e \implies |A_u^{\mathfrak{A}}| = 1$. If e takes the form (ii) $u + v = w$, we then claim that $R_e^{\mathfrak{A}}$ is a bijection between $A_u^{\mathfrak{A}} \cup A_v^{\mathfrak{A}}$ and $A_w^{\mathfrak{A}}$ having cardinality $|A_u^{\mathfrak{A}}| + |A_v^{\mathfrak{A}}|$. Indeed, by the first conjunct of φ_e we have that each element in $A_u^{\mathfrak{A}} \cup A_v^{\mathfrak{A}}$ is paired with a single element in $A_w^{\mathfrak{A}}$; the converse is establishes by the second conjunct. Thus, clearly, $|A_u^{\mathfrak{A}} \cup A_v^{\mathfrak{A}}| = |A_w^{\mathfrak{A}}|$. But

since $\mathfrak{A} \models \psi$ we have that $A^{\mathfrak{A}}_u \cap A^{\mathfrak{A}}_v = \emptyset$. This coupled with our initial assumption that $u \neq v$ gives us $|A^{\mathfrak{A}}_u \cup A^{\mathfrak{A}}_v| = |A^{\mathfrak{A}}_u| + |A^{\mathfrak{A}}_v|$ as required. Lastly, suppose e takes the form (iii) $u \cdot v = w$. By the first conjunct of φ_e for each $ab \in A^{\mathfrak{A}}_u \times A^{\mathfrak{A}}_v$ there is a single $c \in A^{\mathfrak{A}}_w$ such that $abc \in P^{\mathfrak{A}}_e$. Hence, $P^{\mathfrak{A}}_e$ gives rise to a function $f := \{ab \mapsto c \mid abc \in P^{\mathfrak{A}}_e \cap (A^{\mathfrak{A}}_u \times A^{\mathfrak{A}}_v \times A^{\mathfrak{A}}_w)\}$. Thus, writing $f(xy) = z$ in place of $P_e(xyz)$, we claim that f is a bijection between $A^{\mathfrak{A}}_u \times A^{\mathfrak{A}}_v$ and $A^{\mathfrak{A}}_w$ and has cardinality $|A^{\mathfrak{A}}_u| \cdot |A^{\mathfrak{A}}_v|$. It is easily seen that f is surjective from either the second or third conjunct of φ_e. To establish injectivity suppose $f(ab) = f(a'b') = c$. But, by the second conjunct of φ_e, we have that $\mathfrak{A}, c \models \exists_{[=1]} y \left(A_v(y) \wedge \exists x \left(A_u(x) \wedge f(xy) = z \right) \right)$, thus $b = b'$. Notice that this establishes that b is the only element in $A^{\mathfrak{A}}_v$ for which, under the assignment $c \mapsto z$, $b \mapsto y$, $\mathfrak{A}, cb \models \exists x \left(A_u(x) \wedge f(xy) = z \right)$. Combining this fact with the third conjunct of φ_e we have that, again under the assignment $c \mapsto z$, $b \mapsto y$, $\mathfrak{A}, cb \models \exists_{[=1]} x \left(A_u(x) \wedge f(xy) = z \right)$ thus making $a = a'$. We finish our argument by noting that $|A^{\mathfrak{A}}_u| \cdot |A^{\mathfrak{A}}_v| = |A^{\mathfrak{A}}_u \times A^{\mathfrak{A}}_v| = |f| = |A^{\mathfrak{A}}_w|$ as required.

Conversely, suppose $\mathcal{E}$ has a solution. We define $\pi : \mathrm{vars}(\mathcal{E}) \to \mathbb{N}$ to be the satisfying assignment for $\mathcal{E}$ and construct a model $\mathfrak{A}$ of φ as follows. For each variable $u \in \mathrm{vars}(\mathcal{E})$ set $A^{\mathfrak{A}}_u$ be a set of $\pi(u)$ distinct elements and set the domain of $\mathfrak{A}$ to be $A = \bigcup_{u \in \mathrm{vars}(\mathcal{E})} A^{\mathfrak{A}}_u$ where each $A^{\mathfrak{A}}_u \cap A^{\mathfrak{A}}_v = \emptyset$ for $u \neq v$. Clearly, $\mathfrak{A} \models \psi$. If φ_e was constructed from $e \in \mathcal{E}$ of the form (i) $u = 1$, then $A^{\mathfrak{A}}_u = 1$ as required by φ_e. On the other hand, if e is of the form (ii) $u + v = w$, we have that $\mathfrak{A} \models \varphi_e$ by setting $R^{\mathfrak{A}}_e$ to be a bijection between $A^{\mathfrak{A}}_u \cup A^{\mathfrak{A}}_v$ and $A^{\mathfrak{A}}_w$ (this can be done as $\pi(u) + \pi(v) = \pi(w)$ and, by initial assumption, $u \neq v$). Lastly, if e is (iii) $u \cdot v = w$, then $\pi(u) \cdot \pi(v) = \pi(w)$. Thus, index elements of $A^{\mathfrak{A}}_u$ as $a_1 \dots a_{\pi(u)}$, elements of $A^{\mathfrak{A}}_v$ as $b_1 \dots b_{\pi(v)}$ and elements of $A^{\mathfrak{A}}_w$ as $(c_{i,j})^{1 \leq i \leq \pi(u)}_{1 \leq j \leq \pi(v)}$. Clearly, by setting $P^{\mathfrak{A}}_e = \{a_i b_j c_{i,j} \mid 1 \leq i \leq \pi(u), 1 \leq j \leq \pi(v)\}$ we have that $\mathfrak{A} \models \varphi_e$ thus concluding the proof. ◁

C Discussion

▶ **Theorem 18.** *The (finite) satisfiability problem for $\mathcal{L}$ (i.e. the combination of $\mathcal{FLPC}$ and $\mathcal{FO}^2_{Pres}$) is decidable.*

Proof. Let $\mathcal{L}^{\ell+1}$ be the $\ell+1$-variable sub-fragment of $\mathcal{L}$. Taking some sentence $\varphi \in \mathcal{L}^{\ell+1}$ we proceed by induction on the number of variables. If $\ell + 1 = 2$, then φ is a sentence in the two-variable fragment with periodic counting which is known to have a decidable (finite) satisfiability problem [6]. Now, set $\ell + 1 > 2$ and suppose that the (finite) satisfiability problem for $\mathcal{L}^{\ell}$ is decidable. We may assume (by allowing $\alpha_r, \beta_t, \gamma_r, \delta_t$ to contain binary predicates of the form $R(x_{\ell+1}, x_\ell)$ and $R(x_{\ell+1}, x_{\ell+1})$) that φ is in normal-form (5). Defining σ^* to be the set of predicates of arity no more than 2, we have that if φ is satisfiable then, by Corollary 17, it is satisfiable in an ℓ-homogeneous model. Applying the variable reduction outlined in Lemmas 7 and 8 we will then have the required result. ◀

The Complexity of Second-Order HyperLTL

Hadar Frenkel
Bar-Ilan University, Ramat Gan, Israel

Martin Zimmermann
Aalborg University, Denmark

Abstract

We determine the complexity of second-order HyperLTL satisfiability, finite-state satisfiability, and model-checking: All three are equivalent to truth in third-order arithmetic.

We also consider two fragments of second-order HyperLTL that have been introduced with the aim to facilitate effective model-checking by restricting the sets one can quantify over. The first one restricts second-order quantification to smallest/largest sets that satisfy a guard while the second one restricts second-order quantification further to least fixed points of (first-order) HyperLTL definable functions. All three problems for the first fragment are still equivalent to truth in third-order arithmetic while satisfiability for the second fragment is Σ^1_1-complete, i.e., only as hard as for (first-order) HyperLTL and therefore much less complex. Finally, finite-state satisfiability and model-checking are in Σ^2_2 and are Σ^1_1-hard, and thus also less complex than for full second-order HyperLTL.

2012 ACM Subject Classification Theory of computation → Verification by model checking; Theory of computation → Logic and verification

Keywords and phrases HyperLTL, Satisfiability, Model-checking

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.10

Related Version *Full Version*: `https://arxiv.org/abs/2311.15675` [21]

Funding *Martin Zimmermann*: Supported by DIREC – Digital Research Centre Denmark.

Acknowledgements This work was initiated by a discussion at Dagstuhl Seminar 23391 "The Futures of Reactive Synthesis" and some results were obtained at Dagstuhl Seminar 24111 "Logics for Dependence and Independence: Expressivity and Complexity". We are grateful to Gaëtan Regaud for finding and fixing a bug in the proof of Theorem 18 and to the reviewers for their detailed and valuable feedback, which improved the paper considerably.

1 Introduction

The introduction of hyperlogics [11] for the specification and verification of hyperproperties [12] – properties that relate multiple system executions, has been one of the major success stories of formal verification during the last decade. Logics like HyperLTL and HyperCTL* [11], the extensions of LTL [32] and CTL* [14] (respectively) with trace quantification, are natural specification languages for information-flow and security properties, have a decidable model-checking problem [17], and hence found many applications in program verification.

However, while expressive enough to express common information-flow properties, they are unable to express other important hyperproperties, e.g., common knowledge in multi-agent systems and asynchronous hyperproperties (witnessed by a plethora of asynchronous extensions of HyperLTL, e.g., [1, 2, 3, 6, 9, 10, 23, 26, 27, 28]). These examples all have in common that they are *second-order* properties, i.e., they naturally require quantification over *sets* of traces, while HyperLTL (and HyperCTL*) only allows quantification over traces.

In light of this situation, Beutner et al. [4] introduced the logic Hyper^{2}LTL, which extends HyperLTL with second-order quantification, i.e., quantification over sets of traces. They show that the resulting logic, Hyper^{2}LTL, is indeed able to capture common knowledge, asynchronous extensions of HyperLTL, and many other applications.

Consider, e.g., common knowledge in multi-agent systems where each agent i only observes some parts of the system. The agent *knows* that a statement φ holds if it holds on all traces that are *indistinguishable* in the agent's view. We write $\pi \sim_i \pi'$ if the traces π and π' are indistinguishable for agent i. A property φ is common knowledge among all agents if all agents know φ, all agents know that all agents know φ, and so on, i.e., one takes the infinite closure of knowledge among all agents. This infinite closure cannot be expressed using first-order quantification over traces [8], like the one used in HyperLTL. The second-order quantification suggested by Beutner et al. allows us to express common knowledge, as demonstrated by the formula φ_{ck}, which states that φ is common knowledge on all traces of the system (we use a simplified syntax for readability):

$$\varphi_{ck} = \forall \pi.\, \exists X.\, \pi \in X \wedge \left(\forall \pi' \in X. \forall \pi''.\, \left(\bigvee\nolimits_{i=1}^{n} \pi' \sim_i \pi''\right) \rightarrow \pi'' \in X\right) \wedge \forall \pi' \in X.\, \varphi(\pi')$$

The formula φ_{ck} expresses that for every trace t (instantiating π), there exists a set T (an instantiation of the second-order variable X) such that t is in T, T is closed under the observations of all agents (if t' is in T and t'' is indistinguishable from t' for some agent i, then also t'' is in T), and all traces in T satisfy φ.

However, Beutner et al. also note that this expressiveness comes at a steep price: model-checking Hyper^{2}LTL is highly undecidable, i.e., Σ^1_1-hard. Thus, their main result is a partial model-checking algorithm for a fragment of Hyper^{2}LTL where second-order quantification degenerates to least fixed point computations of HyperLTL definable functions. Their algorithm over- and underapproximates these fixed points and then invokes a HyperLTL model-checking algorithm on these approximations. A prototype implementation of the algorithm is able to model-check properties capturing common knowledge, asynchronous hyperproperties, and distributed computing.

However, one question has been left open: Just how complex is Hyper^{2}LTL verification?

Complexity Classes for Undecidable Problems. The complexity of undecidable problems is typically captured in terms of the arithmetical and analytical hierarchy, where decision problems (encoded as subsets of $\mathbb{N}$) are classified based on their definability by formulas of higher-order arithmetic, namely by the type of objects one can quantify over and by the number of alternations of such quantifiers. We refer to Roger's textbook [35] for fully formal definitions and refer to Figure 1 for a visualization.

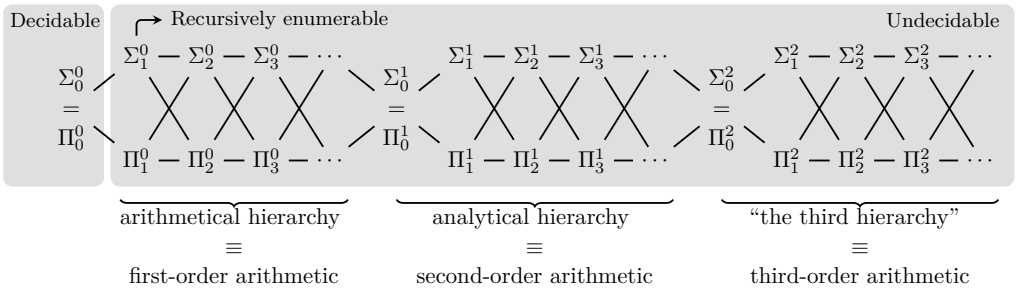


Figure 1 The arithmetical hierarchy, the analytical hierarchy, and beyond.

The class Σ_1^0 contains the sets of natural numbers of the form

$$\{x \in \mathbb{N} \mid \exists x_0.\ \cdots \exists x_k.\ \psi(x, x_0, \ldots, x_k)\}$$

where quantifiers range over natural numbers and ψ is a quantifier-free arithmetic formula. Note that this is exactly the class of recursively enumerable sets. The notation Σ_1^0 signifies that there is a single block of existential quantifiers (the subscript 1) ranging over natural numbers (type 0 objects, explaining the superscript 0). Analogously, Σ_1^1 is induced by arithmetic formulas with existential quantification of type 1 objects (sets of natural numbers) and arbitrary (universal and existential) quantification of type 0 objects. So, Σ_1^0 is part of the first level of the arithmetical hierarchy while Σ_1^1 is part of the first level of the analytical hierarchy. In general, level Σ_n^0 (level Π_n^0) of the arithmetical hierarchy is induced by formulas with at most $n-1$ alternations between existential and universal type 0 quantifiers, starting with an existential (universal) quantifier. Similar hierarchies can be defined for arithmetic of any fixed order by limiting the alternations of the highest-order quantifiers and allowing arbitrary lower-order quantification. In this work, the highest order we are concerned with is three, i.e., quantification over sets of sets of natural numbers.

HyperLTL satisfiability is Σ_1^1-complete [19], HyperLTL finite-state satisfiability is Σ_1^0-complete [16, 20], and, as mentioned above, Hyper^{2}LTL model-checking is Σ_1^1-hard [4], but, prior to this current work, no upper bounds were known for Hyper^{2}LTL.

Another yardstick is truth for order k arithmetic, i.e., the question whether a given sentence of order k arithmetic evaluates to true. In the following, we are in particular interested in the case $k = 3$, i.e., we consider formulas with arbitrary quantification over type 0 objects, type 1 objects, and type 2 objects (sets of sets of natural numbers). Note that these formulas span the whole third hierarchy, as we allow arbitrary nesting of existential and universal third-order quantification.

Our Contributions. In this work, we determine the exact complexity of Hyper^{2}LTL satisfiability, finite-state satisfiability, and model-checking, for the full logic and the two fragments introduced by Beutner et al. [4], as well as for two variations of the semantics.

An important stepping stone for us is the investigation of the cardinality of models of Hyper^{2}LTL. It is known that every satisfiable HyperLTL sentence has a countable model, and that some have no finite models [18]. This restricts the order of arithmetic that can be simulated in HyperLTL and explains in particular the Σ_1^1-completeness of HyperLTL satisfiability [19]. We show that (unsurprisingly) second-order quantification allows to write formulas that only have uncountable models by generalizing the lower bound construction of HyperLTL to Hyper^{2}LTL. Note that the cardinality of the continuum is a trivial upper bound on the size of models, as they are sets of traces.

With this tool at hand, we are able to show that Hyper^{2}LTL satisfiability is equivalent to truth in third-order arithmetic, i.e., much harder than HyperLTL satisfiability. This increase in complexity is not surprising, as second-order quantification can be expected to increase the complexity considerably. But what might be surprising at first glance is that the problem is not Σ_1^2-complete, i.e., at the same position of the third hierarchy that HyperLTL satisfiability occupies in one full hierarchy below (see Figure 1). However, arbitrary second-order trace quantification corresponds to arbitrary quantification over type 2 objects, which allows to capture the full third hierarchy. Furthermore, we also show that Hyper^{2}LTL finite-state satisfiability is equivalent to truth in third-order arithmetic, and therefore as hard as general satisfiability. This should be contrasted with the situation for HyperLTL described above, where finite-state satisfiability is Σ_1^0-complete (i.e., recursively enumerable) and thus much simpler than general satisfiability, which is Σ_1^1-complete.

Finally, our techniques for Hyper^{2}LTL satisfiability also shed light on the exact complexity of Hyper^{2}LTL model-checking, which we show to be equivalent to truth in third-order arithmetic as well, i.e., all three problems we consider have the same complexity. In particular, this increases the lower bound on Hyper^{2}LTL model-checking from Σ^1_1 to truth in third-order arithmetic. Again, this has be contrasted with the situation for HyperLTL, where model-checking is decidable, albeit TOWER-complete [33, 31].

So, quantification over arbitrary sets of traces makes verification very hard. However, Beutner et al. [4] noticed that many of the applications of Hyper^{2}LTL described above do not require full second-order quantification, but can be expressed with restricted forms of second-order quantification. To capture this, they first restrict second-order quantification to smallest/largest sets satisfying a guard (obtaining the fragment Hyper^{2}LTL$_{\text{mm}}$)[1] and then further restrict those to least fixed points induced by HyperLTL definable operators (obtaining the fragment lfp-Hyper^{2}LTL$_{\text{mm}}$). By construction, these least fixed points are unique, i.e., second-order quantification degenerates to least fixed point computation.

As an example, consider again φ_{ck} above. The internal constraint

$$\forall \pi' \in X. \forall \pi''. \left(\bigvee\nolimits_{i=1}^{n} \pi' \sim_i \pi'' \right) \rightarrow \pi'' \in X$$

defines a condition on what traces have to be in the set X, and how they are added gradually to X, a behavior that can be captured by a fixed point computation for the (monotone) operator induced by the formula above. Since the last part $\forall \pi' \in X.\ \varphi(\pi')$ of φ_{ck} universally quantifies over all traces in X, and since X is existentially quantified, it is enough to consider the minimal set that satisfies the internal constraint: if *some* set satisfies a universal condition, then so does the minimal set. This minimal set is exactly the least fixed point of the operator induced by the formula above. Similar behavior is exhibited by many other applications of the logic, which gives the motivation to explore the fragment lfp-Hyper^{2}LTL$_{\text{mm}}$.

Nevertheless, we show that Hyper^{2}LTL$_{\text{mm}}$ retains the same complexity as Hyper^{2}LTL, i.e., all three problems are still equivalent to truth in third-order arithmetic: Just restricting to guarded second-order quantification does not decrease the complexity.

For all results mentioned so far, it is irrelevant whether we allow second-order quantifiers to range over sets of traces that may contain traces that are not in the model (standard semantics) or whether we restrict these quantifiers to subsets of the model (closed-world semantics). But if we consider lfp-Hyper^{2}LTL$_{\text{mm}}$ satisfiability under closed-world semantics, the complexity finally decreases to Σ^1_1-completeness. Stated differently, one can add least fixed points of HyperLTL definable operators to HyperLTL without increasing the complexity of the satisfiability problem. Finally, for lfp-Hyper^{2}LTL$_{\text{mm}}$ finite-state satisfiability and model-checking, we prove Σ^2_2-membership and Σ^1_1 lower bounds for both semantics, thereby confining the complexity to the second level of the third hierarchy.

Table 1 lists our results and compares them to LTL and HyperLTL. Recall that Beutner et al. showed that lfp-Hyper^{2}LTL$_{\text{mm}}$ yields (partial) model checking and monitoring algorithms [4, 5]. Our results confirm the usability of the lfp-Hyper^{2}LTL$_{\text{mm}}$ fragment also from a theoretical point of view, as all problems relevant for verification have significantly lower complexity (albeit, still highly undecidable).

Proofs omitted due to space restrictions can be found in the full version [21].

[1] In [4] this fragment is termed Hyper^{2}LTL$_{\text{fp}}$. For clarity, since it is not fixed point based, but uses minimality/maximality constraints, we use the subscript "mm" instead of "fp".

Table 1 List of our results (in bold) and comparison to related logics. "T3A-equivalent" stands for "equivalent to truth in third-order arithmetic". Entries marked with an asterisk only hold for closed-world semantics, all others hold for both semantics.

Logic	Satisfiability	Finite-state satisfiability	Model-checking
LTL	PSPACE-complete	PSPACE-complete	PSPACE-complete
HyperLTL	Σ^1_1-complete	Σ^0_1-complete	TOWER-complete
Hyper^{2}LTL	**T3A-equivalent**	**T3A-equivalent**	**T3A-equivalent**
Hyper^{2}LTL$_{\mathbf{mm}}$	**T3A-equivalent**	**T3A-equivalent**	**T3A-equivalent**
lfp-Hyper^{2}LTL$_{\mathbf{mm}}$	**Σ^1_1-complete***	**Σ^1_1-hard/in Σ^2_2**	**Σ^1_1-hard/in Σ^2_2**

2 Preliminaries

We denote the nonnegative integers by $\mathbb{N}$. An alphabet is a nonempty finite set. The set of infinite words over an alphabet Σ is denoted by Σ^ω. Let AP be a nonempty finite set of atomic propositions. A trace over AP is an infinite word over the alphabet 2^{AP}. Given a subset $\mathrm{AP}' \subseteq \mathrm{AP}$, the AP'-projection of a trace $t(0)t(1)t(2)\cdots$ over AP is the trace $(t(0) \cap \mathrm{AP}')(t(1) \cap \mathrm{AP}')(t(2) \cap \mathrm{AP}')\cdots$ over AP'.

A transition system $\mathfrak{T} = (V, E, I, \lambda)$ consists of a finite nonempty set V of vertices, a set $E \subseteq V \times V$ of (directed) edges, a set $I \subseteq V$ of initial vertices, and a labeling $\lambda\colon V \to 2^{\mathrm{AP}}$ of the vertices by sets of atomic propositions. We assume that every vertex has at least one outgoing edge. A path ρ through $\mathfrak{T}$ is an infinite sequence $\rho(0)\rho(1)\rho(2)\cdots$ of vertices with $\rho(0) \in I$ and $(\rho(n), \rho(n+1)) \in E$ for every $n \geq 0$. The trace of ρ is defined as $\lambda(\rho) = \lambda(\rho(0))\lambda(\rho(1))\lambda(\rho(2))\cdots$. The set of traces of $\mathfrak{T}$ is $\mathrm{Tr}(\mathfrak{T}) = \{\lambda(\rho) \mid \rho \text{ is a path through } \mathfrak{T}\}$.

Hyper^{2}LTL. Let $\mathcal{V}_1$ be a set of first-order trace variables (i.e., ranging over traces) and $\mathcal{V}_2$ be a set of second-order trace variables (i.e., ranging over sets of traces) such that $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$. We typically use π (possibly with decorations) to denote first-order variables and X, Y, Z (possibly with decorations) to denote second-order variables. Also, we assume the existence of two distinguished second-order variables $X_a, X_d \in \mathcal{V}_2$ such that X_a refers to the set $(2^{\mathrm{AP}})^\omega$ of all traces, and X_d refers to the universe of discourse (the set of traces the formula is evaluated over).

The formulas of Hyper^{2}LTL are given by the grammar

$$\varphi ::= \exists X.\ \varphi \mid \forall X.\ \varphi \mid \exists \pi \in X.\ \varphi \mid \forall \pi \in X.\ \varphi \mid \psi \qquad \psi ::= \mathtt{p}_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi$$

where $\mathtt{p}$ ranges over AP, π ranges over $\mathcal{V}_1$, X ranges over $\mathcal{V}_2$, and $\mathbf{X}$ (next) and $\mathbf{U}$ (until) are temporal operators. Conjunction ($\wedge$), exclusive disjunction ($\oplus$), implication ($\rightarrow$), and equivalence ($\leftrightarrow$) are defined as usual, and the temporal operators eventually ($\mathbf{F}$) and always ($\mathbf{G}$) are derived as $\mathbf{F}\,\psi = \neg\psi\,\mathbf{U}\,\psi$ and $\mathbf{G}\,\psi = \neg\,\mathbf{F}\,\neg\psi$. We measure the size of a formula by its number of distinct subformulas.

The semantics of Hyper^{2}LTL is defined with respect to a variable assignment, i.e., a partial mapping $\Pi\colon \mathcal{V}_1 \cup \mathcal{V}_2 \to (2^{\mathrm{AP}})^\omega \cup 2^{(2^{\mathrm{AP}})^\omega}$ such that

- if $\Pi(\pi)$ for $\pi \in \mathcal{V}_1$ is defined, then $\Pi(\pi) \in (2^{\mathrm{AP}})^\omega$ and
- if $\Pi(X)$ for $X \in \mathcal{V}_2$ is defined, then $\Pi(X) \in 2^{(2^{\mathrm{AP}})^\omega}$.

Given a variable assignment Π, a variable $\pi \in \mathcal{V}_1$, and a trace t, we denote by $\Pi[\pi \mapsto t]$ the assignment that coincides with Π on all variables but π, which is mapped to t. Similarly, for a variable $X \in \mathcal{V}_2$, and a set T of traces, $\Pi[X \mapsto T]$ is the assignment that coincides with Π everywhere but X, which is mapped to T. Furthermore, $\Pi[j, \infty)$ denotes the variable

assignment mapping every $\pi \in \mathcal{V}_1$ in Π's domain to $\Pi(\pi)(j)\Pi(\pi)(j+1)\Pi(\pi)(j+2)\cdots$, the suffix of $\Pi(\pi)$ starting at position j (the assignment of variables $X \in \mathcal{V}_2$ is not updated).

For a variable assignment Π we define

- $\Pi \models \mathtt{p}_\pi$ if $\mathtt{p} \in \Pi(\pi)(0)$,
- $\Pi \models \neg\psi$ if $\Pi \not\models \psi$,
- $\Pi \models \psi_1 \vee \psi_2$ if $\Pi \models \psi_1$ or $\Pi \models \psi_2$,
- $\Pi \models \mathbf{X}\,\psi$ if $\Pi[1,\infty) \models \psi$,
- $\Pi \models \psi_1 \,\mathbf{U}\, \psi_2$ if there is a $j \geq 0$ such that $\Pi[j,\infty) \models \psi_2$ and for all $0 \leq j' < j$ we have $\Pi[j',\infty) \models \psi_1$,
- $\Pi \models \exists \pi \in X.\ \varphi$ if there exists a trace $t \in \Pi(X)$ such that $\Pi[\pi \mapsto t] \models \varphi$,
- $\Pi \models \forall \pi \in X.\ \varphi$ if for all traces $t \in \Pi(X)$ we have $\Pi[\pi \mapsto t] \models \varphi$,
- $\Pi \models \exists X.\ \varphi$ if there exists a set $T \subseteq (2^{\mathrm{AP}})^\omega$ such that $\Pi[X \mapsto T] \models \varphi$, and
- $\Pi \models \forall X.\ \varphi$ if for all sets $T \subseteq (2^{\mathrm{AP}})^\omega$ we have $\Pi[X \mapsto T] \models \varphi$.

Throughout the paper, we use the following shorthands to simplify our formulas:

- We write $\pi =_{\mathrm{AP}'} \pi'$ for a set $\mathrm{AP}' \subseteq \mathrm{AP}$ for the formula $\mathbf{G} \bigwedge_{\mathtt{p} \in \mathrm{AP}'} (\mathtt{p}_\pi \leftrightarrow \mathtt{p}_{\pi'})$ expressing that the AP'-projection of π and the AP'-projection of π' are equal.
- We write $\pi \triangleright X$ for the formula $\exists \pi' \in X.\ \pi =_{\mathrm{AP}} \pi'$ expressing that the trace π is in X. Note that this shorthand cannot be used under the scope of temporal operators, as we require formulas to be in prenex normal form.

A sentence is a formula in which only the variables X_a, X_d can be free. The variable assignment with empty domain is denoted by $\Pi_\emptyset$. We say that a set T of traces satisfies a Hyper^{2}LTL sentence φ, written $T \models \varphi$, if $\Pi_\emptyset[X_a \mapsto (2^{\mathrm{AP}})^\omega, X_d \mapsto T] \models \varphi$, i.e., if we assign the set of all traces to X_a and the set T to the universe of discourse X_d. In this case, we say that T is a model of φ. A transition system $\mathfrak{T}$ satisfies φ, written $\mathfrak{T} \models \varphi$, if $\mathrm{Tr}(\mathfrak{T}) \models \varphi$.

Although Hyper^{2}LTL sentences are required to be in prenex normal form, Hyper^{2}LTL sentences are closed under Boolean combinations, which can easily be seen by transforming such a sentence into an equivalent one in prenex normal form (which might require renaming of variables). Thus, in examples and proofs we will often use Boolean combinations of Hyper^{2}LTL sentences.

▶ Remark 1. HyperLTL is the fragment of Hyper^{2}LTL obtained by disallowing second-order quantification and only allowing first-order quantification of the form $\exists \pi \in X_d$ and $\forall \pi \in X_d$, i.e., one can only quantify over traces from the universe of discourse. Hence, we typically simplify our notation to $\exists \pi$ and $\forall \pi$ in HyperLTL formulas.

Closed-World Semantics. Second-order quantification in Hyper^{2}LTL as defined by Beutner et al. [4] (and introduced above) ranges over arbitrary sets of traces (not necessarily from the universe of discourse) and first-order quantification ranges over elements in such sets, i.e., (possibly) again over arbitrary traces. To disallow this, we introduce *closed-world* semantics for Hyper^{2}LTL, only considering formulas that do not use the variable X_a. We change the semantics of set quantifiers as follows, where the closed-world semantics of atomic propositions, Boolean connectives, temporal operators, and trace quantifiers is defined as before:

- $\Pi \models_{\mathrm{cw}} \exists X.\ \varphi$ if there exists a set $T \subseteq \Pi(X_d)$ such that $\Pi[X \mapsto T] \models \varphi$, and
- $\Pi \models_{\mathrm{cw}} \forall X.\ \varphi$ if for all sets $T \subseteq \Pi(X_d)$ we have $\Pi[X \mapsto T] \models \varphi$.

We say that $T \subseteq (2^{\mathrm{AP}})^\omega$ satisfies φ under closed-world semantics, if $\Pi_\emptyset[X_d \mapsto T] \models_{\mathrm{cw}} \varphi$. Hence, under closed-world semantics, second-order quantifiers only range over subsets of the

universe of discourse. Consequently, first-order quantifiers also range over traces from the universe of discourse.

▶ **Lemma 2.** *Every* Hyper^{2}LTL *sentence* φ *can be translated in polynomial time (in* $|\varphi|$*) into a* Hyper^{2}LTL *sentence* φ' *such that for all sets* T *of traces we have that* $T \models_{\text{cw}} \varphi$ *if and only if* $T \models \varphi'$ *(under standard semantics).*

Thus, all complexity upper bounds we derive for standard semantics also hold for closed-world semantics and all lower bounds for closed-world semantics hold for standard semantics.

▶ Remark 3. Let φ be an X_a-free Hyper^{2}LTL sentence over AP. We have $(2^{\text{AP}})^\omega \models \varphi$ (under standard semantics) if and only if $(2^{\text{AP}})^\omega \models_{\text{cw}} \varphi$, as the second-order quantifiers range in both cases over subsets of $(2^{\text{AP}})^\omega$, which implies that the trace quantifiers in both cases range over traces from $(2^{\text{AP}})^\omega$.

Arithmetic. To capture the complexity of undecidable problems, we consider formulas of arithmetic, i.e., predicate logic with signature $(+, \cdot, <, \in)$, evaluated over the structure $(\mathbb{N}, +, \cdot, <, \in)$. A type 0 object is a natural number in $\mathbb{N}$, a type 1 object is a subset of $\mathbb{N}$, and a type 2 object is a set of subsets of $\mathbb{N}$.

Our benchmark is third-order arithmetic, i.e., predicate logic with quantification over type 0, type 1, and type 2 objects. In the following, we use lower-case roman letters (possibly with decorations) for first-order variables, upper-case roman letters (possibly with decorations) for second-order variables, and upper-case calligraphic roman letters (possibly with decorations) for third-order variables. Note that every fixed natural number is definable in first-order arithmetic, so we freely use them as syntactic sugar. Truth of third-order arithmetic is the following problem: given a sentence φ of third-order arithmetic, does $(\mathbb{N}, +, \cdot, <, \in)$ satisfy φ?

Arithmetic formulas with a single free first-order variable define sets of natural numbers. We are interested in the classes

- Σ^1_1 containing sets of the form $\{x \in \mathbb{N} \mid \exists X_1 \subseteq \mathbb{N}.\ \cdots \exists X_k \subseteq \mathbb{N}.\ \psi(x, X_1, \ldots, X_k)\}$, where ψ is a formula of arithmetic with arbitrary quantification over type 0 objects (but no second-order quantifiers), and
- Σ^2_2 containing sets of the following form, where ψ is a formula of arithmetic with arbitrary quantification over type 0 and type 1 objects (but no third-order quantifiers): $\{x \in \mathbb{N} \mid \exists \mathcal{X}_1 \subseteq 2^{\mathbb{N}}.\ \cdots \exists \mathcal{X}_k \subseteq 2^{\mathbb{N}}. \forall \mathcal{Y}_1 \subseteq 2^{\mathbb{N}}.\ \cdots \forall \mathcal{Y}_{k'} \subseteq 2^{\mathbb{N}}.\ \psi(x, \mathcal{X}_1, \ldots, \mathcal{X}_k, \mathcal{Y}_1, \ldots, \mathcal{Y}_{k'})\}$.

3 The Cardinality of Hyper^{2}LTL Models

In this section, we investigate the cardinality of models of satisfiable Hyper^{2}LTL sentences, i.e., the number of traces in the model.

We begin by stating a (trivial) upper bound, which follows from the fact that models are sets of traces. Here, $\mathfrak{c}$ denotes the cardinality of the continuum (equivalently, the cardinality of $2^{\mathbb{N}}$ and of $(2^{\text{AP}})^\omega$ for any finite nonempty AP).

▶ **Proposition 4.** *Every satisfiable* Hyper^{2}LTL *sentence has a model of cardinality at most* $\mathfrak{c}$.

In this section, we show that this trivial upper bound is tight.

▶ Remark 5. There is a very simple, albeit equally unsatisfactory way to obtain the desired lower bound: Consider $\forall \pi \in X_a.\ \pi \triangleright X_d$ expressing that every trace in the set of all traces is also in the universe of discourse, i.e., $(2^{\text{AP}})^\omega$ is its only model over AP. However, this crucially relies on the fact that X_a is, by definition, interpreted as the set of all traces. In fact, the formula does not even use second-order quantification.

We show how to construct a sentence that has only uncountable models, and which retains that property under closed-world semantics (which in particular means it cannot use X_a). This should be compared with HyperLTL, where every satisfiable sentence has a countable model [18]: Unsurprisingly, the addition of (even closed-world) second-order quantification increases the cardinality of minimal models, even without cheating.

▶ **Example 6.** We begin by recalling a construction of Finkbeiner and Zimmermann giving a satisfiable HyperLTL sentence ψ that has no finite models [18]. The sentence intuitively posits the existence of a unique trace for every natural number n. Our lower bound for Hyper^{2}LTL builds upon that construction.

Fix AP $= \{\mathtt{x}\}$ and consider the conjunction $\psi = \psi_1 \wedge \psi_2 \wedge \psi_3$ of the following three formulas:

1. $\psi_1 = \forall \pi.\ \neg \mathtt{x}_\pi\, \mathbf{U}(\mathtt{x}_\pi \wedge \mathbf{X}\,\mathbf{G}\,\neg \mathtt{x}_\pi)$: every trace in a model is of the form $\emptyset^n\{\mathtt{x}\}\emptyset^\omega$ for some $n \in \mathbb{N}$, i.e., every model is a subset of $\{\emptyset^n\{\mathtt{x}\}\emptyset^\omega \mid n \in \mathbb{N}\}$.
2. $\psi_2 = \exists \pi.\ \mathtt{x}_\pi$: the trace $\emptyset^0\{\mathtt{x}\}\emptyset^\omega$ is in every model.
3. $\psi_3 = \forall \pi.\ \exists \pi'.\ \mathbf{F}(\mathtt{x}_\pi \wedge \mathbf{X}\,\mathtt{x}_{\pi'})$: if $\emptyset^n\{\mathtt{x}\}\emptyset^\omega$ is in a model for some $n \in \mathbb{N}$, then also $\emptyset^{n+1}\{\mathtt{x}\}\emptyset^\omega$.

Then, ψ has exactly one model (over AP), namely $\{\emptyset^n\{\mathtt{x}\}\emptyset^\omega \mid n \in \mathbb{N}\}$.

A trace of the form $\emptyset^n\{\mathtt{x}\}\emptyset^\omega$ encodes the natural number n and ψ expresses that every model contains the encodings of all natural numbers and nothing else. But we can of course also encode sets of natural numbers with traces as follows: a trace t over a set of atomic propositions containing $\mathtt{x}$ encodes the set $\{n \in \mathbb{N} \mid \mathtt{x} \in t(n)\}$. In the following, we show that second-order quantification allows us to express the existence of the encodings of all subsets of natural numbers by requiring that for every subset $S \subseteq \mathbb{N}$ (quantified as the set $\{\emptyset^n\{\mathtt{x}\}\emptyset^\omega \mid n \in S\}$ of traces) there is a trace t encoding S, which means $\mathtt{x}$ is in $t(n)$ if and only if S contains a trace in which $\mathtt{x}$ holds at position n. This equivalence can be expressed in Hyper^{2}LTL. For technical reasons, we do not capture the equivalence directly but instead use encodings of both the natural numbers that are in S and the natural numbers that are not in S.

▶ **Theorem 7.** *There is a satisfiable X_a-free* Hyper^{2}LTL *sentence that only has models of cardinality* $\mathfrak{c}$ *(both under standard and closed-world semantics).*

Proof. We first prove that there is a satisfiable X_a-free Hyper^{2}LTL sentence $\varphi_{allSets}$ whose unique model (under standard semantics) has cardinality $\mathfrak{c}$. To this end, we fix AP $= \{\mathtt{+}, \mathtt{-}, \mathtt{s}, \mathtt{x}\}$ and consider the conjunction $\varphi_{allSets} = \varphi_0 \wedge \cdots \wedge \varphi_4$ of the following formulas:

- $\varphi_0 = \forall \pi \in X_d.\ \bigvee_{\mathtt{p} \in \{\mathtt{+},\mathtt{-},\mathtt{s}\}} \mathbf{G}(\mathtt{p}_\pi \wedge \bigwedge_{\mathtt{p}' \in \{\mathtt{+},\mathtt{-},\mathtt{s}\} \setminus \{\mathtt{p}\}} \neg \mathtt{p}'_\pi)$: In each trace of a model, one of the propositions in $\{\mathtt{+}, \mathtt{-}, \mathtt{s}\}$ holds at every position and the other two propositions in $\{\mathtt{+}, \mathtt{-}, \mathtt{s}\}$ hold at none of the positions. Consequently, we speak in the following about type $\mathtt{p}$ traces for $\mathtt{p} \in \{\mathtt{+}, \mathtt{-}, \mathtt{s}\}$.
- $\varphi_1 = \forall \pi \in X_d.\ (\mathtt{+}_\pi \vee \mathtt{-}_\pi) \rightarrow \neg \mathtt{x}_\pi\, \mathbf{U}(\mathtt{x}_\pi \wedge \mathbf{X}\,\mathbf{G}\,\neg \mathtt{x}_\pi)$: Type $\mathtt{p}$ traces for $\mathtt{p} \in \{\mathtt{+}, \mathtt{-}\}$ in the model have the form $\{\mathtt{p}\}^n\{\mathtt{x}, \mathtt{p}\}\{\mathtt{p}\}^\omega$ for some $n \in \mathbb{N}$.
- $\varphi_2 = \bigwedge_{\mathtt{p} \in \{\mathtt{+},\mathtt{-}\}} \exists \pi \in X_d.\ \mathtt{p}_\pi \wedge \mathtt{x}_\pi$: for both $\mathtt{p} \in \{\mathtt{+}, \mathtt{-}\}$, the type $\mathtt{p}$ trace $\{\mathtt{p}\}^0\{\mathtt{x}, \mathtt{p}\}\{\mathtt{p}\}^\omega$ is in every model.
- $\varphi_3 = \bigwedge_{\mathtt{p} \in \{\mathtt{+},\mathtt{-}\}} \forall \pi \in X_d.\ \exists \pi' \in X_d.\ \mathtt{p}_\pi \rightarrow (\mathtt{p}_{\pi'} \wedge \mathbf{F}(\mathtt{x}_\pi \wedge \mathbf{X}\,\mathtt{x}_{\pi'}))$: for both $\mathtt{p} \in \{\mathtt{+}, \mathtt{-}\}$, if the type $\mathtt{p}$ trace $\{\mathtt{p}\}^n\{\mathtt{x}, \mathtt{p}\}\{\mathtt{p}\}^\omega$ is in a model for some $n \in \mathbb{N}$, then also $\{\mathtt{p}\}^{n+1}\{\mathtt{x}, \mathtt{p}\}\{\mathtt{p}\}^\omega$.

The formulas $\varphi_1, \varphi_2, \varphi_3$ are similar to the formulas ψ_1, ψ_2, ψ_3 from Example 6. So, every model of $\varphi_0 \wedge \cdots \wedge \varphi_3$ contains $\{\{\mathtt{+}\}^n\{\mathtt{x}, \mathtt{+}\}\{\mathtt{+}\}^\omega \mid n \in \mathbb{N}\}$ and $\{\{\mathtt{-}\}^n\{\mathtt{x}, \mathtt{-}\}\{\mathtt{-}\}^\omega \mid n \in \mathbb{N}\}$ as subsets, and no other type $\mathtt{+}$ or type $\mathtt{-}$ traces.

Now, consider a set T of traces over AP (recall that second-order quantification ranges over arbitrary sets, not only over subsets of the universe of discourse). We say that T is contradiction-free if there is no $n \in \mathbb{N}$ such that $\{\mathtt{+}\}^n\{\mathtt{x},\mathtt{+}\}\{\mathtt{+}\}^\omega \in T$ and $\{\mathtt{-}\}^n\{\mathtt{x},\mathtt{-}\}\{\mathtt{-}\}^\omega \in T$. Furthermore, a trace t over AP is consistent with a contradiction-free T if

(C1) $\{\mathtt{+}\}^n\{\mathtt{x},\mathtt{+}\}\{\mathtt{+}\}^\omega \in T$ implies $\mathtt{x} \in t(n)$ and

(C2) $\{\mathtt{-}\}^n\{\mathtt{x},\mathtt{-}\}\{\mathtt{-}\}^\omega \in T$ implies $\mathtt{x} \notin t(n)$.

Note that T does not necessarily specify the truth value of $\mathtt{x}$ in every position of t, i.e., in those positions $n \in \mathbb{N}$ where neither $\{\mathtt{+}\}^n\{\mathtt{x},\mathtt{+}\}\{\mathtt{+}\}^\omega$ nor $\{\mathtt{-}\}^n\{\mathtt{x},\mathtt{-}\}\{\mathtt{-}\}^\omega$ are in T. Nevertheless, for every trace t over $\{\mathtt{x}\}$ there is a contradiction-free T such that the $\{\mathtt{x}\}$-projection of every trace t' over AP that is consistent with T is equal to t. Thus, each of the uncountably many traces over $\{\mathtt{x}\}$ is induced by some subset of the model.

Hence, we define φ_4 as the formula

$$\forall X.\ \overbrace{[\forall \pi \in X.\ \forall \pi' \in X.\ (\mathtt{+}_\pi \wedge \mathtt{-}_{\pi'}) \rightarrow \neg \mathbf{F}(\mathtt{x}_\pi \wedge \mathtt{x}_{\pi'})]}^{X \text{ is contradiction-free}} \rightarrow$$
$$\exists \pi'' \in X_d.\ \forall \pi''' \in X.\ \mathtt{s}_{\pi''} \wedge \underbrace{(\mathtt{+}_{\pi'''} \rightarrow \mathbf{F}(\mathtt{x}_{\pi'''} \wedge \mathtt{x}_{\pi''}))}_{\text{(C1)}} \wedge \underbrace{(\mathtt{-}_{\pi'''} \rightarrow \mathbf{F}(\mathtt{x}_{\pi'''} \wedge \neg\mathtt{x}_{\pi''}))}_{\text{(C2)}},$$

expressing that for every contradiction-free set of traces T, there is a type $\mathtt{s}$ trace t'' in the model (note that π'' is required to be in X_d) that is consistent with T.

While $\varphi_{allSets}$ is not in prenex normal form, it can easily be turned into an equivalent formula in prenex normal form (at the cost of readability).

Now, the set

$$T_{allSets} = \{\{\mathtt{+}\}^n\{\mathtt{x},\mathtt{+}\}\{\mathtt{+}\}^\omega \mid n \in \mathbb{N}\} \cup \{\{\mathtt{-}\}^n\{\mathtt{x},\mathtt{-}\}\{\mathtt{-}\}^\omega \mid n \in \mathbb{N}\} \cup$$
$$\{(t(0) \cup \{\mathtt{s}\})(t(1) \cup \{\mathtt{s}\})(t(2) \cup \{\mathtt{s}\}) \cdots \mid t \in (2^{\{\mathtt{x}\}})^\omega\}$$

of traces satisfies $\varphi_{allSets}$. On the other hand, every model of $\varphi_{allSets}$ must indeed contain $T_{allSets}$ as a subset, as $\varphi_{allSets}$ requires the existence of all of its traces in the model. Finally, due to φ_0 and φ_1, a model (over AP) cannot contain any traces that are not in $T_{allSets}$, i.e., $T_{allSets}$ is the unique model of $\varphi_{allSets}$.

To conclude, we just remark that

$$\{(t(0) \cup \{\mathtt{s}\})(t(1) \cup \{\mathtt{s}\})(t(2) \cup \{\mathtt{s}\}) \cdots \mid t \in (2^{\{\mathtt{x}\}})^\omega\} \subseteq T_{allSets}$$

has indeed cardinality $\mathfrak{c}$, as $(2^{\{\mathtt{x}\}})^\omega$ has cardinality $\mathfrak{c}$.

Finally, let us consider closed-world semantics. We can restrict the second-order quantifier in φ_4 (the only one in $\varphi_{allSets}$) to subsets of the universe of discourse, as the set $T = \{\{\mathtt{+}\}^n\{\mathtt{x},\mathtt{+}\}\{\mathtt{+}\}^\omega \mid n \in \mathbb{N}\} \cup \{\{\mathtt{-}\}^n\{\mathtt{x},\mathtt{-}\}\{\mathtt{-}\}^\omega \mid n \in \mathbb{N}\}$ of traces (which is a subset of every model) is already *rich* enough to encode every subset of $\mathbb{N}$ by an appropriate contradiction-free subset of T. Thus, $\varphi_{allSets}$ has the unique model $T_{allSets}$ even under closed-world semantics. ◀

4 The Complexity of Hyper^{2}LTL Satisfiability

A Hyper^{2}LTL sentence is satisfiable if it has a model. The Hyper^{2}LTL satisfiability problem asks, given a Hyper^{2}LTL sentence φ, whether φ is satisfiable. In this section, we determine tight bounds on the complexity of Hyper^{2}LTL satisfiability and some of its variants.

Recall that in Section 3, we encoded sets of natural numbers as traces over a set of propositions containing $\mathtt{x}$ and encoded natural numbers as singleton sets. The proof of

Theorem 7 relies on constructing a sentence that requires each of its models to encode every subset of $\mathbb{N}$ by a trace in the model. Hence, sets of traces can encode sets of sets of natural numbers, i.e., type 2 objects.

Another important ingredient in the following proof is the implementation of addition and multiplication in HyperLTL. Let $\mathrm{AP}_{arith} = \{\texttt{arg1}, \texttt{arg2}, \texttt{res}, \texttt{add}, \texttt{mult}\}$ and let $T_{(+,\cdot)}$ be the set of all traces $t \in (2^{\mathrm{AP}_{arith}})^\omega$ such that:

- there are unique $n_1, n_2, n_3 \in \mathbb{N}$ with $\texttt{arg1} \in t(n_1)$, $\texttt{arg2} \in t(n_2)$, and $\texttt{res} \in t(n_3)$, and
- either $\texttt{add} \in t(n)$ and $\texttt{mult} \notin t(n)$ for all n, and $n_1 + n_2 = n_3$, or $\texttt{mult} \in t(n)$ and $\texttt{add} \notin t(n)$ for all n, and $n_1 \cdot n_2 = n_3$.

▶ **Proposition 8** (Theorem 5.5 of [20]). *There is a satisfiable* HyperLTL *sentence $\varphi_{(+,\cdot)}$ such that the AP_{arith}-projection of every model of $\varphi_{(+,\cdot)}$ is $T_{(+,\cdot)}$.*

Combining the capability of quantifying over type 0, type 1, and type 2 objects and the encoding of addition and multiplication, we show that Hyper^{2}LTL and truth in third-order arithmetic have the same complexity.

▶ **Theorem 9.** *The* Hyper^{2}LTL *satisfiability problem is polynomial-time equivalent to truth in third-order arithmetic. The lower bound holds even for X_a-free sentences.*

Proof. We begin with the lower bound by reducing truth in third-order arithmetic to Hyper^{2}LTL satisfiability: we present a polynomial-time translation from sentences φ of third-order arithmetic to Hyper^{2}LTL sentences φ' such that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if φ' is satisfiable.

Given a third-order sentence φ, we define

$$\varphi' = \exists X_{allSets}.\ \exists X_{arith}.\ (\varphi_{allSets}[X_d/X_{allSets}] \wedge \varphi'_{(+,\cdot)} \wedge \mathit{hyp}(\varphi))$$

where

- $\varphi_{allSets}[X_d/X_{allSets}]$ is the Hyper^{2}LTL sentence from the proof of Theorem 7 where every occurrence of X_d is replaced by $X_{allSets}$ and thus enforces every subset of $\mathbb{N}$ to be encoded in the interpretation of $X_{allSets}$ (as introduced in the proof of Theorem 7),
- $\varphi'_{(+,\cdot)}$ is the Hyper^{2}LTL formula obtained from the HyperLTL formula $\varphi_{(+,\cdot)}$ by replacing each quantifier $\exists\pi$ ($\forall\pi$, respectively) by $\exists\pi \in X_{arith}$ ($\forall\pi \in X_{arith}$, respectively) and thus enforces that X_{arith} is interpreted by a set whose AP_{arith}-projection is $T_{(+,\cdot)}$, and

where $\mathit{hyp}(\varphi)$ is defined inductively as follows:

- For third-order variables $\mathcal{Y}$,

$$\mathit{hyp}(\exists\mathcal{Y}.\ \psi) = \exists X_{\mathcal{Y}}.\ (\forall\pi \in X_{\mathcal{Y}}.\ \exists\pi' \in X_{allSets}.\ (\pi =_{\{\texttt{+},\texttt{-},\texttt{s},\texttt{x}\}} \pi') \wedge \mathtt{s}_\pi) \wedge \mathit{hyp}(\psi).$$

- For third-order variables $\mathcal{Y}$,

$$\mathit{hyp}(\forall\mathcal{Y}.\ \psi) = \forall X_{\mathcal{Y}}.\ (\forall\pi \in X_{\mathcal{Y}}.\ \exists\pi' \in X_{allSets}.\ (\pi =_{\{\texttt{+},\texttt{-},\texttt{s},\texttt{x}\}} \pi') \wedge \mathtt{s}_\pi) \rightarrow \mathit{hyp}(\psi).$$

- For second-order variables Y, $\mathit{hyp}(\exists Y.\ \psi) = \exists\pi_Y \in X_{allSets}.\ \mathtt{s}_{\pi_Y} \wedge \mathit{hyp}(\psi)$.
- For second-order variables Y, $\mathit{hyp}(\forall Y.\ \psi) = \forall\pi_Y \in X_{allSets}.\ \mathtt{s}_{\pi_Y} \rightarrow \mathit{hyp}(\psi)$.
- For first-order variables y,

$$\mathit{hyp}(\exists y.\ \psi) = \exists\pi_y \in X_{allSets}.\ \mathtt{s}_{\pi_y} \wedge [(\neg\mathtt{x}_{\pi_y})\,\mathbf{U}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{G}\,\neg\mathtt{x}_{\pi_y})] \wedge \mathit{hyp}(\psi).$$

- For first-order variables y,

$$\mathit{hyp}(\forall y.\ \psi) = \forall\pi_y \in X_{allSets}.\ (\mathtt{s}_{\pi_y} \wedge [(\neg\mathtt{x}_{\pi_y})\,\mathbf{U}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{G}\,\neg\mathtt{x}_{\pi_y})]) \rightarrow \mathit{hyp}(\psi).$$

- $hyp(\psi_1 \vee \psi_2) = hyp(\psi_1) \vee hyp(\psi_2)$.
- $hyp(\neg\psi) = \neg hyp(\psi)$.
- For second-order variables Y and third-order variables $\mathcal{Y}$,

$$hyp(Y \in \mathcal{Y}) = \exists \pi \in X_{\mathcal{Y}}.\ \pi_Y =_{\{\mathtt{x}\}} \pi.$$

- For first-order variables y and second-order variables Y, $hyp(y \in Y) = \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathtt{x}_{\pi_Y})$.
- For first-order variables y, y', $hyp(y < y') = \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{F}\,\mathtt{x}_{\pi_{y'}})$.
- For first-order variables y_1, y_2, y,

$$hyp(y_1 + y_2 = y) = \exists \pi \in X_{arith}.\ \mathtt{add}_\pi \wedge \mathbf{F}(\mathtt{arg1}_\pi \wedge \mathtt{x}_{\pi_{y_1}}) \wedge \mathbf{F}(\mathtt{arg2}_\pi \wedge \mathtt{x}_{\pi_{y_2}}) \wedge \mathbf{F}(\mathtt{res}_\pi \wedge \mathtt{x}_{\pi_y}).$$

- For first-order variables y_1, y_2, y,

$$hyp(y_1 \cdot y_2 = y) = \exists \pi \in X_{arith}.\ \mathtt{mult}_\pi \wedge \mathbf{F}(\mathtt{arg1}_\pi \wedge \mathtt{x}_{\pi_{y_1}}) \wedge \mathbf{F}(\mathtt{arg2}_\pi \wedge \mathtt{x}_{\pi_{y_2}}) \wedge \mathbf{F}(\mathtt{res}_\pi \wedge \mathtt{x}_{\pi_y}).$$

While φ' is not in prenex normal form, it can easily be brought into prenex normal form, as there are no quantifiers under the scope of a temporal operator.

As we are evaluating φ' w.r.t. standard semantics and the variable X_d (interpreted with the model) does not occur in φ', satisfaction of φ' is independent of the model, i.e., for all sets T, T' of traces, $T \models \varphi'$ if and only if $T' \models \varphi'$. So, let us fix some set T of traces. An induction shows that $(\mathbb{N}, +, \cdot, <, \in)$ satisfies φ if and only if T satisfies φ'. Altogether we obtain the desired equivalence between $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ and φ' being satisfiable.

For the upper bound, we conversely reduce Hyper^{2}LTL satisfiability to truth in third-order arithmetic: we present a polynomial-time translation from Hyper^{2}LTL sentences φ to sentences φ' of third-order arithmetic such that φ is satisfiable if and only if $(\mathbb{N}, +, \cdot, <, \in) \models \varphi'$. Here, we assume AP to be fixed, so that we can use $|\mathrm{AP}|$ as a constant in our formulas (which is definable in arithmetic).

Let $pair\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ denote Cantor's pairing function defined as $pair(i,j) = \frac{1}{2}(i+j)(i+j+1)+j$, which is a bijection. Furthermore, fix some bijection $e\colon \mathrm{AP} \to \{0, 1, \ldots, |\mathrm{AP}|-1\}$. Then, we encode a trace $t \in (2^{\mathrm{AP}})^\omega$ by the set $S_t = \{pair(j, e(\mathtt{p})) \mid j \in \mathbb{N} \text{ and } \mathtt{p} \in t(j)\} \subseteq \mathbb{N}$. As $pair$ is a bijection, we have that $t \neq t'$ implies $S_t \neq S_{t'}$. While not every subset of $\mathbb{N}$ encodes some trace t, the first-order formula

$$\varphi_{isTrace}(Y) = \forall x.\ \forall y.\ y \geq |\mathrm{AP}| \to pair(x,y) \notin Y$$

checks if a set does encode a trace. Here, we use $pair$ as syntactic sugar, which is possible as the definition of $pair$ only uses addition and multiplication.

As (certain) sets of natural numbers encode traces, sets of (certain) sets of natural numbers encode sets of traces. This is sufficient to reduce Hyper^{2}LTL to third-order arithmetic, which allows the quantification over sets of sets of natural numbers. Before we present the translation, we need to introduce some more auxiliary formulas:

- Let $\mathcal{Y}$ be a third-order variable (i.e., $\mathcal{Y}$ ranges over sets of sets of natural numbers). Then, the formula

$$\varphi_{onlyTraces}(\mathcal{Y}) = \forall Y.\ Y \in \mathcal{Y} \to \varphi_{isTrace}(Y)$$

  checks if a set of sets of natural numbers only contains sets encoding a trace.
- Further, the formula

$$\varphi_{allTraces}(\mathcal{Y}) = \varphi_{onlyTraces}(\mathcal{Y}) \wedge \forall Y.\ \varphi_{isTrace}(Y) \to Y \in \mathcal{Y}$$

  checks if a set of sets of natural numbers contains exactly the sets encoding a trace.

Now, we are ready to define our encoding of Hyper^{2}LTL in third-order arithmetic. Given a Hyper^{2}LTL sentence φ, let

$$\varphi' = \exists \mathcal{Y}_a.\ \exists \mathcal{Y}_d.\ \varphi_{allTraces}(\mathcal{Y}_a) \wedge \varphi_{onlyTraces}(\mathcal{Y}_d) \wedge (ar(\varphi))(0)$$

where $ar(\varphi)$ is defined inductively as presented below. Note that φ' requires $\mathcal{Y}_a$ to contain exactly the encodings of all traces (i.e., it corresponds to the distinguished Hyper^{2}LTL variable X_a in the following translation) and $\mathcal{Y}_d$ is an existentially quantified set of trace encodings (i.e., it corresponds to the distinguished Hyper^{2}LTL variable X_d in the following translation).

In the inductive definition of $ar(\varphi)$, we will employ a free first-order variable i to denote the position at which the formula is to be evaluated to capture the semantics of the temporal operators. As seen above, in φ', this free variable is set to zero in correspondence with the Hyper^{2}LTL semantics.

- $ar(\exists X.\ \psi) = \exists \mathcal{Y}_X.\ \varphi_{onlyTraces}(\mathcal{Y}_X) \wedge ar(\psi)$. Here, the free variable of $ar(\exists X.\ \psi)$ is the free variable of $ar(\psi)$.
- $ar(\forall X.\ \psi) = \forall \mathcal{Y}_X.\ \varphi_{onlyTraces}(\mathcal{Y}_X) \rightarrow ar(\psi)$. Here, the free variable of $ar(\forall X.\ \psi)$ is the free variable of $ar(\psi)$.
- $ar(\exists \pi \in X.\ \psi) = \exists Y_\pi.\ Y_\pi \in \mathcal{Y}_X \wedge ar(\psi)$. Here, the free variable of $ar(\exists \pi \in X.\ \psi)$ is the free variable of $ar(\psi)$.
- $ar(\forall \pi \in X.\ \psi) = \forall Y_\pi.\ Y_\pi \in \mathcal{Y}_X \rightarrow ar(\psi)$. Here, the free variable of $ar(\forall \pi \in X.\ \psi)$ is the free variable of $ar(\psi)$.
- $ar(\psi_1 \vee \psi_2) = ar(\psi_1) \vee ar(\psi_2)$. Here, we require that the free variables of $ar(\psi_1)$ and $ar(\psi_2)$ are the same (which can always be achieved by variable renaming), which is then also the free variable of $ar(\psi_1 \vee \psi_2)$.
- $ar(\neg\psi) = \neg ar(\psi)$. Here, the free variable of $ar(\neg\psi)$ is the free variable of $\neg ar(\psi)$.
- $ar(\mathbf{X}\,\psi) = \exists i'(i' = i + 1) \wedge ar(\psi)$, where i' is the free variable of $ar(\psi)$ and i is the free variable of $ar(\mathbf{X}\,\psi)$.
- $ar(\psi_1\,\mathbf{U}\,\psi_2) = \exists i_2.\ i_2 \geq i \wedge ar(\psi_2) \wedge \forall i_1.\ (i \leq i_1 \wedge i_1 < i_2) \rightarrow ar(\psi_1)$, where i_j is the free variable of $ar(\psi_j)$, and i is the free variable of $ar(\psi_1\,\mathbf{U}\,\psi_2)$.
- $ar(\mathtt{p}_\pi) = pair(i, e(\mathtt{p})) \in Y_\pi$, i.e., i is the free variable of $ar(\mathtt{p}_\pi)$.

Now, an induction shows that $\Pi_\emptyset[X_a \rightarrow (2^{\mathrm{AP}})^\omega, X_d \mapsto T] \models \varphi$ if and only if $(\mathbb{N}, +, \cdot, <, \in)$ satisfies $(ar(\varphi))(0)$ when the variable $\mathcal{Y}_a$ is interpreted by the encoding of $(2^{\mathrm{AP}})^\omega$ and $\mathcal{Y}_d$ is interpreted by the encoding of T. Hence, φ is indeed satisfiable if and only if $(\mathbb{N}, +, \cdot, <, \in)$ satisfies φ'. ◀

In the lower bound proof above, we have turned a sentence φ of third-order arithmetic into a Hyper^{2}LTL sentence φ' such that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if φ' is satisfiable. In fact, we have constructed φ' such that if it is satisfiable, then every set of traces satisfies it, in particular $(2^{\mathrm{AP}})^\omega$. Recall that Remark 3 states that $(2^{\mathrm{AP}})^\omega$ satisfies φ' under standard semantics if and only if $(2^{\mathrm{AP}})^\omega$ satisfies φ' under closed-world semantics. Thus, altogether we obtain that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if φ' is satisfiable under closed-world semantics, i.e, the lower bound holds even under closed-world semantics. Together with Lemma 2, this settles the complexity of Hyper^{2}LTL satisfiability under closed-world semantics.

▶ **Corollary 10.** *The* Hyper^{2}LTL *satisfiability problem under closed-world semantics is polynomial-time equivalent to truth in third-order arithmetic.*

The Hyper^{2}LTL finite-state satisfiability problem asks, given a Hyper^{2}LTL sentence φ, whether there is a finite transition system satisfying φ. Note that we do not ask for a finite

set T of traces satisfying φ. In fact, the set of traces of the finite transition system may still be infinite or even uncountable. Nevertheless, the problem is potentially simpler, as there are only countably many finite transition systems (and their sets of traces are much simpler). However, we show that the finite-state satisfiability problem is as hard as the general satisfiability problem, as Hyper^{2}LTL allows the quantification over arbitrary (sets of) traces, i.e., restricting the universe of discourse to the traces of a finite transition system does not restrict second-order quantification at all (as the set of all traces is represented by a finite transition system). This has to be contrasted with the finite-state satisfiability problem for HyperLTL (defined analogously), which is Σ_1^0-complete (a.k.a. recursively enumerable), as HyperLTL model-checking of finite transition systems is decidable [11].

▶ **Theorem 11.** *The* Hyper^{2}LTL *finite-state satisfiability problem is polynomial-time equivalent to truth in third-order arithmetic. The lower bound holds even for X_a-free sentences.*

Proof. For the lower bound under standard semantics, we reduce truth in third-order arithmetic to Hyper^{2}LTL finite-state satisfiability: we present a polynomial-time translation from sentences φ of third-order arithmetic to Hyper^{2}LTL sentences φ' such that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if φ' is satisfied by a finite transition system.

So, let φ be a sentence of third-order arithmetic. Recall that in the proof of Theorem 9, we have shown how to construct from φ the Hyper^{2}LTL sentence φ' such that the following three statements are equivalent:

- $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$.
- φ' is satisfiable.
- φ' is satisfied all sets T of traces (and in particular by some finite-state transition system).

Thus, the lower bound follows from Theorem 9.

For the upper bound, we conversely reduce Hyper^{2}LTL finite-state satisfiability to truth in third-order arithmetic: we present a polynomial-time translation from Hyper^{2}LTL sentences φ to sentences φ'' of third-order arithmetic such that φ is satisfied by a finite transition system if and only if $(\mathbb{N}, +, \cdot, <, \in) \models \varphi''$.

Recall that in the proof of Theorem 9, we have constructed a sentence

$$\varphi' = \exists \mathcal{Y}_a.\ \exists \mathcal{Y}_d.\ \varphi_{allTraces}(\mathcal{Y}_a) \wedge \varphi_{onlyTraces}(\mathcal{Y}_d) \wedge (ar(\varphi))(0)$$

of third-order arithmetic where $\mathcal{Y}_a$ represents the distinguished Hyper^{2}LTL variable X_a, $\mathcal{Y}_d$ represents the distinguished Hyper^{2}LTL variable X_d, and where $ar(\varphi)$ is the encoding of φ in Hyper^{2}LTL.

To encode the general satisfiability problem it was sufficient to express that $\mathcal{Y}_d$ only contains traces. Here, we now require that $\mathcal{Y}_d$ contains exactly the traces of some finite transition system, which can easily be expressed in second-order arithmetic[2] as follows.

We begin with a formula $\varphi_{isTS}(n, E, I, \ell)$ expressing that the second-order variables E, I, and ℓ encode a transition system with set $\{0, 1, \ldots, n-1\}$ of vertices. Our encoding will make extensive use of the pairing function introduced in the proof of Theorem 9. Formally, we define $\varphi_{isTS}(n, E, I, \ell)$ as the conjunction of the following formulas (where all quantifiers are first-order and we use *pair* as syntactic sugar):

- $n > 0$: the transition system is nonempty.
- $\forall y.\ y \in E \rightarrow \exists v.\ \exists v'.\ (v < n \wedge v' < n \wedge y = pair(v, v'))$: edges are pairs of vertices.
- $\forall v.\ v < n \rightarrow \exists v'.\ (v' < n \wedge pair(v, v') \in E)$: every vertex has a successor.
- $\forall v.\ v \in I \rightarrow v < n$: the set of initial vertices is a subset of the set of all vertices.

[2] With a little more effort, and a little less readability, first-order suffices for this task, as finite transition systems can be encoded by natural numbers.

- $\forall y.\ y \in \ell \to \exists v.\ \exists p.\ (v < n \wedge p < |\mathrm{AP}| \wedge y = \mathit{pair}(v,p))$: the labeling of v by p is encoded by the pair (v,p). Here, we again assume AP to be fixed and therefore can use $|\mathrm{AP}|$ as a constant.

Next, we define $\varphi_{\mathit{isPath}}(P,n,E,I)$, expressing that the second-order variable P encodes a path through the transition system encoded by n, E, and I, as the conjunction of the following formulas:

- $\forall j.\ \exists v.\ (v < n \wedge \mathit{pair}(j,v) \in P \wedge \neg\exists v'.\ (v' \neq v \wedge \mathit{pair}(j,v') \in P))$: the fact that at position j the path visits vertex v is encoded by the pair (j,v). Exactly one vertex is visited at each position.
- $\exists v.\ v \in I \wedge \mathit{pair}(0,v) \in P$: the path starts in an initial vertex.
- $\forall j.\ \exists v.\ \exists v'.\ \mathit{pair}(j,v) \in P \wedge \mathit{pair}(j+1,v') \in P \wedge \mathit{pair}(v,v') \in E$: successive vertices in the path are indeed connected by an edge.

Finally, we define $\varphi_{\mathit{traceOf}}(T,P,\ell)$, expressing that the second-order variable T encodes the trace (using the encoding from the proof of Theorem 9) of the path encoded by the second-order variable P, as the following formula:

- $\forall j.\ \forall p.\ \mathit{pair}(j,p) \in T \leftrightarrow (\exists v.\ \mathit{pair}(j,v) \in P \wedge \mathit{pair}(v,p) \in \ell)$: a proposition holds in the trace at position j if and only if it is in the labeling of the j-th vertex of the path.

Now, we define the sentence φ'' as

$$\exists \mathcal{Y}_a.\ \exists \mathcal{Y}_d.\ \varphi_{\mathit{allTraces}}(\mathcal{Y}_a) \wedge \varphi_{\mathit{onlyTraces}}(\mathcal{Y}_d) \wedge \Big[\underbrace{\exists n.\ \exists E.\ \exists I.\ \exists \ell.\ \varphi_{\mathit{isTS}}(n,E,I,\ell)}_{\text{there exists a transition system } \mathfrak{T}} \wedge \underbrace{(\forall T.\ T \in \mathcal{Y}_d \to \exists P.\ (\varphi_{\mathit{isPath}}(P,n,E,I) \wedge \varphi_{\mathit{traceOf}}(T,P,\ell)))}_{\mathcal{Y}_d \text{ contains only traces of paths through } \mathfrak{T}} \wedge \underbrace{(\forall P.\ (\varphi_{\mathit{isPath}}(P,n,E,I) \to \exists T.\ T \in \mathcal{Y}_d \wedge \varphi_{\mathit{traceOf}}(T,P,\ell)))}_{\mathcal{Y}_d \text{ contains all traces of paths through } \mathfrak{T}.} \Big] \wedge (\mathit{ar}(\varphi))(0),$$

which holds in $(\mathbb{N},+,\cdot,<,\in)$ if and only if φ is satisfied by a finite transition system. ◀

Again, the lower bound proof can easily be extended to the case of closed-world semantics, using the same arguments as in the case of general satisfiability.

▶ **Corollary 12.** *The* Hyper^{2}LTL *finite-state satisfiability problem under closed-world semantics is polynomial-time equivalent to truth in third-order arithmetic.*

5 The Complexity of Hyper^{2}LTL Model-Checking

The Hyper^{2}LTL model-checking problem asks, given a finite transition system $\mathfrak{T}$ and a Hyper^{2}LTL sentence φ, whether $\mathfrak{T} \models \varphi$. Beutner et al. [4] have shown that Hyper^{2}LTL model-checking is Σ^1_1-hard, but there is no known upper bound in the literature. We improve the lower bound considerably, i.e., also to truth in third-order arithmetic, and show that this bound is tight. This is the first upper bound on the problem's complexity.

▶ **Theorem 13.** *The* Hyper^{2}LTL *model-checking problem is polynomial-time equivalent to truth in third-order arithmetic. The lower bound already holds for X_a-free sentences.*

Proof. For the lower bound, we reduce truth in third-order arithmetic to the Hyper^{2}LTL model-checking problem: we present a polynomial-time translation from sentences φ of third-order arithmetic to pairs $(\mathfrak{T}, \varphi')$ of a finite transition system $\mathfrak{T}$ and a Hyper^{2}LTL sentence φ' such that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if $\mathfrak{T} \models \varphi'$.

In the proof of Theorem 9 we have, given a sentence φ of third-order arithmetic, constructed a Hyper^{2}LTL sentence φ' such that $(\mathbb{N}, +, \cdot, <, \in) \models \varphi$ if and only if every set T of traces satisfies φ' (i.e., satisfaction is independent of the model). Thus, we obtain the lower bound by mapping φ to φ' and $\mathfrak{T}^*$, where $\mathfrak{T}^*$ is some fixed transition system.

For the upper bound, we reduce the Hyper^{2}LTL model-checking problem to truth in third-order arithmetic: we present a polynomial-time translation from pairs $(\mathfrak{T}, \varphi)$ of a finite transition system and a Hyper^{2}LTL sentence φ to sentences φ' of third-order arithmetic such that $\mathfrak{T} \models \varphi$ if and only if $(\mathbb{N}, +, \cdot, <, \in) \models \varphi'$.

In the proof of Theorem 11, we have constructed, from a Hyper^{2}LTL sentence φ, a sentence φ' of third-order arithmetic that expresses the existence of a finite transition system that satisfies φ. We obtain the desired upper bound by modifying φ' to replace the existential quantification of the transition system by hardcoding $\mathfrak{T}$ instead. ◀

Again, the lower bound proof can easily be extended to closed-world semantics, using the same arguments as in the case of satisfiability.

▶ **Corollary 14.** *The* Hyper^{2}LTL *model-checking problem under closed-world semantics is polynomial-time equivalent to truth in third-order arithmetic.*

6 Hyper^{2}LTL$_{\mathrm{mm}}$

As we have seen, unrestricted second-order quantification makes Hyper^{2}LTL very expressive and therefore highly undecidable. But restricted forms of second-order quantification are sufficient for many application areas. Beutner et al. [4] introduced Hyper^{2}LTL$_{\mathrm{mm}}$, a fragment[3] of Hyper^{2}LTL in which second-order quantification ranges over smallest/largest sets that satisfy a given guard. For example, the formula $\exists(X, \curlyvee, \varphi_1).\ \varphi_2$ expresses that there is a set T of traces that satisfies both φ_1 and φ_2, and T is *a* smallest set that satisfies φ_1 (i.e., φ_1 is the guard). This fragment is expressive enough to express common knowledge, asynchronous hyperproperties, and causality in reactive systems [4].

The formulas of Hyper^{2}LTL$_{\mathrm{mm}}$ are given by the grammar

$$\varphi ::= \exists(X, \between, \varphi).\ \varphi \mid \forall(X, \between, \varphi).\ \varphi \mid \exists \pi \in X.\ \varphi \mid \forall \pi \in X.\ \varphi \mid \psi$$
$$\psi ::= \mathtt{p}_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi$$

where $\mathtt{p}$ ranges over AP, π ranges over $\mathcal{V}_1$, X ranges over $\mathcal{V}_2$, and $\between \in \{\curlyvee, \curlywedge\}$, i.e., the only modification concerns the syntax of second-order quantification.

Accordingly, the semantics of Hyper^{2}LTL$_{\mathrm{mm}}$ is similar to that of Hyper^{2}LTL but for the second-order quantifiers, for which we define (for $\between \in \{\curlyvee, \curlywedge\}$):

- $\Pi \models \exists(X, \between, \varphi_1).\ \varphi_2$ if there exists a set $T \in \mathrm{sol}(\Pi, (X, \between, \varphi_1))$ such that $\Pi[X \mapsto T] \models \varphi_2$
- $\Pi \models \forall(X, \between, \varphi_1).\ \varphi_2$ if for all sets $T \in \mathrm{sol}(\Pi, (X, \between, \varphi_1))$ we have $\Pi[X \mapsto T] \models \varphi_2$

[3] In [4] this fragment is termed Hyper^{2}LTL$_{\mathrm{fp}}$.

Here, $\mathrm{sol}(\Pi, (X, \between, \varphi_1))$ is the set of all minimal/maximal models of the formula φ_1, which is defined as follows:

$$\mathrm{sol}(\Pi, (X, \curlyvee, \varphi_1)) = \{T \subseteq (2^{\mathrm{AP}})^\omega \mid \Pi[X \mapsto T] \models \varphi_1 \text{ and } \Pi[X \mapsto T'] \not\models \varphi_1 \text{ for all } T' \subsetneq T\}$$
$$\mathrm{sol}(\Pi, (X, \curlywedge, \varphi_1)) = \{T \subseteq (2^{\mathrm{AP}})^\omega \mid \Pi[X \mapsto T] \models \varphi_1 \text{ and } \Pi[X \mapsto T'] \not\models \varphi_1 \text{ for all } T' \supsetneq T\}$$

Note that $\mathrm{sol}(\Pi, (X, \between, \varphi_1))$ may be empty, may be a singleton, or may contain multiple sets, which then are pairwise incomparable.

Let us also define closed-world semantics for Hyper^{2}LTL$_{\mathrm{mm}}$. Here, we again disallow the use of the variable X_a and change the semantics of set quantification to

- $\Pi \models_{\mathrm{cw}} \exists(X, \between, \varphi_1).\ \varphi_2$ if there exists a set $T \in \mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \between, \varphi_1))$ such that $\Pi[X \mapsto T] \models \varphi_2$, and
- $\Pi \models_{\mathrm{cw}} \forall(X, \between, \varphi_1).\ \varphi_2$ if for all sets $T \in \mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \between, \varphi_1))$ we have $\Pi[X \mapsto T] \models \varphi_2$,

where $\mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \curlyvee, \varphi_1))$ and $\mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \curlywedge, \varphi_1))$ are defined as follows:

$$\mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \curlyvee, \varphi_1)) = \{T \subseteq \Pi(X_d) \mid \Pi[X \mapsto T] \models_{\mathrm{cw}} \varphi_1 \text{ and } \Pi[X \mapsto T'] \not\models_{\mathrm{cw}} \varphi_1 \text{ for all } T' \subsetneq T\}$$
$$\mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \curlywedge, \varphi_1)) = \{T \subseteq \Pi(X_d) \mid \Pi[X \mapsto T] \models_{\mathrm{cw}} \varphi_1 \text{ and } \Pi[X \mapsto T'] \not\models_{\mathrm{cw}} \varphi_1 \text{ for all } \Pi(X_d) \supseteq T' \supsetneq T\}.$$

Note that $\mathrm{sol}_{\mathrm{cw}}(\Pi, (X, \between, \varphi_1))$ may still be empty, may be a singleton, or may contain multiple sets, but all sets in it are now incomparable subsets of $\Pi(X_d)$.

A Hyper^{2}LTL$_{\mathrm{mm}}$ formula is a sentence if it does not have any free variables except for X_a and X_d (also in the guards). Models are defined as for Hyper^{2}LTL.

▶ **Proposition 15** (Proposition 1 of [4]). *Every* Hyper^{2}LTL$_{\mathrm{mm}}$ *sentence* φ *can be translated in polynomial time (in* $|\varphi|$*) into a* Hyper^{2}LTL *sentence* φ' *such that for all sets* T *of traces we have that* $T \models \varphi$ *if and only if* $T \models \varphi'$.[4]

The same claim is also true for closed-world semantics, using the same proof.

▶ Remark 16. Every Hyper^{2}LTL$_{\mathrm{mm}}$ sentence φ can be translated in polynomial time (in $|\varphi|$) into a Hyper^{2}LTL sentence φ' such that for all sets T of traces we have that $T \models_{\mathrm{cw}} \varphi$ if and only if $T \models_{\mathrm{cw}} \varphi'$.

Thus, every complexity upper bound for Hyper^{2}LTL also holds for Hyper^{2}LTL$_{\mathrm{mm}}$ and every lower bound for Hyper^{2}LTL$_{\mathrm{mm}}$ also holds for Hyper^{2}LTL. In the following, we show that lower bounds can also be transferred in the other direction, i.e., from Hyper^{2}LTL to Hyper^{2}LTL$_{\mathrm{mm}}$. Thus, contrary to the design goal of Hyper^{2}LTL$_{\mathrm{mm}}$, it is in general not more feasible than full Hyper^{2}LTL.

We begin again by studying the cardinality of models of Hyper^{2}LTL$_{\mathrm{mm}}$ sentences, which will be the key technical tool for our complexity results. Again, as such formulas are evaluated over sets of traces, whose cardinality is bounded by $\mathfrak{c}$, there is a trivial upper bound. Our main result is that this bound is tight even for the restricted setting of Hyper^{2}LTL$_{\mathrm{mm}}$. The proof is similar to the one of Theorem 7, we just have to modify φ_4 so that the universal second-order quantifier only ranges over maximal contradiction-free sets.

[4] The polynomial-time claim is not made in [4], but follows from the construction when using appropriate data structures for formulas.

▶ **Theorem 17.** *There is a satisfiable X_a-free* Hyper^{2}LTL$_{\text{mm}}$ *sentence that only has models of cardinality* $\mathfrak{c}$ *(under standard and closed-world semantics).*

Now, let us describe how we settle the complexity of Hyper^{2}LTL$_{\text{mm}}$ satisfiability and model-checking: Recall that Hyper^{2}LTL allows set quantification over arbitrary sets of traces while Hyper^{2}LTL$_{\text{mm}}$ restricts quantification to minimal/maximal sets of traces that satisfy a guard formula. By using a sentence $\varphi_{\mathfrak{c}}$ as guard that has only models of cardinality $\mathfrak{c}$, the minimal sets satisfying the guard have cardinality $\mathfrak{c}$. Thus, we can obtain every possible set over propositions not used by $\varphi_{\mathfrak{c}}$ as the projection of a subset of a minimal set satisfying the guard $\varphi_{\mathfrak{c}}$. Thus, quantification of arbitrary sets of traces can be mimicked by quantification of minimal and maximal sets satisfying a guard.

▶ **Theorem 18.** Hyper^{2}LTL$_{\text{mm}}$ *satisfiability, finite-state satisfiability, and model-checking are polynomial-time equivalent to truth in third-order arithmetic. The lower bounds hold even for X_a-free sentences.*

Let us conclude by mentioning that Theorem 18 can again be extended to Hyper^{2}LTL$_{\text{mm}}$ under closed-world semantics, using the same arguments as for full Hyper^{2}LTL.

▶ **Corollary 19.** Hyper^{2}LTL$_{\text{mm}}$ *satisfiability, finite-state satisfiability, and model-checking under closed-world semantics are polynomial-time equivalent to truth in third-order arithmetic.*

7 The Least Fixed Point Fragment of Hyper^{2}LTL$_{\text{mm}}$

We have seen that even restricting second-order quantification to smallest/largest sets that satisfy a guard formula is essentially as expressive as full Hyper^{2}LTL, and thus as difficult. However, Beutner et al. [4] note that applications like common knowledge and asynchronous hyperproperties do not even require quantification over smallest/largest sets satisfying a guard, they "only" require quantification over least fixed points of HyperLTL definable functions. This finally yields a fragment with (considerably) lower complexity: we show that satisfiability under closed-world semantics is Σ^1_1-complete while finite-state satisfiability and model-checking are in Σ^2_2 and Σ^1_1-hard (under both semantics). For satisfiability under closed-world semantics, this matches the complexity of HyperLTL satisfiability.

A Hyper^{2}LTL$_{\text{mm}}$ sentence using only minimality constraints has the form

$$\varphi = \gamma_1.\ Q_1(Y_1, \Upsilon, \varphi_1^{\text{con}}).\ \gamma_2.\ Q_2(Y_2, \Upsilon, \varphi_2^{\text{con}}).\ \dots \gamma_k.\ Q_k(Y_k, \Upsilon, \varphi_k^{\text{con}}).\ \gamma_{k+1}.\ \psi$$

satisfying the following properties:

- Each γ_j is a block $\gamma_j = Q_{\ell_{j-1}+1}\pi_{\ell_{j-1}+1} \in X_{\ell_{j-1}+1} \cdots Q_{\ell_j}\pi_{\ell_j} \in X_{\ell_j}$ of trace quantifiers (with $\ell_0 = 0$). As φ is a sentence, this implies that we have $\{X_{\ell_j+1}, \dots, X_{\ell_j}\} \subseteq \{X_a, X_d, Y_1, \dots, Y_{j-1}\}$.
- The free variables of ψ_j^{con} are among the trace variables quantified in the $\gamma_{j'}$ and $X_a, X_d, Y_1, \dots, Y_j$.
- ψ is a quantifier-free formula. Again, as φ is a sentence, the free variables of ψ are among the trace variables quantified in the γ_j.

Now, φ is an lfp-Hyper^{2}LTL$_{\text{mm}}$ sentence[5], if additionally each φ_j^{con} has the form

$$\varphi_j^{\text{con}} = \dot{\pi}_1 \triangleright Y_j \wedge \dots \wedge \dot{\pi}_n \triangleright Y_j \wedge \forall \ddot{\pi}_1 \in Z_1.\ \dots \forall \ddot{\pi}_{n'} \in Z_{n'}.\ \psi_j^{\text{step}} \rightarrow \ddot{\pi}_m \triangleright Y_j$$

[5] Our definition here differs slightly from the one of [4] in that we allow to express the existence of some traces in the fixed point (via the subformulas $\dot{\pi}_i \triangleright Y_j$). All examples and applications of [4] are also of this form.

for some $n \geq 0$, $n' \geq 1$, where $1 \leq m \leq n'$, and where we have

- $\{\dot{\pi}_1, \dots, \dot{\pi}_n\} \subseteq \{\pi_1, \dots, \pi_{\ell_j}\}$,
- $\{Z_1, \dots, Z_{n'}\} \subseteq \{X_a, X_d, Y_1, \dots, Y_j\}$, and
- ψ_j^{step} is quantifier-free with free variables among $\ddot{\pi}_1, \dots, \ddot{\pi}_{n'}, \pi_1, \dots, \pi_{\ell_j}$.

As always, φ_j^{con} can be brought into the required prenex normal form.

Let us give some intuition for the definition. To this end, fix some $j \in \{1, 2, \dots, k\}$ and a variable assignment Π whose domain contains at least all variables quantified before Y_j, i.e., all $Y_{j'}$ and all variables in the $\gamma_{j'}$ for $j' < j$, as well as X_a and X_d. Then,

$$\varphi_j^{\text{con}} = \dot{\pi}_1 \in Y_j \wedge \dots \wedge \dot{\pi}_n \in Y_j \wedge \big(\forall \ddot{\pi}_1 \in Z_1.\ \dots \forall \ddot{\pi}_{n'} \in Z_{n'}.\ \psi_j^{\text{step}} \rightarrow \ddot{\pi}_m \triangleright Y_j\big)$$

induces the monotonic function $f_{\Pi,j} \colon 2^{(2^{\text{AP}})^\omega} \to 2^{(2^{\text{AP}})^\omega}$ defined as

$$f_{\Pi,j}(S) = S \cup \{\Pi(\dot{\pi}_1), \dots, \Pi(\dot{\pi}_n)\} \cup \{\Pi'(\ddot{\pi}_m) \mid \Pi' = \Pi[\ddot{\pi}_1 \mapsto t_1, \dots, \ddot{\pi}_{n'} \mapsto t_{n'}] \\ \text{for } t_i \in \Pi(Z_i) \text{ if } Z_i \neq Y_j \text{ and } t_i \in S \text{ if } Z_i = Y_j \text{ s.t. } \Pi' \models \psi_j^{\text{step}}\}.$$

We define $S_0 = \emptyset$, $S_{\ell+1} = f_{\Pi,j}(S_\ell)$, and

$$\text{lfp}(\Pi, j) = \bigcup\nolimits_{\ell \in \mathbb{N}} S_\ell,$$

which is the least fixed point of $f_{\Pi,j}$. Due to the minimality constraint on Y_j in φ, $\text{lfp}(\Pi, j)$ is the unique set in $\text{sol}(\Pi, (Y_j, \Upsilon, \varphi_j^{\text{con}}))$. Hence, an induction shows that $\text{lfp}(\Pi, j)$ only depends on the values $\Pi(\pi)$ for trace variables π quantified before Y_j as well as the values $\Pi(X_d)$ and $\Pi(X_a)$, but not on the values $\Pi(Y_{j'})$ for $j' < j$ (as they are unique).

Thus, as $\text{sol}(\Pi, (Y_j, \Upsilon, \varphi_j^{\text{con}}))$ is a singleton, it is irrelevant whether Q_j is an existential or a universal quantifier. Instead of interpreting second-order quantification as existential or universal, here one should understand it as a deterministic least fixed point computation: choices for the trace variables and the two distinguished second-order variables uniquely determine the set of traces that a second-order quantifier assigns to a second-order variable.

▶ Remark 20. Note that the traces that are added to a fixed point assigned to Y_j either come from another $Y_{j'}$ with $j' < j$, from the model (via X_d), or from the set of all traces (via X_a). Thus, for X_a-free formulas, all second-order quantifiers range over (unique) subsets of the model, i.e., there is no need for an explicit definition of closed-world semantics. The analogue of closed-world semantics for lfp-Hyper^{2}LTL$_{\text{mm}}$ is to restrict oneself to X_a-free sentences.

In the remainder of this section, we study the complexity of lfp-Hyper^{2}LTL$_{\text{mm}}$. For satisfiability, the key step is again to study the size of models of satisfiable sentences. For X_a-free lfp-Hyper^{2}LTL$_{\text{mm}}$, as for HyperLTL, we are able to show that each satisfiable sentence has a countable model. The following result is proven by generalizing the proof for the analogous result for HyperLTL [18] showing that every model T of a HyperLTL sentence φ contains a countable $R \subseteq T$ that is closed under the application of Skolem functions. This implies that R is also a model of φ.

▶ **Lemma 21.** *Every satisfiable X_a-free* lfp-Hyper^{2}LTL$_{\text{mm}}$ *sentence has a countable model.*

Proof Sketch. Let $\varphi = \gamma_1 Q_1(Y_1, \Upsilon, \varphi_1^{\text{con}}).\ \gamma_2 Q_2(Y_2, \Upsilon, \varphi_2^{\text{con}}).\ \dots \gamma_k Q_2(Y_k, \Upsilon, \varphi_k^{\text{con}}).\ \gamma_{k+1}.\ \psi$ be a satisfiable lfp-Hyper^{2}LTL$_{\text{mm}}$ sentence where

$$\varphi_j^{\text{con}} = \dot{\pi}_1 \triangleright Y_j \wedge \dots \wedge \dot{\pi}_n \triangleright Y_j \wedge \forall \ddot{\pi}_1 \in Z_1.\ \dots \forall \ddot{\pi}_{n'} \in Z_{n'}.\ \psi_j^{\text{step}} \rightarrow \ddot{\pi}_m \triangleright Y_j.$$

We assume w.l.o.g. that each trace variable is quantified at most once in φ. This implies that for each trace variable π quantified in some γ_j or in some φ_j^{con}, there is a unique second-order variable X_π such that π ranges over X_π.

Membership of traces in least fixed points assigned to the variables Y_j can be characterized by trees labeled by traces that make the inductive construction of the stages of the least fixed points explicit. Intuitively, consider the formula φ_j^{con} above inducing the unique least fixed point $\text{lfp}(\Pi, j)$ that Y_j ranges over. It expresses that a trace t is in the fixed point either because it is of the form $\Pi(\dot{\pi}_i)$ for some $i \in \{1, \dots, n\}$ where $\dot{\pi}_i$ is a trace variable quantified before the quantification of Y_j, or t is in the fixed point because there are traces $t_1, \dots, t_{n'}$ such that assigning them to the $\ddot{\pi}_i$ satisfies ψ_j^{step} and $t = t_m$. Thus, the traces $t_1, \dots, t_{n'}$ witness that t is in the fixed point. However, each t_i must be selected from $\Pi(Z_i)$, which, if $Z_i = Y_{j'}$ for some j', again needs witnesses. Thus, a witness is in general a tree whose vertices are labeled by traces and indexes in $\{1, 2,' ldots, k\}$ indicating in which fixed point the trace is in.

As φ is satisfiable, there exists a set T of traces such that $T \models \varphi$. We show that there is a countable $R \subseteq T$ with $R \models \varphi$. Intuitively, we show that the smallest set R that is closed under the application of the Skolem functions and that contains the traces labeling witness trees (for the fixed points computed w.r.t. T) for the traces in R has the desired properties.

The full proof requires additional notation, e.g., a formalization of the notion of witness trees, and can be found in the full version [21]. ◀

Before we continue with our complexity results, let us briefly mention that the formula from Remark 5 on Page 7 shows that the restriction to X_a-free sentences is essential to obtain the upper bound above.

With this upper bound, we can express the existence of (w.l.o.g.) countable models of a given X_a-free sentence φ via arithmetic formulas that only use existential quantification of type 1 objects (sets of natural numbers), which are rich enough to express countable sets T of traces and objects (e.g., Skolem functions and more) witnessing that T satisfies φ. This places satisfiability in Σ_1^1 while the matching lower bound already holds for HyperLTL [19].

▶ **Theorem 22.** lfp-Hyper^{2}LTL$_{\text{mm}}$ *satisfiability for X_a-free sentences is Σ_1^1-complete.*

Proof Sketch. The Σ_1^1 lower bound already holds for HyperLTL satisfiability [19], as HyperLTL is a fragment of X_a-free lfp-Hyper^{2}LTL$_{\text{mm}}$ (see Remark 1). Hence, we focus in the following on the upper bound, which is a generalization of the corresponding upper bound for HyperLTL [19].

Let φ be an X_a-free lfp-Hyper^{2}LTL$_{\text{mm}}$ sentence. From Lemma 21, φ is satisfiable if and only if it has a countable model T. Thus, to prove that the lfp-Hyper^{2}LTL$_{\text{mm}}$ satisfiability problem for X_a-free sentences is in Σ_1^1, we express the existence of a countable set T of traces and a witness that T is indeed a model of φ.

As we want to show a Σ_1^1 upper bound, we have to express the existence of a countable model by a sentence of arithmetic with existential quantification over sets of natural numbers and existential and universal quantification over natural numbers. A bit more in detail, since we only have to work with countable sets (as second-order quantifiers in φ range over subsets of the countable model), we can use natural numbers to "name" traces. Thus, a countable set of traces is a mapping from $\mathbb{N} \times \mathbb{N}$ (names and positions) to 2^{AP}, which can be encoded by a set of natural numbers. Then, we can encode the existence of the following type 1 objects:

- Variable assignments, such that membership of their assigned traces into respective fixed point sets can be captured in first-order arithmetic.
- Functions for the existentially quantified first-order variables of φ, which can be verified to be Skolem functions (in first-order arithmetic).
- Functions expressing the satisfaction of subformulas of φ.

Furthermore, first-order arithmetic can express that the variable assignments indeed map set variables to least fixed points.

Altogether, this allows us to capture the satisfiability of lfp-Hyper^{2}LTL$_{\text{mm}}$ in Σ^1_1. ◀

Finally, we consider finite-state satisfiability and model-checking. Note that we have to deal with uncountable sets of traces in both problems, as the sets of traces of finite transition systems may be uncountable. The lower bounds are proven by reductions from a variant of the recurrent tiling problem [24] while the upper bounds are obtained by expressing least fixed points in second-order arithmetic.

▶ **Theorem 23.** lfp-Hyper^{2}LTL$_{\text{mm}}$ *finite-state satisfiability and model-checking are both in* Σ^2_2 *and* Σ^1_1*-hard, where the lower bounds already hold for* X_a*-free sentences.*

8 Related Work

As mentioned in Section 1, the complexity problems for HyperLTL were thoroughly studied [16, 19, 20]. For Hyper^{2}LTL, Beutner et al. mainly focused on the algorithmic aspects by providing model checking [4] and monitoring [5] algorithms, and did not study the respective complexity problems in depth.

Logics related to Hyper^{2}LTL are asynchronous and epistemic logics. Much research has been done regarding epistemic properties [13, 15, 29, 36] and their relations to hyperproperties [8]. However, most of this work concerns expressiveness and decidability results (e.g., [7]), and not complexity analysis for the undecidable fragments. This is similar for asynchronous hyperlogics [1, 2, 3, 6, 9, 10, 23, 26, 27, 28], where most work concerns decidability results and expressive power, but not complexity analysis.

Another related logic is TeamLTL [28], a hyperlogic for the specification of dependence and independence. Lück [30] studied similar problems to those we study in this paper and showed that, in general, satisfiability and model checking of TeamLTL with Boolean negation is equivalent to truth in third-order arithmetic. Kontinen and Sandström [25] generalize this result and show that any logic between TeamLTL with Boolean negation and second-order logic inherits the same complexity results. Kontinen et al. [26] study set semantics for asynchronous TeamLTL, and provide positive complexity and decidability results. Gutsfeld et al. [22] study an extension of TeamLTL to express refined notions of asynchronicity and analyze the expressiveness and complexity of their logic, proving it also highly undecidable. While TeamLTL is closely related to Hyper^{2}LTL, the exact relation between them is still unknown.

9 Conclusion

We have investigated and settled the complexity of satisfiability, finite-state satisfiability, and model-checking for Hyper^{2}LTL and Hyper^{2}LTL$_{\text{mm}}$ and (almost) settled it for lfp-Hyper^{2}LTL$_{\text{mm}}$. For the former two, all three problems are equivalent to truth in third-order arithmetic, and therefore (not surprisingly) much harder than the corresponding problems for HyperLTL, which are "only" Σ^1_1-complete, Σ^0_1-complete, and TOWER-complete, respectively. This shows that the addition of second-order quantification increases the already high complexity of HyperLTL significantly. However, for the fragment lfp-Hyper^{2}LTL$_{\text{mm}}$, in which second-order quantification degenerates to least fixed point computations, the

complexity is much lower: satisfiability under closed-world semantics is Σ^1_1-complete and finite-state satisfiability as well as model-checking are in Σ^2_2.

Recently, Regaud and Zimmermann [34] have solved several problems left open in this work, e.g., they settled the complexity of Hyper^{2}LTL$_{\text{mm}}$ with only minimality constraints or only maximality constraints, the complexity of lfp-Hyper^{2}LTL$_{\text{mm}}$ under standard semantics, and closed the gaps in our results for lfp-Hyper^{2}LTL$_{\text{mm}}$ finite-state satisfiability and model-checking. Furthermore, they settled the complexity of all three decision problems we consider here for HyperQPTL [33].

References

1 Ezio Bartocci, Thomas A. Henzinger, Dejan Nickovic, and Ana Oliveira da Costa. Hypernode automata. In Guillermo A. Pérez and Jean-François Raskin, editors, *CONCUR 2023*, volume 279 of *LIPIcs*, pages 21:1–21:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.CONCUR.2023.21.

2 Jan Baumeister, Norine Coenen, Borzoo Bonakdarpour, Bernd Finkbeiner, and César Sánchez. A temporal logic for asynchronous hyperproperties. In Alexandra Silva and K. Rustan M. Leino, editors, *CAV 2021, Part I*, volume 12759 of *LNCS*, pages 694–717. Springer, 2021. doi:10.1007/978-3-030-81685-8_33.

3 Raven Beutner and Bernd Finkbeiner. HyperATL*: A logic for hyperproperties in multi-agent systems. *Log. Methods Comput. Sci.*, 19(2), 2023. doi:10.46298/LMCS-19(2:13)2023.

4 Raven Beutner, Bernd Finkbeiner, Hadar Frenkel, and Niklas Metzger. Second-order hyperproperties. In Constantin Enea and Akash Lal, editors, *CAV 2023, Part II*, volume 13965 of *LNCS*, pages 309–332. Springer, 2023. doi:10.1007/978-3-031-37703-7_15.

5 Raven Beutner, Bernd Finkbeiner, Hadar Frenkel, and Niklas Metzger. Monitoring second-order hyperproperties. In Mehdi Dastani, Jaime Simão Sichman, Natasha Alechina, and Virginia Dignum, editors, *AAMAS 2024*, pages 180–188. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2024. doi:10.5555/3635637.3662865.

6 Alberto Bombardelli, Laura Bozzelli, César Sánchez, and Stefano Tonetta. Unifying asynchronous logics for hyperproperties. *arXiv*, 2404.16778, 2024. doi:10.48550/arXiv.2404.16778.

7 Laura Bozzelli, Bastien Maubert, and Aniello Murano. On the complexity of model checking knowledge and time. *ACM Trans. Comput. Log.*, 25(1):8:1–8:42, 2024. doi:10.1145/3637212.

8 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Unifying hyper and epistemic temporal logics. In Andrew M. Pitts, editor, *FoSSaCS 2015*, volume 9034 of *LNCS*, pages 167–182. Springer, 2015. doi:10.1007/978-3-662-46678-0_11.

9 Laura Bozzelli, Adriano Peron, and César Sánchez. Asynchronous extensions of HyperLTL. In *LICS 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470583.

10 Laura Bozzelli, Adriano Peron, and César Sánchez. Expressiveness and decidability of temporal logics for asynchronous hyperproperties. In Bartek Klin, Slawomir Lasota, and Anca Muscholl, editors, *CONCUR 2022*, volume 243 of *LIPIcs*, pages 27:1–27:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CONCUR.2022.27.

11 Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In Martín Abadi and Steve Kremer, editors, *POST 2014*, volume 8414 of *LNCS*, pages 265–284. Springer, 2014. doi:10.1007/978-3-642-54792-8_15.

12 Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *J. Comput. Secur.*, 18(6):1157–1210, 2010. doi:10.3233/JCS-2009-0393.

13 Catalin Dima. Revisiting satisfiability and model-checking for CTLK with synchrony and perfect recall. In Michael Fisher, Fariba Sadri, and Michael Thielscher, editors, *CLIMA IX*, volume 5405 of *LNCS*, pages 117–131. Springer, 2008. doi:10.1007/978-3-642-02734-5_8.

14 E. Allen Emerson and Joseph Y. Halpern. "sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986. doi:10.1145/4904.4999.

15 Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. MIT Press, 1995. doi:10.7551/MITPRESS/5803.001.0001.

16 Bernd Finkbeiner and Christopher Hahn. Deciding hyperproperties. In Josée Desharnais and Radha Jagadeesan, editors, *CONCUR 2016*, volume 59 of *LIPIcs*, pages 13:1–13:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.13.

17 Bernd Finkbeiner, Markus N. Rabe, and César Sánchez. Algorithms for Model Checking HyperLTL and HyperCTL*. In Daniel Kroening and Corina S. Pasareanu, editors, *CAV 2015, Part I*, volume 9206 of *LNCS*, pages 30–48. Springer, 2015. doi:10.1007/978-3-319-21690-4_3.

18 Bernd Finkbeiner and Martin Zimmermann. The First-Order Logic of Hyperproperties. In *STACS 2017*, volume 66 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.30.

19 Marie Fortin, Louwe B. Kuijer, Patrick Totzke, and Martin Zimmermann. HyperLTL satisfiability is Σ^1_1-complete, HyperCTL* satisfiability is Σ^2_1-complete. In Filippo Bonchi and Simon J. Puglisi, editors, *MFCS 2021*, volume 202 of *LIPIcs*, pages 47:1–47:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.MFCS.2021.47.

20 Marie Fortin, Louwe B. Kuijer, Patrick Totzke, and Martin Zimmermann. HyperLTL satisfiability is highly undecidable, HyperCTL* is even harder. *arXiv*, 2303.16699, 2023. Journal version of [19]. Under submission. doi:10.48550/arXiv.2303.16699.

21 Hadar Frenkel and Martin Zimmermann. The complexity of second-order HyperLTL. *arXiv*, 2311.15675, 2023. doi:10.48550/arXiv.2311.15675.

22 Jens Oliver Gutsfeld, Arne Meier, Christoph Ohrem, and Jonni Virtema. Temporal team semantics revisited. In Christel Baier and Dana Fisman, editors, *LICS 2022*, pages 44:1–44:13. ACM, 2022. doi:10.1145/3531130.3533360.

23 Jens Oliver Gutsfeld, Markus Müller-Olm, and Christoph Ohrem. Automata and fixpoints for asynchronous hyperproperties. *Proc. ACM Program. Lang.*, 5(POPL):1–29, 2021. doi:10.1145/3434319.

24 David Harel. Recurring Dominoes: Making the Highly Undecidable Highly Understandable. *North-Holland Mathematical Studies*, 102:51–71, 1985. doi:10.1016/S0304-0208(08)73075-5.

25 Juha Kontinen and Max Sandström. On the expressive power of TeamLTL and first-order team logic over hyperproperties. In Alexandra Silva, Renata Wassermann, and Ruy J. G. B. de Queiroz, editors, *WoLLIC 2021*, volume 13038 of *LNCS*, pages 302–318. Springer, 2021. doi:10.1007/978-3-030-88853-4_19.

26 Juha Kontinen, Max Sandström, and Jonni Virtema. Set semantics for asynchronous TeamLTL: Expressivity and complexity. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *MFCS 2023*, volume 272 of *LIPIcs*, pages 60:1–60:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.MFCS.2023.60.

27 Juha Kontinen, Max Sandström, and Jonni Virtema. A remark on the expressivity of asynchronous TeamLTL and HyperLTL. In Arne Meier and Magdalena Ortiz, editors, *FoIKS 2024*, volume 14589 of *LNCS*, pages 275–286. Springer, 2024. doi:10.1007/978-3-031-56940-1_15.

28 Andreas Krebs, Arne Meier, Jonni Virtema, and Martin Zimmermann. Team semantics for the specification and verification of hyperproperties. In Igor Potapov, Paul G. Spirakis, and James Worrell, editors, *MFCS 2018*, volume 117 of *LIPIcs*, pages 10:1–10:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.10.

29 Alessio Lomuscio and Franco Raimondi. The complexity of model checking concurrent programs against CTLK specifications. In Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss,

and Peter Stone, editors, *AAMAS 2006*, pages 548–550. ACM, 2006. doi:10.1145/1160633.1160733.

30 Martin Lück. On the complexity of linear temporal logic with team semantics. *Theor. Comput. Sci.*, 837:1–25, 2020. doi:10.1016/j.tcs.2020.04.019.

31 Corto Mascle and Martin Zimmermann. The keys to decidable HyperLTL satisfiability: Small models or very simple formulas. In Maribel Fernández and Anca Muscholl, editors, *CSL 2020*, volume 152 of *LIPIcs*, pages 29:1–29:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CSL.2020.29.

32 Amir Pnueli. The temporal logic of programs. In *FOCS 1977*, pages 46–57. IEEE, October 1977. doi:10.1109/SFCS.1977.32.

33 Markus N. Rabe. *A temporal logic approach to information-flow control*. PhD thesis, Saarland University, 2016. URL: http://scidok.sulb.uni-saarland.de/volltexte/2016/6387/.

34 Gaëtan Regaud and Martin Zimmermann. The complexity of fragments of second-order HyperLTL, 2025. Under preparation.

35 Hartley Rogers. *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, MA, USA, 1987.

36 Ron van der Meyden and Nikolay V. Shilov. Model checking knowledge and time in systems with perfect recall (extended abstract). In C. Pandu Rangan, Venkatesh Raman, and Ramaswamy Ramanujam, editors, *FSTTCS 1999*, volume 1738 of *LNCS*, pages 432–445. Springer, 1999. doi:10.1007/3-540-46691-6_35.

On the Expansion of Monadic Second-Order Logic with Cantor-Bendixson Rank and Order Type Predicates

Thomas Colcombet
Université Paris Cité, CNRS, IRIF, France

Alexander Rabinovich
The Blavatnik School of Computer Science, Tel-Aviv University, Israel

Abstract

In this work, we consider two extensions of monadic second-order logic, and study in what cases the classical decidability results are preserved.

The first extension, MSO[CBrank$_\beta$], is MSO (over the signature of the binary tree) augmented with the extra ability to express that the subtree over a set X has Cantor-Bendixson rank β, for some fixed countable ordinal β. We show that this extension is decidable over the binary tree if and only if β is finite, which means that it is decidable if and only if it is equivalent in expressiveness to MSO.

The second extension, MSO[otp$_\alpha$], is MSO (over the signature of order) augmented with the extra ability to express that the suborder induced by a set X has order type α for some fixed countable ordinal α. We show that this extension is decidable over countable ordinals if and only if $\alpha < \omega^\omega$, which means that it is decidable if and only if it is equivalent in expressiveness to MSO.

The first result can be established as a consequence of the second. The second result relies on the undecidability results of the logic BMSO (itself relying on the undecidability of MSO+U) in the case of ω^β for β a limit ordinal, and on entirely new techniques when β is a successor ordinal. We also have some partial extensions of the second result to some uncountable cases.

2012 ACM Subject Classification Theory of computation → Tree languages; Theory of computation → Logic and verification

Keywords and phrases Logic, Algorithmic model theory, Monadic second-order logic, Ordinals, Binary tree

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.11

Funding *Alexander Rabinovich*: Visits to IRIF were supported by the Fondation Sciences Mathématiques de Paris (FSMP) and Université Paris Cité.

1 Introduction

This work studies extensions of monadic second-order logic for which the decidability status was not known before.

Monadic second-order logic (MSO) is the extension of first-order logic with set quantifiers. It plays a key role in the context of automata and verification. The central decidability results in this context are (1) the seminal paper of Büchi [10] that proves the decidability of MSO$(\mathbb{N}, <)$ using automata, (2) the breakthrough [24] where Rabin establishes that MSO is decidable on infinite trees of height ω, again using automata, and from which the decidability of MSO over countable chains can be deduced, and finally (3) the introduction by Shelah [31] of model-theoretic techniques for showing the decidability of the MSO-theory of countable linear orderings, traditionally referred to as the "composition method."

These historical contributions show the extremely strong decidability properties that enjoy MSO. Their extension beyond MSO and beyond linear orderings and trees has been and still is a strong motivation for new research in the field (see more in Section 1.3). The present work pursues this quest by considering two natural extensions of monadic second-order logic (MSO for short).

1.1 Contributions

Our two contributions, Theorems 1 and 2 share a similar form. We consider in both cases a natural notion of measure of the "complexity" of a set, namely, the Cantor-Bendixson rank in the theory of infinite trees for the first one, and the order type in the theory of well-founded linear orders for the second. We then study the decidability status of MSO extended with a construction of the form:

"the complexity of the set X is α"

for a unique fixed value of α. Our results show that either this new construction was already expressible in MSO, or decidability is lost.

First extension: Cantor-Bendixson rank of trees

We consider here an extension of the monadic theory of trees.

The Cantor-Bendixson rank of a tree is an ordinal that measures its branching complexity (see Section 3 for more on trees and the Cantor-Bendixson rank), and is undefined if the tree contains an induced full binary tree. A subset X of a tree is downward closed if whenever $v \in X$, then all the nodes on the path from the root of T to v are in X. Given an ordinal α, for X some downward closed subset of an infinite tree, let $\mathsf{CBrank}_\alpha(X)$ express that "the tree restricted to universe X has Cantor-Bendixson rank α". We denote by $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ monadic second-order logic extended with the new predicate $\mathsf{CBrank}_\alpha(-)$. We prove:

▶ **Theorem 1.** *For all countable ordinals α, the following properties are equivalent:*

- *the $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$-theory of the full binary tree is decidable,*
- *α is finite,*
- *CBrank_α is MSO-definable.*

It can be summarised as the impossibility to extend the main theorem of Rabin of decidability of MSO over the full binary tree with the ability to express the Cantor-Bendixson rank of trees.

Second extension: the order type of an ordinal

We consider here an extension of the monadic theory of linear orders.

Büchi [11] proved a kind of "a small model property": if an MSO-formula is satisfiable in any countable ordinal, then it is satisfiable in an ordinal $< \omega^\omega$. Hence, ω^ω is MSO-undefinable. On the other hand, for every ordinal $\alpha < \omega^\omega$ one can express "the order type of X is α." A natural question follows: is the extension of MSO by the ability to express "the order type of X is ω^ω" still decidable? We provide a negative answer to this question.

Given an ordinal α, we consider the predicate $\mathsf{otp}_\alpha(X)$ which holds if the order type of the set X is the ordinal α, i.e., if the linear order restricted to universe X is isomorphic to α. We denote by $\mathsf{MSO}[\mathsf{otp}_\alpha]$ the monadic second-order logic of order extended with the new predicate $\mathsf{otp}_\alpha(-)$. Our main result reads as follows.

▶ **Theorem 2.** *For all countable ordinals α (and more generally for all $\alpha \leq \omega_1^{\omega_1}$, where ω_1 is the first uncountavle ordinal), the following properties are equivalent:*

- *the* MSO$[\mathsf{otp}_\alpha]$*-theory of α is decidable,*
- $\alpha < \omega^\omega$,
- $\mathsf{otp}_\alpha(-)$ *is* MSO*-definable.*

Let us recall that the MSO-theory is known to be decidable for the class of ordinals smaller than ω_2, as well as separately for each ordinal smaller than ω_2 (see [12] for the countable case, [31] for ordinals up to ω_2, and [21] for showing that this question is independent of ZFC at ω_2, where ω_2 is the first ordinal of the cardinality greater than the cardinality of ω_1). It is clear that if α is an MSO-definable ordinal, then MSO$[\mathsf{otp}_\alpha]$ and MSO are effectively expressively equivalent, and thus MSO$[\mathsf{otp}_\alpha]$ is decidable. Our result shows that if α is not MSO-definable and smaller than or equals to $\omega_1^{\omega_1}$, then, the logic is strictly more expressive than MSO; however, decidability is lost.

Note that we do not rule out the possibility that there exists some uncountable ordinal α larger or equal to $\omega_1^{\omega_1}$ such that the MSO$[\mathsf{otp}_\alpha]$-theory of α is still decidable.

1.2 Overview of the proofs

For both theorems, the difficult part is to prove the undecidability of the theory.

The undecidability part of the first theorem, Theorem 1, is obtained from Theorem 2. The main observation is that when the lexicographically order over the leaves of a binary tree T of Cantor-Bendixson rank α, is an ordinal, then its order type is in the interval $[\omega^\alpha, \omega^{\alpha+1})$. This is the crux of the reduction of the undecidability.

The undecidability part of the second theorem, Theorem 2, amounts to prove the undecidablity of the MSO$[\mathsf{otp}_\alpha]$-theory for all non MSO-definable ordinals α up to $\omega_1^{\omega_1}$. In fact, it boils down to treat three different cases.

- In Theorem 24, we prove that the MSO$[\mathsf{otp}_{\omega^\beta}]$-theory of ω^β is undecidable for all countable limit ordinals β (or more generally for β of cofinality ω). This covers, in particular, the important case of ω^ω which is the first non MSO-definable ordinal. This proof relies on a simple reduction of the BMSO-theory of ω, which is known to be undecidable (see Theorem 9).
- In Theorem 25, we establish that the MSO$[\mathsf{otp}_{\omega^\beta}]$-theory of ω^β can be reduced to the MSO$[\mathsf{otp}_{\omega^{\beta+1}}]$-theory of $\omega^{\beta+1}$. This is the most interesting part. It involves using the construct $\mathsf{otp}_{\omega^{\beta+1}}$ over an input that has been decomposed into ω-blocks in such a way that in almost all the blocks it faithfully simulates $\mathsf{otp}_{\omega^\beta}$.
- The two above results treat the case of all ordinals of the form ω^β that are smaller than $\omega_1^{\omega_1}$ and are not MSO-definable. But some ordinals are not of the form ω^β, such as, for instance, $\omega^\omega + 1$. Extending the result to these cases requires some extra work, but can be achieved without difficulty.

Another argument in the proof is a characterization of MSO-definable ordinals. This is well known in the countable case: a countable ordinal α is MSO-definable if and only if it is smaller than ω^ω. For proving Theorem 2, there is also a need to characterize the MSO-definable ordinals smaller than ω_2. We provide such a result in Theorem 45. This statement is not deep but, as far as we know, had not been mentioned before.

1.3 Historical Background

It was known in the 50s from Robinson that the extensions of MSO with a plus function, MSO$(\mathbb{N}, +)$, or even the doubling function MSO$(\mathbb{N}, <, x \mapsto 2x)$ were undecidable [27]. Elgot and Rabin studied in [17] the MSO theory of structures of the form $(\mathbb{N}, <, P)$, where P is

some unary predicate. They give a sufficient condition on P which ensures decidability of the MSO theory of $(\mathbb{N}, <, P)$. In particular, it holds when P denotes the set of factorials, or the set of powers of some integer. The frontier between decidability and undecidability of related theories was explored in numerous later papers [14, 18, 30, 29, 26, 25, 33, 34, 1].

The Büchi decidability theorem result (and the automata method) was extended to the MSO theory of any countable ordinal [11], to ω_1 - the first uncountable ordinal, and to any ordinal less than ω_2 - the first ordinal of the cardinality greater than ω_1, [13]. Gurevich, Magidor and Shelah [21] proved that the decidability status of the MSO theory of ω_2 depends on set-theoretical assumptions. What can be said about MSO theories for linear orderings beyond ordinals? Using automata, Rabin [24] proved decidability of the MSO theory of the binary tree, from which he deduces decidability of the MSO theory of $\mathbb{Q}$, which in turn implies decidability of the MSO theory of the class of countable linear orderings. Shelah [31] improved model-theoretical techniques yielding new decidability proofs over linear orderings, and proved that the MSO theory of the real line $(\mathbb{R}, <)$ is undecidable. The frontier between decidable and undecidable cases was specified in later papers by Gurevich and Shelah [20, 22, 23]); we refer the reader to the survey [19].

A logic that was much studied in recent years, introduced in [4], is MSO+U. The logic MSO+U extends MSO with a new quantifier-like construct $\mathsf{U}X.\varphi(X)$ expressing that there are sets of arbitrary large cardinality for which $\varphi(X)$ holds. Some non-trivial fragments of MSO+U are known to be decidable over ω (if this new construct is not allowed to appear negatively inside iteslf). [5]. It took more than ten years before it was shown undecidable over ω [8]. Works concerned with weak variants of this logic have also been pursued, yielding decidability results, such as in [9], but these cannot be considered as syntactic extensions of MSO.

The logic BMSO is a logic expressing properties of infinite sequences of numbers. It was designed in order to retain the quantitative aspects of MSO+U, while removing the ability to measure the cardinality of sets [2]. It turns out that its theory is intereducible to the one of MSO+U (see [2]), and thus it is also undecidable by [8]. The decidability of some important fragments of this logic still remain open. The undecidability result concerning MSO+U has been extended in [6], and then eventually, the existence of (almost) any significative extension of MSO that would retain decidability over ω has been ruled out in [7]: as soon as any non-regular property is expressible (with some mild closure assumption), then theory is undecidable. All these undecidability results boil down to reductions to the work [8].

The logic *cost-MSO* is another logic inspired by MSO+U, this time exhibiting the ability to express bound properties on the cardinality of sets, but removing the ability to quantify express asymptotic properties on these quantities. This logic is known to be decidable over finite words [15] and trees [16] over infinite words and partially over infinite trees [3]. The decidability status of cost-MSO over the full binary tree is a difficult open problem in the area.

1.4 Structure of the paper

Some definitions and results concerning MSO and BMSO are recalled in Section 2. Section 3 establishes our result over infinite trees, namely Theorem 1. Section 4 presents the proof of Theorem 24 stating that the $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$-theory of ω^β is undecidable for all countable limit ordinals β (or more generally β of cofinality ω). In Section 5, we prove that the $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$-theory of ω^β can be reduced to the $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$ of $\omega^{\beta+1}$ (Theorem 25). In Section 6, we combine the results of the two previous sections for proving our main result, Theorem 2, for all countable ordinals. Section 7 concludes the paper.

For space considerations all the results beyond the countable case, i.e., for ordinals up to $\omega_1^{\omega_1}$, do not appear in the main body of the submission, and are found in Appendix B.

2 Preliminaries

In this section we recall standard definitions and notions about ordinals (Section 2.1), monadic second-order logic (Section 2.2) and definability (Section 2.3). In Section 2.4, we introduce BMSO, and the undecidability Theorem 9 for it, which is crucial in our proof.

2.1 Ordinals

We shall use the standard terminology over ordinals. Here, an *ordinal* is seen, up to isomorphism, as a set equipped with a well-founded total order $<$. We use the classical notations over ordinals (order, sum, product, exponentiation). A *limit ordinal* is a non-empty ordinal that has no maximal element. A *successor ordinal* is an ordinal that has a maximum.

Given a subset X of an ordinal α, we denote $\alpha|_X$ its restriction to X. The *order type* of X is the ordinal $\alpha|_X$. We shall denote $[x, y)$ the set of $\{z \in \alpha \colon x \leqslant z < y\}$, and $[x, \infty)$ for the set $\{z \in \alpha \colon x \leqslant z\}$. An interval of the form $[x, \infty)$ is called a *final non-empty segment*. Given an ordinal α, a set X is *cofinal* (in α) if for all $x \in \alpha$, there is $y \in X$ with $y \geqslant x$. The *cofinality of* α is the least order type of X for X cofinal subset of α. Let us recall that all countable limit ordinals have cofinality ω.

2.2 Monadic second-order logic of order

We use standard notations and terminology about monadic second-order logic of order.

The *monadic second-order logic of order* (or *monadic logic of order* or simply *monadic logic*, abbreviated as MSO) is the extension of first-order logic over the signature $\{<\}$, where $<$ is a binary relational symbol interpreted as a total order, with (a) *monadic (second-order) variables* interpreted as subsets of the universe (usually denoted by uppercase letters X, Y, Z), (b) existential and universal quantifiers over for monadic variables $\exists X.\psi$ and $\forall X.\psi$, and (c) atomic formulas $y \in X$ expressing that y belongs to X.

In this paper, we shall always interpret this logic over partial orders or ordinals. Given a structure $\mathcal{M} := (M, <)$, which is a model over the signature $\{<\}$, a monadic formula $\varphi(X_1, \dots, X_k, y_1, \dots, y_\ell)$, sets $U_1, \dots, U_k \subseteq M$, and elements $v_1, \dots, v_\ell \in M$, we denote the fact that the formula holds on the model $\mathcal{M}$ with valuations U_i for X_i, and v_j for y_j as

$$\mathcal{M} \models \varphi(U_1, \dots, U_k, v_1, \dots, v_\ell) \ .$$

We shall use the classical technique of *relativization* of formulas. The next lemma is obtained easily by a syntactic transformation of the formula.

▶ **Lemma 3** (Relativization). *Let $\varphi(Y_1, \dots, Y_l)$ be a formula, U a variable not appearing in φ. We can compute a formula $\varphi^U(Y_1, \dots, Y_l, U)$ such that for every structure $\mathcal{M}$ and every non-empty $D \subseteq M$ and every l-tuple $\bar{P}$ of subsets of D:*

$$\mathcal{M} \models \varphi^U(\bar{P}, D) \text{ if and only if } \mathcal{M}_{\restriction D} \models \varphi(\bar{P}),$$

where $\mathcal{M}|_D$ is the substructure of $\mathcal{M}$ over D.

When this is the case, we say that φ holds in $(\mathcal{M}, \bar{P})$ *relativized* to D.

To ease the notation, we shall use some shorthands such as overlined variables $\bar{X}, \bar{Y}$ to denote tuples of monadic variables. We allow ourselves to write $X \subseteq Y$ to denote the inclusion of sets, and we use more generally any abbreviation if it is clear that it can be translated into MSO syntax. For instance, we shall use formulas such as ("X is cofinal" $\wedge$ "X has order type ω").

The main results concerning the MSO-theory of ordinals are the following:

▶ **Theorem 4** ([12]). *The MSO-theory of the class of countable ordinals is decidable. The MSO-theory of every countable ordinal is decidable.*

▶ **Theorem 5** ([31]). *The MSO-theory of the class of ordinals smaller than ω_2 is decidable. The MSO-theory of every ordinal smaller than ω_2 is decidable.*

▶ **Theorem 6** ([21]). *The MSO-theory of ω_2 is independent of ZFC.*

Given an ordinal β, we consider the extension of monadic logic in which the extra atomic formula $\mathsf{otp}_\beta(X)$ for X a monadic variable can be used, and is interpreted in an ordinal α as "the order type of X is β". It is denoted MSO$[\mathsf{otp}_\beta]$.

2.3 Definability

We say that a sentence ψ *defines the ordinal* α if α is the unique ordinal such that $\alpha \models \psi$. An ordinal α is *definable in logic* Ł, or simply (Ł*-definable*) if there is a sentence of Ł that defines α. The following important lemma is well known.

▶ **Lemma 7.** *For all countable ordinals α, α is MSO-definable if and only if $\alpha < \omega^\omega$.*

Definability of ordinals below ω^ω is fairly straightforward, by an inductive definition. The undefinability above ω^ω, follows from the small model property for MSO over the countable ordinals.

We say that a formula $\psi(x)$ *defines α inside the ordinal β* if there is a unique $b \in \beta$ such that $\beta \models \psi(b)$ and the $[0, b)$ has order type α. For a logic Ł, *α is Ł-definable inside β* if there is a formula $\psi(x) \in$ Ł that defines α in β.

It is clear that if α is MSO or MSO$[\mathsf{otp}_\gamma]$-definable by some sentence, then, by relativization of the sentence to $[0, b)$ it is definable inside β with the same logic for every $\beta > \alpha$. However, the other direction fails, as witnessed by the next lemma, to be put in contrast with Theorem 7.

▶ **Lemma 8.** *ω^ω is MSO-definable inside $\beta := \omega^\omega + \gamma$ for every $0 < \gamma < \omega^\omega$.*

Proof. By Theorem 7, there exists an MSO-sentence ψ_γ that defines γ. The formula that says that x is the minimal element such that ψ_γ holds when relativized to $[x, \infty)$. ◀

2.4 BMSO

The BMSO-logic is an extension of MSO that can express the existence of a bound on numerical quantity. Formally, the syntax of BMSO is the same as the one of MSO, extended with a new construct $B(X)$, for X a monadic variable. These formulas are interpreted on *ω-sequences of natural numbers*, that we see as the ordinal ω extended with a map f from ω to $\mathbb{N}$. The construct $B(X)$ is interpreted as "there exists $n \in \mathbb{N}$ such that $f(x) \leqslant n$ for all $x \in X$".

For instance, the formula $u \models \forall X.B(X)$ for an ω-sequence of natural numbers u if and only if u is bounded. The formula $\forall X.(\forall x \in X.\exists y \in X.(x < y)) \to \neg B(X)$ expresses the non-existence of an infinite set which is bounded: in other words, it expresses that f tends to infinity.

▶ **Theorem 9** (consequence of [2, 8]). *Satisfiability of* BMSO *is undecidable over* ω*-sequences of natural numbers.*

Proof. Theorem 2 in [2] states that BMSO is equivalent to another logic, AMSO. Theorem 13 states that the satisfiability of AMSO is equivalent to the one of MSO+U. Finally, Theorem 1.1 in [8] establishes the undecidability of MSO+U. ◀

3 The tree case

In this section we derive Theorem 1 from Theorem 2. This section is organized as follows. First, we recall some standard definitions and results about trees. Then, we recall a definition of Cantor-Bendixson rank of trees and state some well-known facts. Finally, we prove Theorem 1.

3.1 Trees

A *tree* $(T, <)$ is a structure over a signature with a unique a binary relation $<$ such that (1) there is a minimal element (called the *root*), and (2) for every $b \in T$ the set $\{a \mid a < b\}$ is finite and linearly ordered by $<$. Elements of the tree are called *nodes*. A node u is *parent* of a node v (and v is a *child* of u) if $u < v$ and there is no w such that $u < w < v$. A tree is *binary* if every node has at most two children. Nodes u and v are *incomparable* if neither $u \leqslant v$ nor $v \leqslant u$. An *antichain* is a subset of a tree such that all pairs of distinct nodes are incomparable. For a subset A of a tree we denote by $A{\downarrow}$ the downward closure of A, i.e., the set $\{b \mid \exists a \in A(b < a)\}$. For a node u, denote $T^{u\uparrow}$ the tree T restricted to nodes that are above or equal to u. $T^{u\uparrow}$ is itself a tree, and is called the *subtree at* u. Note that if T is a binary tree, then $T^{u\uparrow}$ also is. A tree is *regular* if it has finitely-many subtrees up to isomorphism.

The *full binary tree* is a tree for which (a) there is a partition of the children into *left* and *right children* and (b) all nodes have exactly two children, one being left, and the other right. The full binary tree is considered as a structure for the signature $\{<, \mathit{Left}(), \mathit{Right}()\}$. The *standard representation* of the full binary tree has as the domain the finite strings over $\{L, R\}$; the relation $<$ is interpreted as the prefix relation, and a node is left (respectively, right) if its last letter is L (respectively, R).

The major decidability result is *Rabin's theorem* [24]:

▶ **Theorem 10.** *The* MSO*-theory of the full binary tree is decidable.*

We shall also use the so-called *Rabin's Basis Theorem* [24]:

▶ **Theorem 11.** *If an* MSO*-sentence has a [binary] tree model, it has a [binary] regular tree model.*

We use $<_{\text{lex}}$ for the lexicographical (linear) order on the nodes of the full binary tree. It is definable in the standard way. Rabin proved [24] that there is a definable antichain Q such that $(Q, <_{\text{lex}})$ is order isomorphic to the rationals. As a consequence, he obtained that the MSO-theory of the rationals is decidable. Moreover, since every countable ordinal is embedable into the rationals, and "$(A, <)$ is an ordinal" is MSO-definable, he derived that the MSO-theory of the class of countable ordinals is decidable.

We will use the fact that $<_{\text{lex}}$ is MSO-definable, "A is an antichain" and "$A{\downarrow}$" are MSO-definable.

3.2 Cantor-Bendixson rank

We are going to define the Cantor-Bendixson rank (or CB rank) of a binary tree. There are several equivalent definitions. The only properties of Cantor-Bendixson rank that we need are stated in Theorem 16 and Theorem 17. The reader might skip the definition and use these lemmas as black-boxes.

▶ **Definition 12** (Sum of trees). *Let $T_i = (|T_i|, <_i)$ for $i \in \{0,1\}$ be trees over disjoint universes. their sum is the tree $T_0 + T_1$ with universe $|T_0| \cup |T_1|$ and its order relation defined as $n_1 \leqslant n_2$ if there is $i \in \{0,1\}$ such that $n_1, n_2 \in |T_i|$ and $n_1 \leqslant_i n_2$, or n_1 is the root of T_0.*

The ω-sum of trees is defined as follows.

▶ **Definition 13** (ω-sum of trees). *Let $T_i = (|T_i|, <_i)$ for $i \in \omega$ be trees over disjoint universes. We define the tree*

$$\sum_{i\in\omega} T_i$$

as having universe $\bigcup_{i\in\omega} |T_i|$ and its order relation is defined as $n_1 \leqslant n_2$ if there is i such that $n_1, n_2 \in |T_i|$ and $n_1 \leqslant_i n_2$ or n_1 is the root of T_i and $n_2 \in |T_j|$ for $j \geqslant i$.

If the disjointness assumption does not hold, we replace T_i by disjoint isomorphic copies and proceed as above.

In order to simplify notations, we will consider only finitely branching trees. The sets of trees of Cantor-Bendixson rank $\leqslant \alpha$ can be defined by transfinite induction.

▶ **Definition 14** (*Cantor-Bendixson rank* of a tree). *Define two families of trees CBrank_α and CBrank^+_α, where α is a countable ordinal.*

1. CBrank_0 *contains only one element trees.*
2. CBrank^+_α *is the closure of $\cup_{\beta\leqslant\alpha}\mathsf{CBrank}_\beta$ under $+$, i.e. $T \in \mathsf{CBrank}^+_\alpha$ if $T = T_0 + T_1 + \cdots + T_k$ for $T_i \in \cup_{\beta\leqslant\alpha}\mathsf{CBrank}_\beta$.*
3. $\mathsf{CBrank}_\alpha := \sum_{i\in\omega} T_i$, *where $T_i \in \mathsf{CBrank}^+_\beta$ for $\beta < \alpha$.*

If there is no α such that $T \in \mathsf{CBrank}^+_\alpha$, then the Cantor-Bendixson rank is undefined; otherwise we set $\mathsf{CBrank}(T)$ of T to be $\inf\{\alpha \mid T \in \mathsf{CBrank}^+_\alpha\}$, and the tree is called tame.

It is well known that a tree T has a CB rank if there is no embedding of the full binary tree in T, equivalently, T has only countably many branches. We are not going to use this fact.

▶ **Example 15.** Let S_1 be the subtree of the full binary tree T_2 over R^*, S_2 the subtree of T_2 over R^*L^*. Let S_{2i} (respectively, S_{2i+1}) be the subtree of T_2 over $(R^*L^*)^i$ (respectively, over and $(R^*L^*)^iR^*$). Then $\mathsf{CBrank}(S_i) = i$ for $i \geqslant 1$. Let S'_1 be the subtree of T_2 over R^*L; then $\mathsf{CBrank}(S'_1) = 1$.

The following lemma is well-known and is easily proved by induction.

▶ **Lemma 16.** *For every $n \in \mathbb{N}$, the set of binary trees of Cantor-Bendixson rank n is definable, i.e., there is an MSO sentence ϕ_n such that $(T, <) \models \phi_n$ if $(T, <)$ is a binary tree of Cantor-Bendixson rank n.*

The next lemma is proved in the Appendix.

▶ **Lemma 17.** *Let X be an antichain in the full binary tree such that $(X, <_{\text{lex}})$ is isomorphic to an ordinal. Then, $(X\downarrow, <)$ has Cantor-Bendixson rank α if and only if the order type of $(X, <_{\text{lex}})$ belongs to $[\omega^\alpha, \omega^{\alpha+1})$.*

3.3 Theorem 2 implies Theorem 1

Now, relying on Lemma 17, we can express in $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ that the order type of $(X, <_{\text{lex}})$ is ω^α for an antichain X of the full binary tree. The conjunction $\varphi_{\omega^\alpha}(X)$ of (1)-(5) below expresses this property.

1. X is an antichain:

$$\forall y \in X \forall x \in X (y \leqslant x) \to (x = y)$$

2. $(X, <_{\text{lex}})$ is isomorphic to an ordinal, i.e., every non-empty subset has a minimum element:

$$\forall Y \subseteq X \big((Y \neq \emptyset) \to \exists y \in Y (\forall z \in Y (y \leqslant_{\text{lex}} z))\big)$$

3. The downward closure of X has the CB rank α:

$$\mathsf{CBrank}_\alpha(X\downarrow), \text{ and}$$

4. For every final non-empty segment Y of $(X, <_{\text{lex}})$ the CB rank of the downward closure of Y is α:

$$\forall y \in X \big(\mathsf{CBrank}_\alpha(\{z \in X \mid z \geqslant y\}\downarrow)\big).$$

5. For no proper prefix $(Y, <_{\text{lex}})$ of $(X, <_{\text{lex}})$ the CB rank of the downward closure of Y is α.

(1)-(5) are not $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ formulas; however, they can be easily translated into (less readable) $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ formulas.

Next, let ψ be an $\mathsf{MSO}[\mathsf{otp}_{\omega^\alpha}]$ sentence. Let ψ^X be the relativization of ψ to a fresh variable X. Let $\Psi^X \in \mathsf{MSO}[\mathsf{CBrank}_\alpha]$ be obtained from ψ^X when all the occurrences of $\mathsf{otp}_{\omega^\alpha}(Y)$ are replaced by $\varphi_{\omega^\alpha}(Y)$. Finally, let Ψ be $\exists X(\varphi_{\omega^\alpha}(X) \wedge \Psi^X)$. Then, $\omega^\alpha \models \psi$ if and only if Ψ holds in the full binary tree. Hence, we have:

▶ **Lemma 18.** *There is an algorithm that for every* $\mathsf{MSO}[\mathsf{otp}_{\omega^\alpha}]$ *sentence* ψ *constructs an* $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ *sentence* Ψ *such that* $\omega^\alpha \models \psi$ *if and only if* Ψ *holds in the full binary tree.*

Moreover, in Theorem 18, the algorithm treats α symbolically, i.e., if φ is obtained from ψ when α is replaced by β, then the corresponding translation of φ replaces in Ψ everywhere α by β.

As a corollary of Theorem 18, Theorem 7 and Theorem 2, we obtain:

▶ **Corollary 19.** $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$ *is undecidable for every* $\alpha \geqslant \omega$.

And as a corollary of Rabin's theorem, Theorem 10, we obtain:

▶ **Corollary 20.** *For all* $\alpha \geqslant \omega$, *the property "the CB rank of a binary tree is* α*" is not* MSO*-definable.*

Theorem 1 follows from Theorem 10, Theorem 16, Theorem 19 and Theorem 20.

4 The ω^β case for β an ordinal of cofinality ω

The goal of this section is to establish Theorem 24, stating that the MSO[otp_α]-theory of α is undecidable for α of the form ω^β where β is a countable limit ordinal[1]. This covers in particular the case of ω^ω which is the first non-definable ordinal. Theorem 24 is obtained by reduction from the undecidability of BMSO (Theorem 9).

For the rest of the section, we fix an ordinal β which is limit of an ω-sequence $\widehat{\beta} := 0 = \beta_0 < \beta_1 < \cdots$. This is possible for all countable limit ordinals, and more generally for ordinals that have cofinality ω.

Given an ordinal $0 < \alpha < \omega^\beta$, it is said to be of *$\widehat{\beta}$-rank* k if $\omega^{\beta_k} \leqslant \alpha < \omega^{\beta_{k+1}}$. Let us denote $\widehat{\beta}\text{–rank}(\alpha)$ the $\widehat{\beta}$-rank of α. Recall that $\alpha|_X$ is the substructure of α over X. If X is some subset of an ordinal α, we also denote $\widehat{\beta}\text{–rank}(X)$ for the $\widehat{\beta}\text{–rank}(\alpha|_X)$.

The key argument used in this section is that we can relate the order type of some infinite sums of ordinals to the boundedness of sequences of $\widehat{\beta}$-ranks. This is formalized in the following lemma.

▶ **Lemma 21.** *Given an ω-sequence $(\alpha_i)_{i\in\omega}$ of ordinals smaller than ω^β, then the following properties are equivalent:*

- $\sum_{i\in\omega} \alpha_i = \omega^\beta$,
- *The ω-sequence of natural numbers defined for all $i \in \omega$ as $u_i := \widehat{\beta}\text{–rank}(\alpha_i)$ is unbounded.*

Proof. Assume the ranks of the α_i are unbounded, then there exists an increasing ω-sequence $0 = i_0 < i_1 < \ldots$ such that $j \leqslant \widehat{\beta}\text{–rank}(\alpha_{i_j})$ for all j. As a consequence, we have $\omega^{\beta_j} \leqslant \alpha_{i_j} \leqslant \sum_{i=i_j}^{i_{j+1}-1} \alpha_i$ for all j. We obtain

$$\omega^\beta = \sum_{j\in\omega} \omega^{\beta_j} \leqslant \sum_{j\in\omega} \sum_{i=i_j}^{i_{j+1}-1} \alpha_i = \sum_{i\in\omega} \alpha_i \ .$$

Conversely, assume the $\widehat{\beta}$-ranks of the α_i's would be bounded by some N; this means that $\alpha_i \leq \omega^{\beta_{N+1}}$ for all i. We get $\sum_{i\in\omega} \alpha_i \leqslant \omega^{\beta_{N+1}} \times \omega < \omega^\beta$. ◀

Let $S \subseteq \omega^\beta$ be of order type ω. This means that there exists an increasing ω-sequence $s_0 < s_1 < \cdots$ with $S = \{s_i : i \in \omega\}$. We abbreviate it as $S = \{s_0 < s_1 < \cdots\}$. Given a set $S = \{s_0 < s_1 < \cdots\} \subseteq \omega^\beta$, it is said to *encode the sequence* $u \in \mathbb{N}^\omega$ defined as $u(i) = \widehat{\beta}\text{–rank}([s_i, s_{i+1}))$ for all $i \in \omega$. Finally, given a set $X \subseteq \omega$, the *S-code of X*, written $\overline{X}^S$, is defined as

$$\overline{X}^S := \bigcup_{i\in X} [s_i, s_{i+1}) \ .$$

▶ **Fact 22.** *The key facts concerning these definitions are the following:*

1. *All ω-sequences of natural numbers $u \in \mathbb{N}^\omega$ are encoded by some ω-sequence $S = \{s_0 < s_1 < \cdots\}$. Take, for instance, $s_i = \omega^{\beta_{u(0)}} + \omega^{\beta_{u(1)}} + \cdots + \omega^{\beta_{u(i-1)}}$ for all $i \in \omega$.*
2. *Conversely, every $S \subseteq \omega^\beta$ of order type ω encodes a sequence $u \in \mathbb{N}^\omega$; namely, the one in which $u(i)$ is the β-rank of $[s_i, s_{i+1})$ for all $i \in \omega$.*
3. *For sets S and X, "S is of order type ω", and "X is an S-coded set", are properties definable in first-order logic (X and S are seen as unary predicates).*
4. *If S encodes u, then $u \models \neg B(X)$ if and only if $\omega^\beta \models \mathsf{otp}_{\omega^\beta}(\overline{X}^S)$. This is a direct consequence of Theorem 21 applied to the sequence of α_i, where $X = \{x_0 < x_1 < \cdots\}$ and α_i is the order type of $[s_{x_i}, s_{x_i+1})$.*

[1] This works in the more general case of β being a limit ordinal of cofinality ω.

▶ **Lemma 23.** *Given a formula $\varphi(X_1, \dots, X_k, x_1, \dots, x_l)$ of* BMSO*, there exists effectively a formula $\varphi^*(S, X_1, \dots X_k, x_1, \dots, x_l)$ of* $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$ *such that whenever S encodes u, $A_i \subseteq \omega$, and $a_j \in \omega$,*

$$u \models \varphi(A_1, \dots, A_k, a_1, \dots, a_\ell)$$

if and only if

$$\omega^\beta \models \varphi^*(S, \overline{A_1}^S, \dots, \overline{A_k}^S, s_{a_1}, \dots, s_{a_\ell}) \ .$$

Proof. The translation is defined by structural induction, as in the following table:

$$\begin{aligned}
(x < y)^* &:= x < y \ , & (x \in X)^* &:= x \in X, \\
(B(X))^* &:= \neg \mathsf{otp}_{\omega^\beta}(X) \ , & (\varphi \wedge \psi)^* &:= \varphi^* \wedge \psi^* \ , \\
(\forall x\, \psi)^* &:= \forall x\, (x \in S \to \psi^*) \ , & (\neg\varphi)^* &:= \neg\varphi^* \ , \\
(\forall X \psi)^* &:= \forall X (\text{“}X \text{ is an } S\text{-coded set”} \to \psi^*) \ .
\end{aligned}$$

The conclusion of the lemma is obtained along the same induction, relying on Theorem 22. ◀

▶ **Corollary 24.** *For all countable limit ordinals β (and more generally for all ordinals of cofinality ω), the* $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$*-theory of ω^β is undecidable.*

Proof. Consider a BMSO sentence φ, and, using the notations in Theorem 23, and the above facts, construct the sentence

$$\psi \quad := \quad \exists S \ (\text{“}S \text{ has order type } \omega\text{”} \quad \wedge \quad \varphi^*(S)) \ .$$

Then, φ has a sequence $u \in \mathbb{N}^\omega$ which models it if and only if ω^β models ψ. Indeed, if $u \models \varphi$, one can take some $S \subseteq \omega^\beta$ encoding u. It is of order type ω by definition, and by Theorem 23, $\omega^\omega \models \varphi^*(S)$. Conversely, if some S is the witness that $\omega^\omega \models \psi$, then S is of order type ω. Thus S encodes some sequence $u \in \mathbb{N}^\omega$, and since $\omega^\beta \models \varphi^*(S)$, by Theorem 23, $u \models \varphi$. In combination with Theorem 9, we get that the $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$-theory of ω^β is undecidable. ◀

5 Reduction of $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$ to $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$

We have shown in the previous section the undecidability of $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$ for β an ordinal of cofinality ω. In this section, we show that if $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$ is decidable for some β, then the same goes for $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$. This case requires more work.

We aim at proving the following:

▶ **Lemma 25.** *For all ordinals β, the* $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$*-theory of ω^β is reducible to the the* $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$*-theory of $\omega^{\beta+1}$.*

In other words, given an $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$-sentence φ, our goal is to construct an $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$-sentence ψ such that

$$\omega^\beta \models \varphi \qquad \text{if and only if} \qquad \omega^{\beta+1} \models \psi \ .$$

The principle of this construction is that formula ψ will guess a decomposition of $\omega^{\beta+1}$ into ω intervals, such that almost all of them have order type ω^β. The intuition is then that formula ψ will “simulate” φ independently in each of these blocks. The construction is done in such a way that this “simulation” is “faithful” on almost all the blocks. There, we say that a unary property P holds for *almost all* elements of a set D if P holds on all, but finitely many elements of D.

The first key ingredient for achieving this is Theorem 27. It provides an $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$-formula that allows to chop $\omega^{\beta+1}$ into ω pieces while guaranteeing that almost all of them are of order type ω^β.

We first state some elementary facts about ordinals that will prove useful in the construction.

▶ **Fact 26.** *The following standard facts hold:*

1. *The order type of every final non-empty segment of ω^β has order type ω^β.*
2. *Let $(\alpha_i)_{i\in\omega}$ be an ω-sequences of ordinals smaller than $\omega^{\beta+1}$. Then $\sum_{i\in\omega}\alpha_i = \omega^{\beta+1}$ if and only if $\alpha_j \geqslant \omega^\beta$ for infinitely many j.*
3. *For all ordinals $\alpha < \omega^{\beta+1}$, either $\alpha = \omega^\beta \times k$ for some natural $k > 0$, or α has some final non-empty segment of order type smaller than ω^β.*

▶ **Lemma 27.** *There is effectively an $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$ formula $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(X)$ such that for all $S = \{s_0 < s_1 < \cdots\}$, $\omega^{\beta+1} \models \mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ if and only if the order type of $[s_i, s_{i+1})$ is ω^β for almost all $i \in \omega$.*

Proof. We proceed in three steps, in which we successively describe formulas that approximate each time better the expected behavior of $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}$.

Step 1. The $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$-formula $\mathrm{Split}^{\infty}_{\omega^\beta\times *}(S)$ expressing that

- $\bigcup_{i\in\omega}[a_i, s_{i+1})$ has order type $\omega^{\beta+1}$ for all $(a_i)_{i\in\omega}$ such that $a_i \in [s_i, s_{i+1})$ for all $i \in \omega$,

which holds if and only if there exist infinitely many indices $i \in \omega$ such that $[s_i, s_{i+1})$ is of the order type $\omega^\beta \times k$ for some $k \geqslant 1$.

Indeed, assume that $[s_i, s_{i+1})$ is of the form $\omega^\beta \times k$ with $k \geqslant 1$ for infinitely many $i \in \omega$, then by the first item of Theorem 26, $[a_i, s_{i+1})$ has an order type of the form $\omega^\beta \times \ell$ with $\ell \geqslant 1$ for these i's. Consequently, by the second item of Theorem 26, $\bigcup_{i\in\omega}[a_i, s_{i+1})$ has order type $\omega^{\beta+1}$.

Conversely, assume that $[s_i, s_{i+1})$ is not of the form $\omega^\beta \times k$ with $k \geqslant 1$ for almost all $i \in \omega$. This means by the third item of Theorem 26, that for these i's, there exists $a_i \in [s_i, s_{i+1})$ such that the order type of $[a_i, s_i)$ is smaller than ω^β. This can be completed, by choosing arbitrary a_i in the finitely many other segments, into an ω-sequence of a_i's as in the formula. This time using the second item of Theorem 26, we get that $\bigcup_{i\in\omega}[a_i, s_{i+1})$ has an order type smaller than $\omega^{\beta+1}$.

Step 2. In a second step, we claim that the formula $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta\times *}$ that expresses that

- for all infinite sets $S' \subseteq S$, $\mathrm{Split}^{\infty}_{\omega^\beta\times *}(S')$ holds

if and only if $[s_i, s_{i+1})$ is of the form $\omega^\beta \times k$ with $k \geqslant 1$ for almost all $i \in \omega$.

Indeed, if $[s_i, s_{i+1})$ is of the form $\omega^\beta \times k$ with $k \geqslant 1$ for almost all $i \in \omega$, this also holds for every infinite subsequence, and as a consequence, $\mathrm{Split}^{\infty}_{\omega^\beta\times *}(S')$ for all infinite $S' \subseteq S$.

Conversely, assume that $[s_i, s_{i+1})$ is not of the form $\omega^\beta \times k$ with $k \geqslant 1$ for infinitely many $i \in \omega$. In this case, it is possible to extract a subsequence $S' = \{s'_0 < s'_1 < \cdots\} \subseteq S$ such that the order type of $[s'_i, s'_{i+1})$ is not of the form $\omega^\beta \times k$ with $k \geqslant 1$, for all $i \in \omega$. According to the previous step, $\mathrm{Split}^{\infty}_{\omega^\beta\times *}(S')$ does not hold for this choice of S', and hence $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta\times *}(S)$ does not either.

Step 3. Finally, we can define the formula $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ that expresses that

- $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta\times *}(S)$ holds, and
- for all $S' \supseteq S$, if $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta\times *}(S')$ holds then $S' \setminus S$ is not cofinal.

We have to show that it fulfils the conclusion of the lemma, i.e., that $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ holds if and only if the order type of $[s_i, s_{i+1})$ is ω^β for almost all $i \in \omega$.

Indeed, assume that $[s_i, s_{i+1})$ has order type ω^β for almost all $i \in \omega$, and consider some $S' = \{s'_0 < s'_1 < \cdots\} \supseteq S$. If $S' \setminus S$ is cofinal, then for such sufficiently large j, we would have that $s_i < s'_j < s_{i+1}$ for some i such that $[s_i, s_{i+1})$ has order type ω^β. But in this case, $[s_i, s'_j)$ has an order type smaller than ω^β and since $s'_{j-1} \geqslant s_i$, the interval $[s'_{j-1}, s'_j)$ also does. Overall, we have constructed infinitely many intervals $[s'_{j-1}, s'_j)$ of order type smaller than ω^β, and thus $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta \times *}(S')$ does not hold. This is a contradiction, and hence $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ is satisfied.

Conversely, assume by contradiction that $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ holds, and that $[s_i, s_{i+1})$ has an order type different than ω^β for infinitely many $i \in \omega$. Since $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta \times *}(S)$ holds, this means that $[s_i, s_{i+1})$ has order type $\omega^\beta \times k$ with $k > 1$ for infinitely many $i \in \omega$. But each such interval $[s_i, s_{i+1})$ can be decomposed into $[s_i, s'_i)$ and $[s'_i, s_{i+1})$, both of them of order type $\omega^\beta \times k$ for some $k \geqslant 1$. Thus, the set S' obtained by adding to S all such elements s'_i would make $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta \times *}(S')$ satisfied. Since $S' \setminus S$ is infinite, this contradicts the fact that $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ holds. ◀

From now on, we assume that $S = \{s_0 < s_1 < \cdots\}$ is such that $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$ holds. Let α_i be the order type of $[s_i, s_{i+1})$ be the corresponding sequence of ordinals. For the sake of simplicity, we shall not mention the first order variables in formulas in the rest of the proof, since these can be seen as a special case of monadic variables that would be interpreted as singletons (something definable).

▶ **Lemma 28.** *There is an algorithm which given an* $\mathsf{MSO}[\mathsf{otp}_{\omega^\beta}]$*-formula*

$$\varphi(X_1, \dots X_k),$$

constructs an $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta+1}}]$*-formula*

$$\varphi^*(S, F, X_1, \dots X_k)$$

such that for all $A_1, \dots, A_k \subseteq \omega^{\beta+1}$ *and all infinite* $F \subseteq \omega$*,*

$$\omega^{\beta+1} \models \varphi^*(S, \overline{F}^S, A_1, \dots, A_k)$$

if and only if

$$\omega^{\beta+1}|_{[s_i, s_i+1)} \models \varphi(A_1 \cap [s_i, s_{i+1}), \dots, A_k \cap [s_i, s_{i+1}))$$

for almost all $i \in F$*.*

Proof. We shall define the formula φ^* by structural induction on φ, and establish the conclusions of the lemma at the same time. The case of existential quantifier, the case of conjunction, and the case of MSO predicates are elementary. The crucial point is the negation.

For the sake of simplicity, we shall write $\omega^{\beta+1} \models^{S,F} \varphi(\bar{A})$ in order to express that $\omega^{\beta+1}|_{[s_i, s_i+1)} \models \varphi(A_1 \cap [s_i, s_{i+1}), \dots, A_k \cap [s_i, s_{i+1}))$ for almost all i such that $s_i \in F$.

Case of a conjunction, i.e., $\varphi(\bar{X}) = (\varphi_1(\bar{X}) \wedge \varphi_2(\bar{X}))$. We define

$$\varphi^*(S, F, \bar{X}) := \varphi_1^*(S, F, \bar{X}) \wedge \varphi_2^*(S, F, \bar{X}) \ .$$

The induction hypothesis holds: indeed, $\omega^\beta \models \varphi^*(S, F, \bar{A})$ if and only if $\omega^\beta \models \varphi_1^*(S, F, \bar{A})$ and $\omega^\beta \models \varphi_2^*(S, F, \bar{A})$, if and only if (by induction hypothesis) $\omega^{\beta+1} \models^{S,F} \varphi_1(\bar{A})$ and $\omega^{\beta+1} \models^{S,F} \varphi_2(\bar{A})$, if and only if $\omega^{\beta+1} \models^{S,F} \varphi_1(\bar{A}) \wedge \varphi_2(\bar{A})$, if and only if $\omega^{\beta+1} \models^{S,F} \varphi(\bar{A})$.

Case of an existential set quantifier, i.e., $\varphi(\bar{X}) = \exists Y.\varphi_1(\bar{X}, Y)$. We define

$$\varphi^*(S, F, \bar{X}) := \exists Y.\ \varphi_1^*(S, F, \bar{X}, Y)\ .$$

Correctness is also straighforward.

Case of an MSO-formula $\varphi(\bar{X})$, and in particular of the atomic formulas $x \in Y$ and $x < y$ (recall that in this case, we see first-order variable as singleton sets). In this case, we simply set $\varphi^*(S, \overline{F}^S, \bar{X})$ to express $\omega^{\beta+1} \models^{S,F} \varphi(\bar{X})$. This is easily definable using relativization and the fact that "almost all" can be expressed in MSO.

Case of an order type predicate, i.e., $\varphi(\bar{X}) := \mathsf{otp}_{\omega^\beta}(X_m)$. For simplicity, we shall treat the case of $\varphi(\bar{X}) := \neg\mathsf{otp}_{\omega^\beta}(X_m)$, and leave the question of removing the negation to the negation case below. We set:

$$\varphi^*(S, F, \bar{X}) := \neg\mathsf{otp}_{\omega^{\beta+1}}(X_m \cap \overline{F}^S)\ .$$

The correctness of this construction relies on the second item of Theorem 26. Indeed, "$\mathsf{otp}_{\omega^{\beta+1}}(A_m \cap \overline{F}^S)$ does not hold" means that $A_m \cap [s_i, s_{i+1})$ has order type smaller than ω^β for almost all $i \in F$, which is the same as $\omega^{\beta+1} \models^{S,F} \neg\mathsf{otp}_{\omega^\beta}(X_m)$.

Case of a negation, i.e., $\varphi(\bar{X}) = \neg\varphi_1(X_1, \dots, X_k)$. We set

$$\varphi^*(S, F, \bar{X}) := \forall F' \subseteq F.\ (\text{“}F' \text{ infinite”}) \to \neg\varphi_1^*(S, F', \bar{X})\ .$$

Let us prove that the induction hypothesis holds. Let us assume that $\omega^{\beta+1} \models^{S,F} \neg\varphi(\bar{A})$. This is equivalent to the fact that $\neg\varphi(\bar{A} \cap [s_i, s_{i+1}))$ holds for almost all i with $s_i \in F$. Hence, this is also true for almost all i such that $s_i \in F'$ when $F' \subseteq F$ is infinite. Thus $\omega^{\beta+1} \models \varphi^*(S, F, \bar{A})$. Conversely, assume that $\omega^{\beta+1} \models^{S,F} \neg\varphi(\bar{A})$ does not hold. This means that there are infinitely many i such that s_i in F and $\varphi(\bar{A} \cap [s_i, s_{i+1}))$ holds. Let us chose $F' \subseteq F$ infinite to contain these indices. Then, this F' is a witness that $\omega^{\beta+1} \models \varphi^*(S, F, \bar{A})$ does not hold either. ◀

As a consequence of Theorem 28 and of Theorem 27, we obtain:

▶ **Corollary 29.** *For every* MSO$[\mathsf{otp}_{\omega^\beta}]$*-sentence* φ*, there exists effectively an* MSO$[\mathsf{otp}_{\omega^{\beta+1}}]$*-sentence* ψ *such that*

$$\omega^\beta \models \varphi \qquad \textit{if and only if} \qquad \omega^{\beta+1} \models \psi\ .$$

Proof. Set ψ to be the formula

$$\exists S\ \text{“}S \text{ has order type } \omega\text{”} \wedge \text{“}S \text{ is cofinal”} \wedge \mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S) \wedge \varphi^*(S, S)\ ,$$

in which φ^* is the formula produced by Theorem 28.

First implication. Let us assume that $\omega^\beta \models \varphi$. We have to prove that $\omega^{\beta+1} \models \psi$. For this, let us choose S to be $\{s_i := \omega^\beta \times i \mid i < \omega\}$. This set S is of order type ω and cofinal in $\omega^{\beta+1}$. It is also such that $[s_i, s_{i+1})$ as order type ω^β for all i. Hence, by Theorem 27, $\omega^{\beta+1} \models \mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$. Since $\omega^\beta \models \varphi$, we get by Theorem 28 that $\omega^{\beta+1} \models \varphi^*(S, S)$, and hence $\omega^{\beta+1} \models \psi$.

Conversely, let us assume that $\omega^{\beta+1} \models \psi$. This means that there exists S of order type ω and cofinal that satisfies $\mathrm{Split}^{\mathrm{aa}}_{\omega^\beta}(S)$. Let $S = \{s_0 < s_1 < \dots\}$. By Theorem 27, this means that $[s_i, s_{i+1})$ has order type ω^β for almost all i. Furthermore, $\omega^{\beta+1} \models \varphi^*(S, S)$. Hence, by Theorem 28, this means that $\omega^{\beta+1}|_{[s_i, s_{i+1}} \models \varphi$ for almost all $i \in \omega$. This means that it holds for at least one $i < \omega$ such that $\omega^{\beta+1}|_{[s_i, s_{i+1})} = \omega^\beta$. Hence, $\omega^\beta \models \varphi$. ◀

6 Countable ordinals

By combining Theorems 24 and 25 from the previous sections, we have proved that the $\mathsf{MSO}[\mathsf{otp}_\alpha]$-theory of α is undecidable for all countable ordinals α of the form ω^β. The next step is to show it for all countable $\alpha \geqslant \omega^\omega$:

▶ **Lemma 30.** *For all countable ordinals $\alpha \geqslant \omega^\omega$, the $\mathsf{MSO}[\mathsf{otp}_\alpha]$-theory of α is undecidable.*

This part of the proof does not involve new interesting arguments. It is presented below for the completeness.

Recall some elementary facts about ordinal arithmetic. Cantor proved that every ordinal α can be uniquely expressed as a finite sum

$$\alpha = \omega^{\beta_n} + \cdots + \omega^{\beta_1} + \omega^{\beta_0} \ ,$$

for ordinals $\beta_n \geqslant \cdots \geqslant \beta_1 \geqslant \beta_0$. This is called the *Cantor normal form* of α.

▶ Notations. In order to avoid multi level subscripts, we will sometimes write $\mathsf{MSO}[\alpha]$ instead of $\mathsf{MSO}[\mathsf{otp}_\alpha]$

▶ **Lemma 31.** *Let $\alpha = \omega^{\beta_n} + \cdots + \omega^{\beta_1} + \omega^{\beta_0}$ where $\beta_n \geqslant \cdots \geqslant \beta_1 \geqslant \beta_0$. If the $\mathsf{MSO}[\alpha]$-theory of α is decidable, the $\mathsf{MSO}[\omega^{\beta_n}]$-theory of ω^{β_n} is also decidable.*

Proof. First note that there is an $\mathsf{MSO}[\alpha]$ formula $\psi_{=\omega^{\beta_n}}(x)$ such that $\alpha \models \psi_{=\omega^{\beta_n}}(b)$ if and only if $b = \omega^{\beta_n}$. Indeed, this formula says that x is the minimal element such that the set $\{y \ : \ y \geqslant x\}$ does not have the order type α.

Now, for every $\mathsf{MSO}[\omega^{\beta_n}]$ sentence φ, we can construct an $\mathsf{MSO}[\alpha]$ sentence φ^* such that $\omega^{\beta_n} \models \varphi$ if and only if $\alpha \models \varphi^*$. Indeed, it is sufficient to relativize the quantifiers to β_n, and this is possible according to the above remark, and replace every occurrence of the predicate $\mathsf{otp}_{\omega^{\beta_n}}(X)$ by $\mathsf{otp}_\alpha(X \cup \{z \mid z > \omega^{\beta_n}\})$. ◀

We are now ready to complete the proof of our theorem in the countable case.

Proof of Theorem 30. Let $\alpha \geqslant \omega^\omega$ be a countable ordinal. Its Cantor normal form $\beta_n \geqslant \cdots \geqslant \beta_1 \geqslant \beta_0$ is such that β_n is countable infinite. Hence, by Theorems 24 and 25, the $\mathsf{MSO}[\mathsf{otp}_{\omega^{\beta_n}}]$-theory of ω^{β_n} is undecidable. By Theorem 31, this implies that the $\mathsf{MSO}[\mathsf{otp}_\alpha]$-theory of α is undecidable. ◀

To sum up we have the following corollary:

▶ **Corollary 32.** *Let α be a countable ordinal. TFAE*

1. *$\alpha \geqslant \omega^\omega$.*
2. *α is not MSO definable.*
3. *The $\mathsf{MSO}[\alpha]$ theory of α is undecidable.*
4. *The $\mathsf{MSO}[\alpha]$ theory of any class of ordinals that contains an ordinal $\geqslant \alpha$ is undecidable.*

Proof. We have already proved the equivalence between (1)-(3). The implication (4)⇒(3) is immediate.

Let us proof that (3) ⇒(4). First note that there is an $\mathsf{MSO}[\alpha]$ formula $\Psi_\alpha(x)$ which defines α inside every $\beta > \alpha$, i.e., for every $\beta > \alpha$ there is a unique b such that $\beta \models \Psi_\alpha(b)$ and the substructure of β over the prefix $\{a \mid a < b\}$ is isomorphic to α. Now, using this formula, for every $\mathsf{MSO}[\alpha]$ sentence Φ it is easy to construct an $\mathsf{MSO}[\alpha]$ sentence Φ^α such that $\alpha \models \Phi$ if and only if $\beta \models \Phi^\alpha$ for every (equivalently some) $\beta > \alpha$. ◀

7 Conclusion

This paper belongs to the body of works that aims at extending the expressive power of monadic second-order logic, while retaining decidability results. More precisely, we have studied the question of decidability of monadic second-order logic of ordinals when extended with predicates about the order type of sets. Since the order type of some ordinals is non definable (in particular for all countable ordinals from ω^ω upward), such extensions of MSO are strictly more expressive than MSO. Our main result for ordinals, Theorem 2, shows that up to ordinal $\omega_1^{\omega_1}$, there is nothing to be gained: if extending MSO with an order type predicate is decidable, then this order type was already definable in plain MSO, and thus the obtained logic is expressively equivalent to MSO.

The proof techniques involve a reduction of non-decidability of the satisfiability BMSO, which itself relies on the deep result of undecidability of the satisfiability of MSO+U over ω. This has to be combined with extra involved arguments for catching all the ordinals $< \omega_1^{\omega_1}$. However, when reaching $\omega_1^{\omega_1}$ all these tools seems to become useless, and the main open question we are left with is the following:

▶ **Problem 33.** *Is it true that α is MSO-definable if and only if the $\mathsf{MSO}[\mathsf{otp}_\alpha]$-theory of α is decidable for all $\alpha < \omega_2$?*

Another problem is:

▶ **Problem 34.** *What is the degree of undecidability of the $\mathsf{MSO}[\mathsf{otp}_\alpha]$-theory of α?*

We provided a reduction from the satisfiability of BMSO to the satisfiability problem of $\mathsf{MSO}[\mathsf{otp}_\alpha]$. The satisfiability problem for BMSO is not in RE and not in Co-RE. We do not know whether there is a reduction in the other direction. Similar problems are about the degree of undecidability of the $\mathsf{MSO}[\mathsf{CBrank}_\alpha]$-theory of the full binary tree.

The *Hausdorff rank* (sometimes called *VD*-rank or *F*-rank) is naturally definable for linear orders [28]. In particular, a linear order $(L, <)$ has a Hausdorff rank if and only if it is scattered, equivalently if and only if there is no order preserving embedding of the rationals in $(L, <)$. Given an ordinal α, let $\mathsf{Hrank}_\alpha(X)$ express that X has Hausdorff rank α. We denote by $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$ the monadic second-order logic of order extended with the new predicate $\mathsf{Hrank}_\alpha(-)$.

It is well known that "a linear order has Hausdorff rank n" is definable for every $n \in \mathbb{N}$.

▶ **Theorem 35.** *For every countable ordinals $\alpha \geqslant \omega$, the $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$-theory of the class of countable linaer orders is undecidable.*

The proof is a reduction of Theorem 35 to Theorem 2, and it is based on the following observations:

1. An ordinal γ has Hausdorff rank $\leqslant \alpha$ if and only if $\gamma \leqslant \omega^\alpha$.
2. Hence, for an infinite ordinal γ: $OTP(\gamma) = \omega^\alpha$ can be expresses as $\mathsf{Hrank}(\gamma) = \alpha$ and $\mathsf{Hrank}(\{y \mid y \geqslant x\}) = \alpha$ for every $x < \gamma$.

Therefore, there is an $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$ formula $\varphi_\alpha(X)$ that expresses "X is a well-order and the order type of X is ω^α." Hence, by Theorem 2, we obtain that the $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$-theory of the linear orders of Hausdorff rank α as well as the $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$-theory of the class of countable linear orders is undecidable.

We can prove a stronger result:

▶ **Theorem 36.** *For every countable ordinals $\beta \geq \alpha \geq \omega$, and every countable linear order $(L, <)$ of Hausdorff rank β, the $\mathsf{MSO}[\mathsf{Hrank}_\alpha]$-theory of $(L, <)$ is undecidable.*

Our proof of Theorem 36 is direct. The proof techniques are similar to those of Theorem 2, however, we have not found a reduction of Theorem 36 to Theorem 2.

References

1. Valérie Berthé, Toghrul Karimov, Joris Nieuwveld, Joël Ouaknine, Mihir Vahanwala, and James Worrell. On the decidability of monadic second-order logic with arithmetic predicates. In Pawel Sobocinski, Ugo Dal Lago, and Javier Esparza, editors, *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2024, Tallinn, Estonia, July 8-11, 2024*, pages 11:1–11:14. ACM, 2024. doi:10.1145/3661814.3662119.
2. Achim Blumensath, Olivier Carton, and Thomas Colcombet. Asymptotic monadic second-order logic. 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 I*, volume 8634 of *Lecture Notes in Computer Science*, pages 87–98. Springer, 2014. doi:10.1007/978-3-662-44522-8_8.
3. Achim Blumensath, Thomas Colcombet, Denis Kuperberg, Pawel Parys, and Michael Vanden Boom. Two-way cost automata and cost logics over infinite trees. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 16:1–16:9. ACM, 2014. doi:10.1145/2603088.2603104.
4. Mikolaj Bojanczyk. A bounding quantifier. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *Computer Science Logic, 18th International Workshop, CSL 2004, 13th Annual Conference of the EACSL, Karpacz, Poland, September 20-24, 2004, Proceedings*, volume 3210 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004. doi:10.1007/978-3-540-30124-0_7.
5. Mikolaj Bojanczyk and Thomas Colcombet. Boundedness in languages of infinite words. *Log. Methods Comput. Sci.*, 13(4), 2017. doi:10.23638/LMCS-13(4:3)2017.
6. Mikolaj Bojanczyk, Laure Daviaud, Bruno Guillon, Vincent Penelle, and A. V. Sreejith. Undecidability of a weak version of MSO+U. *Log. Methods Comput. Sci.*, 16(1), 2020. doi:10.23638/LMCS-16(1:12)2020.
7. Mikolaj Bojanczyk, Edon Kelmendi, Rafal Stefanski, and Georg Zetzsche. Extensions of ω-regular languages. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 266–272. ACM, 2020. doi:10.1145/3373718.3394779.
8. Mikolaj Bojanczyk, Pawel Parys, and Szymon Torunczyk. The MSO+U theory of (n, <) is undecidable. In Nicolas Ollinger and Heribert Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*, pages 21:1–21:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.STACS.2016.21.
9. Mikolaj Bojanczyk and Szymon Torunczyk. Weak MSO+U over infinite trees. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012, February 29th - March 3rd, 2012, Paris, France*, volume 14 of *LIPIcs*, pages 648–660. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012. doi:10.4230/LIPIcs.STACS.2012.648.
10. J. R. Büchi. On a decision method in the restricted second-order arithmetic. In *Proc. Int. Congress Logic, Methodology and Philosophy of science, Berkeley 1960*, pages 1–11. Stanford University Press, 1962.
11. J. R. Büchi. Transfinite automata recursions and weak second order theory of ordinals. In *Proc. Int. Congress Logic, Methodology, and Philosophy of Science, Jerusalem 1964*, pages 2–23. HOLLAND, 1965.
12. J Richard Büchi and Dirk Siefkes. *Decidable Theories: Vol. 2: The Monadic Second Order Theory of All Countable Ordinals*, volume 328. Springer, 2006.
13. J.Richard Büchi and Charles Zaiontz. Deterministic automata and the monadic theory of ordinals ω_2. *Z. Math. Logik Grundlagen Math.*, 29:313–336, 1983.
14. O. Carton and W. Thomas. The monadic theory of morphic infinite words and generalizations. *Inform. Comput.*, 176:51–76, 2002.

15 Thomas Colcombet. Regular cost functions, part I: logic and algebra over words. *Log. Methods Comput. Sci.*, 9(3), 2013. doi:10.2168/LMCS-9(3:3)2013.

16 Thomas Colcombet and Christof Löding. Regular cost functions over finite trees. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, pages 70–79. IEEE Computer Society, 2010. doi:10.1109/LICS.2010.36.

17 Calvin C. Elgot and Michael O. Rabin. Decidability and undecidability of extensions of second (first) order theory of (generalized) successor. *J. Symb. Log.*, 31(2):169–181, 1966. doi:10.2307/2269808.

18 S. Fratani. The theory of successor extended with several predicates. *preprint*, 2009.

19 Y. Gurevich. Monadic second-order theories. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, pages 479–506. Springer-Verlag, Perspectives in Mathematical Logic, 1985.

20 Yuri Gurevich. Modest theory of short chains. i. *J. Symb. Log.*, 44(4):481–490, 1979. doi:10.2307/2273287.

21 Yuri Gurevich, Menachem Magidor, and Saharon Shelah. The monadic theory of ω_2. *J. Symb. Log*, 48(2):387–398, 1983.

22 Yuri Gurevich and Saharon Shelah. Modest theory of short chains. ii. *J. Symb. Log.*, 44(4):491–502, 1979. doi:10.2307/2273288.

23 Yuri Gurevich and Saharon Shelah. Interpreting second-order logic in the monadic theory of order. *J. Symb. Log.*, 48(3):816–828, 1983. doi:10.2307/2273475.

24 M.O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

25 A. Rabinovich. On decidability of monadic logic of order over the naturals extended by monadic predicates. *Inf. Comput*, 205(6):870–889, 2007. doi:10.1016/J.IC.2006.12.004.

26 A. Rabinovich and W. Thomas. Decidable theories of the ordering of natural numbers with unary predicates. In Zoltán Ésik, editor, *Computer Science Logic, 20th International Workshop, CSL 2006, 15th Annual Conference of the EACSL, Szeged, Hungary, September 25-29, 2006, Proceedings*, volume 4207 of *Lecture Notes in Computer Science*, pages 562–574. Springer, 2006. doi:10.1007/11874683_37.

27 R.M. Robinson. Restricted set-theoretical definitions in arithmetic. *Proc. Am. Math. Soc.*, 9:238–242, 1958.

28 J.G. Rosenstein. *Linear Orderings*. ISSN. Elsevier Science, 1982. URL: https://books.google.com.sg/books?id=y3YpdW-sbFsC.

29 A. L. Semenov. Decidability of monadic theories. In M. P. Chytil and V. Koubek, editors, *Proceedings of the 11th Symposium on Mathematical Foundations of Computer Science*, volume 176 of *LNCS*, pages 162–175, Praha, Czechoslovakia, September 1984. Springer. doi:10.1007/BFB0030296.

30 A. L. Semenov. Logical theories of one-place functions on the set of natural numbers. *Mathematics of the USSR - Izvestia*, 22:587–618, 1984.

31 S. Shelah. The monadic theory of order. *Annals of Mathematics*, 102:379–419, 1975.

32 Saharon Shelah. The monadic theory of order. *The Annals of Mathematics*, 102(3):379, November 1975. doi:10.2307/1971037.

33 D. Siefkes. Decidable extensions of monadic second order successor arithmetic. *Automatentheorie und Formale Sprachen, (Tagung, Math. Forschungsinst, Oberwolfach), 1969; (Bibliograph. Inst., Mannheim)*, pages 441–472, 1970.

34 W. Thomas. A note on undecidable extensions of monadic second order successor arithmetic. *Arch. Math. Logik Grundlagenforsch.*, 17:43–44, 1975. doi:10.1007/BF02280812.

A Proof of Lemma 17

Note that $T_0 + T_1$ is a binary tree only if T_0 has at most one child. Since we are dealing with binary trees where every child is either left or right (these trees are considered as structures for the signature $\{<, \mathit{Left}(), \mathit{Right}()\}$) we have further refine sum and ω-sum operations.

First we characterize when a tree T of rank α is isomorphic to the downward closure of an antichain X of the full binary tree such that $(X, <_{\text{lex}})$ is (isomorphic to) an ordinal. Then we state properties of these trees and finally, we prove Theorem 17.

▶ **Definition 37** (BWT trees). *Let BWT be the set of binary trees such that $T \in BWT$ if the children of T are partitioned into the left and right children and every node has a leaf as a descendant, and the lexicographic order on the leaves is a well-order.*

▶ **Lemma 38.** *Let X be an antichain in the full binary tree such that $(X, <_{\text{lex}})$ is isomorphic to an ordinal. Then $T := (X\downarrow, <)$ is in BWT.*

▶ **Lemma 39.** *Let $\pi = u_1u_1\cdots \in \{L,R\}^\omega$ be an ω-branch of $T \in BWT$.*

Let T_i (for $i \in \omega$) be the subtree of T over $X_i := \{v \mid v \geqslant u_1 \dots u_i \text{ and } \neg(v \geqslant u_1 \dots u_{i+1})\}$. Then

1. *$T_i \in BWT$.*
2. *Infinitely many T_i have more than one element.*
3. *There is i_R such that for every $i \geqslant i_R$: if T_i has more than one element, then $u_{i+1} = R$.*

Proof.

(1) $T_i \in BWT$. Indeed (a) every node in X_i has a leaf descendant in X_i, (b) the leaves are well-ordered by $<_{\text{lex}}$ and (c) children are partitioned into left/right as in T.

(2) If T_i is singleton for all $i > j$, then for all $i > j$ the i-th node on π has no leaf as descendant. Contradiction.

(3) Let $F := \{v \mid vL \in \pi \text{ and there is a leaf above } vR\}$. We claim that F is finite. Indeed if $i \in F$ and u_i is a leaf above vR then $vL <_{\text{lex}} u_i$ and every descendant u of vL is $<_{\text{lex}} u_i$. In particular, $u_{i+1} <_{\text{lex}} u_i$ for all $i \in F$. Hence, if F is infinite, then $<_{\text{lex}}$ is not well-order on the leaves. Contradiction. ◀

The $+$ and ω-sum operations on trees are refined by sums and infinite sums of BWT trees.

▶ **Definition 40** (Sums of BWT trees). *Let $T_i = (|T_i|, <_i)$ for $i \in \{0,1\}$ be BWT trees over disjoint universes. If the root of T_0 does not have right (respectively left) child define $T_0 +_R T_1$ (respectively, $T_0 +_L T_1$) to be a tree with universe $|T_0| \cup |T_1|$ and its order relation defined as $n_1 \leqslant n_2$ if there is $i \in \{0,1\}$ such that $n_1, n_2 \in |T_i|$ and $n_1 \leqslant_i n_2$, or n_1 is the root of T_0; the root of T_1 becomes right (respectively, left) child of the root of T_0, and for other nodes their left/right status is inherited from T_0 and T_1.*

The infinite sums of BWT trees is defined as follows.

▶ **Definition 41** (Infinite sums of BWT trees). *Let $T_i = (|T_i|, <_i)$ for $i \in \omega$ be BWT trees over disjoint universes, and let $u_1u_2\cdots \in \{L,R\}^\omega$ be an ω-string. $\sum^u_{i\in\omega} T_i$ is defined if*

1. *The root of T_i does not have u_{i+1} child ($i \in \omega$).*
2. *Infinitely many of T_i have more than one element.*
3. *There is i_R such that for $i \geqslant i_R$: if T_i has more than one element, then $u_{i+1} = R$.*

We define the tree

$$\sum^u_{i\in\omega} T_i$$

as having universe $\bigcup_{i\in\omega} |T_i|$ and its order relation is defined as $n_1 \leqslant n_2$ if there is i such that $n_1, n_2 \in |T_i|$ and $n_1 \leqslant_i n_2$, or n_1 is the root of T_i and $n_2 \in |T_j|$ for $j \geqslant i$.

The root of T_{i+1} becomes the right (respectively, left) child of the root of T_i if $u_{i+1} = R$ (respectively, $u_{i+1} = L$), and for other nodes of T_i their left/right status is inherited from T_i.

Note the requirement that infinitely many of T_i have more than one element ensures that every node in the ω-sum has a leaf as a descendant. The third requirement ensures that the lexicographic order on the leaves of $\sum^u_{i\in\omega} T_i$ is well-order.

If the disjointness assumption does not hold in the above definitions, we replace T_i by disjoint isomorphic copies and proceed as above.

▶ **Lemma 42.**
1. *If T_i are in BWT and $T_0 +_R T_1$ is defined, then $T_0 +_R T_1$ is in BWT.*
2. *If T_i are in BWT and $T_0 +_L T_1$ is defined, then $T_0 +_L T_1$ is in BWT.*
3. *If T_i are in BWT and $\sum^u_{i\in\omega} T_i$ is defined, then $\sum^u_{i\in\omega} T_i$ is in BWT.*

For $T \in BWT$, we denote by $OTP(T)$ the order type of the lexicographic order on the leaves of T.

▶ **Lemma 43.**
1. $OTP(T_0 +_R T_1) = OTP(T_0) + OTP(T_1)$.
2. $OTP(T_0 +_L T_1) = OTP(T_1) + OTP(T_0)$.
3. *Assume that if T_i have more than one element, then $u_{i+1} = R$. Then $OTP(\sum^u_{i\in\omega} T_i) = \sum_{i\in\omega} OTP(T_i)$.*

Define $BWT_\alpha := \mathsf{CBrank}_\alpha \cap BWT$ and $BWT^+_\alpha := \mathsf{CBrank}^+_\alpha \cap BWT$, where α is a countable ordinal. Note that BWT_0 are one element trees and BWT^+_0 are finite binary trees with a partition of children into left and right.

▶ **Lemma 44.** *For $\alpha > 0$.*
1. *$T \in BWT_\alpha$ if and only if $T = \sum^u_{i\in\omega} T_i$ for $T_i \in \cup_{\beta<\alpha} BWT^+_\beta$.*
2. *$T \in BWT^+_\alpha$ if and only if there are $T'_0, \dots, T'_k \in \cup_{\beta\leqslant\alpha} BWT_\beta$ and $v_1 \dots v_k \in \{L, R.\}$ such that $T = (((T'_0 +_{v_1} T'_1) +_{v_2} T'_2) \cdots +_{v_k} T'_k)$.*
3. *Assume that $\mathsf{CBrank}(T) = \alpha$ and $T \in BWT_\alpha$. Let u and T_i be as in 1. Then*
   a. *If α is limit, then for every $\beta < \alpha$ there is $T_i \notin BWT^+_\beta$.*
   b. *If $\alpha = \gamma + 1$, then infinitely often $T_i \in BWT^+_\gamma \setminus \cup_{\beta<\gamma} BWT^+_\beta$.*

Proof. (1) $\Leftarrow$ direction is easily follows by the induction on α.

$\Rightarrow$-direction. $T \in \mathsf{CBrank}_\alpha$, therefore there is an ω-branch $\pi = u_1 u_2 \cdots \in \{L, R\}^\omega$ such that $T = \sum_{i\in\omega} T_i$, where $T_i \in \cup_{\beta<\alpha} \mathsf{CBrank}^+_\beta$ is the subtree of T over $X_i := \{v \mid v \geqslant u_1 \dots u_{i-1}$ and $\neg(v \geqslant u_1 \dots u_i)\}$. Since $T \in BWT$, it follows that $T_i \in BWT$ and therefore, $T_i \in \cup_{\beta<\alpha} BWT^+_\beta$. It is also clear that $T = \sum^\pi_{i\in\omega} T_i$.

(2) and (3) easily follows from the definitions. ◀

Proof Theorem 17. Let X be an antichain in the full binary tree such that $(X, <_{\text{lex}})$ is isomorphic to an ordinal. We have to prove that $T := (X\downarrow, <)$ has Cantor-Bendixson rank α if and only if the order type of $(X, <_{\text{lex}})$ belongs to $[\omega^\alpha, \omega^{\alpha+1})$.

Note that T is a BWT by Theorem 38. The proof proceeds by induction on α using Theorem 43 and Theorem 44.

The base case $\alpha = 0$ is immediate.

Assume that Lemma holds for all $\beta < \alpha$.

Let $T \in BWT_\alpha$. By Theorem 44, $T = \sum^u_{i\in\omega} T_i$ for $T_i \in \cup_{\beta<\alpha} BWT^+_\beta$. By the inductive hypothesis $OTP(T_i) < \omega^\alpha$.

Let $u^k := u_{k+1}u_{k+2}\dots$. Then $u := u_1 \dots u_k u^k$. By Theorem 39(3), we can choose k such that if T_{k+i} have more than one element, then $u_{k+i+1} = R$. Hence, by Theorem 43(3), $OTP(\sum^{u^k}_{i\in\omega} T_{k+i}) = \sum_{i\in\omega} OTP(T_{i+k}) < \omega^\alpha \times \omega$. $T := (T_0 +_{u_1} +(T_1 +_{u_2} +_{u_3} (\cdots +_{u_k} T'))))$, where $T' := \sum^{u^k}_{i\in\omega} T_{k+i}$. We proved that $OTP(T_i) < \omega^\alpha$ and $OTP(T') < \omega^\alpha \times \omega = \omega^{\alpha+1}$. Hence, by Theorem 43(1)-(2), $OTP(T) < \omega^{\alpha+1}$.

Let us show that $OTP(T) \geqslant \omega^\alpha$.

By Theorem 44(3), if α is limit then there is no $\beta < \alpha$ such that all $T_i \in BWT^+_\beta$ for $i > k$. Therefore, there is no $\beta < \alpha$ such that $OTP(T_i) < \omega^\beta$ for $i > k$. Hence $OTP(\sum_{i\in\omega}^{u^k} T_{k+i}) \geqslant \omega^\alpha$.

By Theorem 44(3), if $\alpha = \gamma + 1$ is a successor, then infinitely often $T_i \in BWT^+_\gamma \setminus \cup_{\beta<\gamma} BWT^+_\beta$. Hence, by the inductive hypothesis, infinitely often $OTP(T_i) \geqslant \omega^\gamma$. Hence, $OTP(\sum_{i\in\omega}^{u'} T_{k+i}) \geqslant \omega^\gamma \times \omega = \omega^\alpha$. Therefore, $OTP(T) \geqslant \omega^\alpha$. This completes the proof for the case when $T \in BWT_\alpha$.

Now assume that $T \in BWT^+_\alpha$ and $\mathsf{CBrank}(T) = \alpha$. Then T is a sum of trees in BWT_α where at least one of the summands has CBrank equal to α. Therefore, the OTP of this summand is at least ω^α, and hence, the OTP of the sum is at least ω^α. ◀

B Beyond countable ordinals

In this section, we consider the decidability questions concerning $\mathsf{MSO}[\mathsf{otp}_\alpha]$ for α smaller than $\omega_1^{\omega_1}$. It essentially relies, as for the countable case, on the techniques in Sections 4–6, but it requires a bit more care. Also, some of the arguments go beyond $\omega_1^{\omega_1}$, but not all.

Thus, our first task is to understand precisely what are the MSO-definable ordinals beyond the countable. Let ω_1 be the first uncountable ordinal and ω_2 be the initial ordinal of the cardinal $\aleph_2$.

Let us first note that ω_1 is MSO-definable, indeed, this is the least limit ordinal which is not of cofinality ω. It is easy to express in an MSO that an ordinal is limit and that an ordinal has a cofinal ω-sequence. In the same way, ω_2 is MSO-definable since it is the least ordinal which is not of cofinality ω_1 or ω.

In Section B.1 we characterize MSO-definable ordinals in $[\omega_1, \omega_2)$. In Section B.2 we consider ordinals of the form ω_1^β where β is countable. We prove that for these ordinals $\mathsf{MSO}[\mathsf{otp}_{\omega_1^\beta}]$ is decidable if and only if ω_1^β is MSO definable. In Section B.3 we extend the equivalence between decidability and definability to all ordinals $< \omega_1^{\omega_1}$. The ordinal $\omega_1^{\omega_1}$ is undefinable, unfortunately we do not know whether $\mathsf{MSO}[\mathsf{otp}_{\omega_1^{\omega_1}}]$ is decidable.

B.1 Definable ordinals $< \omega_2$

In this subsection we characterize the MSO-definable ordinals $< \omega_2$. We use the following variant of the Cantor normal form: If $\alpha \in (0, \omega_2)$, then α has a unique decomposition of the form

$$\alpha = \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_0} \times \gamma_0 \ ,$$

where $\omega_2 > \beta_n > \cdots > \beta_1 > \beta_0 \geqslant 0$ and γ_i is a non-zero countable ordinal for all i. Let us call it the *ω_1-representation of α*.

▶ **Proposition 45** (MSO-definable ordinals). *An ordinal $\alpha < \omega_2$ is MSO-definable if and only if its ω_1-representation $\omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_0} \times \gamma_0$ is such that $\beta_i < \omega$ and $\gamma_i < \omega^\omega$ for all i.*

Proof. $\Leftarrow$ direction.

Let us recall that MSO-definable ordinals are closed under sum and multiplication. Therefore, every finite power of ω_1 is MSO-definable, and since every $\gamma < \omega^\omega$ is MSO-definable, we obtain that all the ordinals that have an ω_1-representation as in the statement, are MSO-definable.

Our proof of the other direction use elements of the compositional methods [32].

For every $n \in \mathbb{N}$ we say that ordinals α and β are $\equiv_n$-equivalent (notation $\alpha \equiv_n \beta$) if for every MSO sentence Φ of the quantifier depth at most n: $\alpha \models \Phi$ if and only if $\beta \models \Phi$. We will use the following well-known facts:

▶ **Fact 46** ($\equiv_n$ is a congruence). *The relation $\equiv_n$ is an equivalence relation and it is a congruence with respect to $+$ and $\times$. If $\alpha \equiv_n \beta$ then for every γ:*
1. *$\alpha + \gamma \equiv_n \beta + \gamma$ and $\gamma + \alpha \equiv_n \gamma + \beta$.*
2. *$\alpha \times \gamma \equiv_n \beta \times \gamma$*

▶ **Fact 47.** *There is a function $M : \mathbb{N} \to \mathbb{N}$ such that for every n if $\alpha \equiv_{M(n)} \beta$ then $\gamma \times \alpha \equiv_n \gamma \times \beta$.*

▶ **Fact 48** (cf Theorem 3.5(B) [31]).
1. *For every $\delta < \omega_2$ there is $\delta(k) < \omega_1^\omega$ such that $\delta \equiv_k \delta(k)$.*
2. *For every $\delta < \omega_1$ there is $\delta(k) < \omega^\omega$ such that $\delta \equiv_k \delta(k)$.*

Now we are ready to prove the $\Rightarrow$ direction of Theorem 45.

Assume that the ω_1-representation of α is

$$\alpha = \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_i} \times \gamma_i + \cdots + \omega_1^{\beta_0} \times \gamma_0 \ ,$$

If $\beta_i \geqslant \omega$ for some i, then $\alpha \geqslant \omega_1^\omega$. Therefore, if α satisfies a sentence Φ of quantifier depth k, by Fact 48(1), there is $\alpha(k) < \omega_1^\omega$ that satisfies Φ. Hence, α is not MSO definable.

Hence, if α is MSO definable then all $\beta_i < \omega$. Now assume that $\gamma_i > \omega^\omega$ for some i. Toward a contradiction, let α be definable by a sentence Φ of quantifier depth k. By Fact 48(2) there is $\gamma_i' < \omega^\omega$ such that $\gamma \equiv_{M(k)} \gamma_i'$, where M is a function from Fact 47. Therefore, by Fact 47, $\omega_1^{\beta_i} \times \gamma_i \equiv_k \omega_1^{\beta_i} \times \gamma_i'$. Hence, by Fact 46, α is $\equiv_k$-equivalent to α' which has the following ω_1-representation:

$$\alpha' = \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_i} \times \gamma_i' + \cdots + \omega_1^{\beta_0} \times \gamma_0$$

Therefore, $\alpha' \models \Phi$. But $\alpha' \neq \alpha$ (by uniqueness of ω_1 representation) and this contradicts that α is definable by Φ. ◀

B.2 ω_1^β for countable β

In this subsection we consider ordinals of the form ω_1^β where β is countable. We prove that for these ordinals MSO[$\mathsf{otp}_{\omega_1^\beta}$] is decidable if and only if ω_1^β is MSO definable.

Theorem 24 states that if β is of cofinality ω, the MSO[$\mathsf{otp}_{\omega^\beta}$]-theory of ω^β is undecidable.

Note that $\omega_1 = \omega^{\omega_1}$ and $\omega_1^\beta = \omega^{\omega_1 \times \beta}$. Observe that if β is ω-cofinal then $\omega_1 \times \beta$ is ω-cofinal. Hence,

▶ **Corollary 49.** *If β is ω-cofinal, then the MSO[$\mathsf{otp}_{\omega_1^\beta}$]-theory of ω_1^β is undecidable.*

Now we are going to show how to reduce MSO[ω_1^β] to MSO[$\omega_1^{\beta+1}$]. From this reduction and Theorem 49 we deduce undecidability of MSO[ω_1^β] for all $\beta \in [\omega, \omega_1)$.

The following standard facts hold:

▶ **Fact 50.**
1. *Let $(\alpha_i)_{i \in \omega}$ be an ω-sequence such that $\alpha_i < \omega^\beta$ for all i. Then,*

$$\sum_{i \in \omega} \alpha_i \leqslant \omega^\beta.$$

2. *Let* $(\alpha_i)_{i\in\omega}$ *be an* ω*-sequences of ordinals smaller than* $\omega^{\beta+1}$.

$$\sum_{i\in\omega}\alpha_i = \omega^{\beta+1} \text{ if and only if } \alpha_j \geqslant \omega^\beta \text{ for cofinally many } j.$$

3. *Let* $(\alpha_i)_{i\in\omega_1}$ *be an* ω_1*-sequences of ordinals smaller than* $\omega_1^{\beta+1}$.

$$\sum_{i\in\omega_1}\alpha_i = \omega_1^{\beta+1} \text{ if and only if } \alpha_j \geqslant \omega_1^\beta \text{ for cofinally many } j.$$

4. *Let* $(\alpha_i)_{i\in\omega_1}$ *be an* ω_1*-sequence such that* $\alpha_i < \omega_1^\beta$ *for all* i. *Then,*

$$\sum_{i\in\omega_1}\alpha_i \leqslant \omega_1^\beta.$$

Theorem 50(1)-(2) was used in the reduction of $\mathsf{MSO}[\omega^\beta]$ to $\mathsf{MSO}[\omega^{\beta+1}]$. We will use Theorem 50(3)-(4) in our reduction of $\mathsf{MSO}[\omega_1^\beta]$ to $\mathsf{MSO}[\omega_1^{\beta+1}]$.

Note that for $\beta \geq \omega$, it is impossible to express in $\mathsf{MSO}[\omega_1^{\beta+1}]$ "an interval is of length ω_1^β." Even there is no $\mathsf{MSO}[\omega_1^{\beta+1}]$ formula $B(x)$ such that $\omega_1^{\beta+1} \models B(b)$ if and only if the interval $[0,b) := \{c \mid c < b\}$ has the order type ω^β.

Our first aim is to chop $\omega_1^{\beta+1}$ into ω_1 disjoint intervals all of length ω_1^β, except a countable many.

▶ **Lemma 51.** *There is an* $\mathsf{MSO}[\omega_1^{\beta+1}]$ *formula* $\mathit{Chop}(X_B, X_E)$ *such that for every* $B, E \subseteq \omega_1^{\beta+1}$ *we have* $\omega_1^{\beta+1} \models \mathit{Chop}(B,E)$ *if and only if*

1. B *and* E *have order type* ω_1 *and* $b_i < e_i < b_{i+1}$ *for* $i \in \omega_1$. *Hence, for* $i \neq j$ *the intervals* (b_i, e_i) *and* (b_j, e_j) *are disjoint.*
2. *For all but countable many* i, *the order type of* $[b_i, e_i)$ *is* ω_1^β.

Proof. Let $B := (b_i)_{i\in\omega_1}$ and $E := (e_i)_{i\in\omega_1}$ be increasing ω_1 sequences such that $b_i < e_i < b_{i+1}$ for all $i \in \omega_1$. (This can be formalized in MSO.) We want to ensure that $\mathrm{OTP}([b_i, e_i)) = \omega_1^\beta$ for all $i \in \omega_1$, except countable many one.

Let us formalize it as follows:

Requirements.

(1) $\bigcup_{i\in I}[b_i, e_i)$ has order type $\omega_1^{\beta+1}$ for every cofinal subset I of ω_1, and

(2) for all $C := (c_i)_{i\in\omega_1}$ such that $c_i \in [b_i, e_i)$ the order type of $\bigcup[b_i, c_i)$ is not $\omega_1^{\beta+1}$ (it should be $< \omega_1^{\beta+1}$).

It is easy to formalize (1) and (2) by an $\mathsf{MSO}[\omega_1^{\beta+1}]$ formula.

We claim that (1) and (2) hold if and only if for all, but countable many i: $\mathrm{OTP}([b_i, e_i)) = \omega_1^\beta$.

Indeed, if for all i: $\mathrm{OTP}([b_i, e_i)) = \omega_1^\beta$, then, by Theorem 50(4), for all $C := (c_i)_{i\in\omega_1}$ such that $c_i \in [b_i, e_i)$ we have $\mathrm{OTP}((\bigcup[b_i, c_i)) \leqslant \omega_1^\beta$. Therefore, if for all but countable many i: $\mathrm{OTP}([b_i, e_i)) = \omega_1^\beta$, then $\mathrm{OTP}((\bigcup[b_i, c_i)) < \omega_1^{\beta+1}$. Therefore, (1) and (2) hold.

For the other direction. If (1) holds, then by Theorem 50(4) there are at most countable many i such that $\mathrm{OTP}([b_i, e_i)) < \omega_1^\beta$. If (2) holds, then, by Theorem 50(3), there are at most countable many i such that $\mathrm{OTP}([b_i, e_i)) > \omega_1^\beta$. Hence, there are at most countable many i such that $\mathrm{OTP}([b_i, e_1)) \neq \omega_1^\beta$. We proved the Lemma ◀

Now we will reduce $\mathsf{MSO}[\omega_1^\beta]$ to $\mathsf{MSO}[\omega_1^{\beta+1}]$. This is exactly like the reduction of Theorem 28, but for negation we should replace[2] "infinite" by ω_1.

[2] Instead of the filter of co-finite subset of ω, we use the filter of co-countable subsets of ω_1.

▶ **Lemma 52.** *There is an algorithm which given an* MSO$[\omega_1^\beta]$*-formula* $\varphi(X_1, \dots X_k)$ *constructs an* MSO$[\omega_1^{\beta+1}]$*-formula* $\varphi^*(X_B, X_E, X_1, \dots X_k)$ *such that for all* $A_1, \dots, A_k \subseteq \omega_1^{\beta+1}$ *and all* B, E *which satisfy* $\omega_1^{\beta+1} \models \mathit{Chop}(B, E)$*:*

$$\omega_1^{\beta+1} \models \varphi^*(B, E, A_1, \dots, A_k)$$

if and only if

$$\omega_1^{\beta+1}|_{[b_i, e_i)} \models \varphi(A_1 \cap [b_i, e_i), \dots, A_k \cap [b_i, e_i))$$

for all but countable many i*.*

Proof. We shall define the formula φ^* by structural induction on φ, and establish the conclusions of the lemma at the same time. The case of existential quantifier, the case of conjunction, and the case of MSO predicates are elementary. The crucial point is the negation.

Case of a conjunction, i.e., $\varphi(\bar{X}) = (\varphi_1(\bar{X}) \wedge \varphi_2(\bar{X}))$. We define

$$\varphi^*(X_B, X_E, \bar{X}) := \varphi_1^*(X_B, X_E, \bar{X}) \wedge \varphi_2^*(X_B, X_E, \bar{X}) \ .$$

Correctness follows from the fact that co-countable subsets of ω_1 are closed under the intersection.

Case of an existential set quantifier, i.e., $\varphi(\bar{X}) = \exists Y.\varphi_1(\bar{X}, Y)$. We define

$$\varphi^*(X_B, X_E, \bar{X}) := \exists Y.\ \varphi_1^*(X_B, X_E, \bar{X}, Y) \ .$$

Correctness is also straightforward.

Case of an MSO-formula $\varphi(\bar{X})$, and in particular of the atomic formulas $x \in Y$ and $x < y$ is easy and very similar to Theorem 28.

Case of an order type predicate, i.e., $\varphi(\bar{X}) := \mathsf{otp}_{\omega_1^\beta}(X_m)$. For simplicity, we shall treat the case of $\varphi(X) := \neg\mathsf{otp}_{\omega_1^\beta}(X_m)$, and leave the question of removing the negation to the negation case below. We set:

$$\varphi^*(X_B, X_E, X_m) := \mathit{Chop}(X_B, X_E) \wedge \neg\mathsf{otp}_{\omega_1^{\beta+1}}(X_m \cap F), \ .$$

where $F := \{z \mid \exists x \in X_B \exists y \in X_E (x \leqslant z < y \wedge (x, y) \cap X_B = \emptyset \wedge (x, y) \cap X_E = \emptyset)\}$. The correctness of this construction relies on Theorem 50.

Case of a negation, i.e., $\varphi(\bar{X}) := \neg\varphi_1(X_1, \dots, X_k)$. We set $\varphi^*(X_B, X_E, \bar{X})$ to be the conjunction of

A $\mathit{Chop}(X_B, X_E)$, and

B $\neg\varphi_1^*(X'_B, X'_E, \bar{X})$ holds for every X'_B, X'_E which are cofinal subsets of X_B and X_E such that $\mathit{Chop}(X'_B, X'_E)$.

Let us assume that $\omega^{\beta+1} \models \mathit{Chop}(B, E)$. Then $B := \{b_i \mid i \in \omega_1\}$ and $E := \{e_i \mid i \in \omega_1\}$, where b_i and e_i are increasing ω_1-sequences.

Condition **B** is equivalent to "$\neg\varphi_1(\bar{A} \cap [b_i, e_i))$ holds for all but countable many i.

Hence, the inductive hypothesis holds. ◀

As a consequence of Theorem 49 and Theorem 52 we obtain:

▶ **Corollary 53.** *The* MSO$[\omega_1^\beta]$ *is undecidable on* ω_1^β *for* $\beta \in [\omega, \omega_1)$*.*

B.3 Definability is equivalent to Decidability for $\alpha < \omega_1^{\omega_1}$

We are ready to extend Theorem 2 up to $\omega_1^{\omega_1}$.

▶ **Theorem 54.** *For all ordinals $\alpha < \omega_1^{\omega_1}$, the MSO[$\mathrm{otp}_\alpha$]-theory of α is decidable if and only if α is MSO-definable.*

Proof. For countable α it was proved in Theorem 30.

If α is definable, then MSO[α] is equivalent to MSO. Since, the MSO theory of every $\alpha < \omega_2$ is decidable, we obtain that the MSO[α] theory of α is decidable.

It remains to show that if an uncountable $\alpha < \omega_1^{\omega_1}$ is undefinable, then MSO[α] is undecidable.

Let $\alpha = \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_0} \times \gamma_0$, where $\omega_2 > \beta_n > \cdots > \beta_1 > \beta_0 \geqslant 0$ and γ_i is a non-zero countable ordinal for all i. Assume that $\alpha < \omega_1^{\omega_1}$ and α is undefinable.

By Proposition 45, there is i such that (A) $\beta_i \geqslant \omega$ or (B) $\gamma_i \geqslant \omega^\omega$.

If (A) holds then $\beta_n \in [\omega, \omega_1)$. In this case there is an MSO[α] formula $B(x)$ such that $\alpha \models B(b)$ if and only if the interval $[0, b)$ has the order type $\omega_1^{\beta_n}$. Indeed, let $[x, \infty)$ (respectively, $[0, x)$) be the set $\{y \mid y \leqslant x\}$ (respectively, $\{y \mid y < x\}$) and let $B(x)$ says that x is the minimal element such that $\neg\alpha([x, \infty))$. It is clear that $\alpha \models B(b)$ if the interval $[0, b)$ has the order type $\omega_1^{\beta_n}$. Now, similarly to the proof of Theorem 31, for every MSO[$\omega_1^{\beta_n}$] sentence C we can (effectively) construct an MSO[α] sentence C^* such that $\omega_1^{\beta_n} \models C$ if and only if $\alpha \models C^*$. Since, MSO[$\omega_1^{\beta_n}$] is undecidable by Corollary 53, we derive that MSO[α] is undecidable.

If (A) does not holds and (B) holds we have that all $\beta_i < \omega$ and therefore $\omega_1^{\beta_i}$ are definable for all $i \leqslant n$, and there are $\gamma_i \geqslant \omega^\omega$. Let i be the maximal index such that $\gamma_i \geqslant \omega^\omega$.

There is an MSO formula $B(x)$ such that $\alpha \models B(b)$ if and only if the order type of $[0, b)$ is $\delta := \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_{i+1}} \times \gamma_{i+1}$ indeed δ is a definable ordinal, hence such B exists. There is an MSO formula $E(y)$ such that $\alpha \models E(e)$ if the order type of $[0, e)$ is $\delta_1 := \omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_i} \times \gamma_i$. Indeed $E(x)$ states x is the minimal such that that the order type of $[x, \infty)$ is less than $\omega_1^{\beta_i}$. The order type of $[b, e)$ is $\mu := \omega_1^{\beta_i} \times \gamma_i$.

Now we will use two reductions. The first one reduces MSO[γ_i] to MSO[$\omega_1^{\beta_i} \times \gamma_i$]

▷ Claim 55. For every $k < \omega$ and an $MSO[\gamma]$ sentence A there is an MSO[$\omega_1^k \times \gamma$] sentence A^* such that $\gamma \models A$ if and only if $\omega_1^k \times \gamma \models A^*$.

Proof. Since an ordinal ω_1^k is MSO definable, there is a formula $Mult(X)$ which defines the set of multiples of ω_1^k, i.e., for every ordinal α: $\alpha \models Mult(S)$ if and only if $S := \{a \in \alpha \mid [0, a)$ is a multiple of $\omega_1^k\}$. Let $A^\star$ be defined as $\exists S(Mult(S) \wedge B)$, where B is the relativisation of A on S.

For an MSO (or MSO[γ]) sentence A: $\gamma \models A$ if and only if $\omega_1^k \times \gamma \models A^\star$. If A is an MSO[γ] sentence, then $A^\star$ is an MSO[γ] sentence, and we have to express γ-order type predicates "$X \subseteq S$ has order type γ" by $\omega_1^k \times \gamma$ order type predicates.

For $x = \omega_1^k \times \beta$ (a multiple of ω_1^k), we denote by $[x]$ an interval $[\omega_1^k \times \beta, \omega_1^k \times (\beta + 1))$. For a set X of multiples of ω_1^k, we denote by $[X]$ the set $\cup_{x \in X}[x]$. Note that $[X]$ is MSO definable from X.

Hence, our desired A^* can be defined from $A^\star$ by replacing the subformulas "X has the order type γ" by $\exists Y(Y = [X] \wedge$"Y has the order type $\omega_1^k \times \gamma$"). ◁

Since, $\gamma_i \geqslant \omega^\omega$ and it is countable, MSO[γ_i] is undecidable, hence, we obtain by Theorem 55 that MSO[$\omega_1^{\beta_i} \times \gamma_i$] is undecidable.

The second reduction reduces MSO[$\omega_1^{\beta_i} \times \gamma_i$] to MSO[$\alpha$].

▷ Claim 56. For every $\mathsf{MSO}[\omega_1^{\beta_i} \times \gamma_i]$ sentence A there is an $\mathsf{MSO}[\alpha]$ sentence A^* such that $\omega_1^{\beta_i} \times \gamma_i \models A$ if and only if $\alpha \models A^*$.

Proof. Recall that there are MSO formulas $B(x)$ and $E(y)$ such that $\alpha \models B(b) \wedge E(e)$ if and only if $[0, b)$ has the order type $\omega_1^{\beta_n} \times \gamma_n + \cdots + \omega_1^{\beta_{i+1}} \times \gamma_{i+1}$ and $[b, e)$ has the order type $\omega_1^{\beta_i} \times \gamma_i$.

As A^* we can take $\exists xy(B(x) \wedge E(y) \wedge C)$, where C is obtained from A first by relativizing A to the interval $[x, y)$. and then replacing atomic subformulas "X has order type $\omega_1^{\beta_i} \times \gamma_i$" by "$X \cup Z$ has the order type α," where $Z := \{z \mid z < x \vee z \geqslant y\}$. ◁

Since $\mathsf{MSO}[\omega_1^{\beta_i} \times \gamma_i]$ is undecidable, we obtain by Theorem 56 that $\mathsf{MSO}[\alpha]$ is undecidable. and this completes our proof of Theorem 54. ◀

First-Order Logic with Equicardinality in Random Graphs

Simi Haber
Bar-Ilan University, Ramat Gan, Israel

Tal Hershko
California Institute of Technology, Pasadena, CA, USA

Mostafa Mirabi
Taft School, Watertown, CT, USA

Saharon Shelah
Hebrew University of Jerusalem, Israel

Abstract

We answer a question of Blass and Harary about the validity of the zero-one law in random graphs for extensions of first-order logic (FOL). For a given graph property P, the *Lindström extension* of FOL by P is defined as the minimal (regular) extension of FOL able to express P. For several graph properties P (e.g. Hamiltonicity), it is known that the Lindström extension by P is also able to interpret a segment of arithmetic, and thus strongly disobeys the zero-one law. Common to all these properties is the ability to express the Härtig quantifier, a natural extension of FOL testing if two definable sets are of the same size. We prove that the Härtig quantifier is sufficient for the interpretation of arithmetic, thus providing a general result which implies all known cases of Lindström extensions which are able to interpret a segment of arithmetic.

2012 ACM Subject Classification Theory of computation → Finite Model Theory; Mathematics of computing → Random graphs

Keywords and phrases finite model theory, first-order logic, monadic second-order logic, random graphs, zero-one laws, generalized quantifiers, equicardinality

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.12

Funding *Saharon Shelah*: Research partially supported by the grant "Independent Theories" NSF-BSF, (BSF 3013005232) and by NSF grant no: DMS 1833363. Paper 1250 on Shelah's publication list.

1 Introduction

In this paper we study the Erdös-Rényi binomial random graph model $G(n,p)$. Recall that $G(n,p)$ is defined as a probability distribution over the set of all labeled (simple) graphs with the vertex set $[n] := \{1,2,\ldots,n\}$, by requiring that each of the $\binom{n}{2}$ potential edges appears with probability p and independently of all other edges. Note that $G\left(n,\frac{1}{2}\right)$ is the uniform distribution over the set of $2^{\binom{n}{2}}$ labeled graphs with vertex set $[n]$.

In what follows, we use $\mathbb{P}(\cdot)$ to denote probabilities and $\mathbb{E}(\cdot)$ to denote expected values.

1.1 Background and Previous Results

The study of random graphs was pioneered by Erdös and Rényi in the 1960s, originating from two seminal papers [9, 10]. One of the earliest phenomena recognized in their work is the fact that many natural graph properties – including connectivity, Hamiltonicity, planarity,

k-colorability for a fixed k and containing H as a subgraph for a fixed graph H – hold either in almost all graphs, or in almost none of them. Formally, let P be a graph property and fix $p \in (0,1)$. We say that P holds *asymptotically almost surely* (a.a.s. for short) in $G(n,p)$ if

$$\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \text{ satisfies } P\right) = 1$$

and that P holds *asymptotically almost never* (a.a.n. for short) in $G(n,p)$ if

$$\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \text{ satisfies } P\right) = 0.$$

Then, for example, for every fixed $p \in (0,1)$,

- Connectivity holds a.a.s. in $G(n,p)$.
- Hamiltonicity holds a.a.s. in $G(n,p)$.
- Planarity holds a.a.n. in $G(n,p)$.
- For every fixed $k \in \mathbb{N}$, k-colorability holds a.a.n. in $G(n,p)$.
- For every fixed finite graph H, the property of containing H as a subgraph holds a.a.s. in $G(n,p)$.

As a reference, see any introductory text in random graphs, e.g. [19].

The observation that many natural graph properties hold either a.a.s. or a.a.n. in $G(n,p)$ motivates the following definition.

▶ **Definition 1.** *Let $\mathcal{A}$ be a set of graph properties. We say that $\mathcal{A}$* obeys the zero-one law *in $G(n,p)$ if for every property $P \in \mathcal{A}$,*

$$\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \text{ satisfies } P\right) \in \{0,1\}.$$

With this definition, we may formulate the following informal observation: if $\mathcal{A}$ is a set of natural graph properties then $\mathcal{A}$ obeys the zero-one law. This is not a formal statement, due to the lack of a formal definition of a "natural graph property".

From a logician's point of view, a natural class of graph properties is the class $\mathcal{FO}$ of *first-order properties*. These are properties which can be expressed as a sentence in the first-order language of graphs, whose signature consists of a single binary relation $\sim$ representing adjacency.[1] Indeed, a classic result, proven independently by Glebskii et al. [13] and Fagin [11], states that $\mathcal{FO}$ obeys the zero-one law in $G(n,p)$.

▶ **Theorem 2** (GKLT-Fagin). *Fix $p \in (0,1)$. Then the set of first-order graph properties $\mathcal{FO}$ obeys the zero-one law in $G(n,p)$.*

The GKLT-Fagin zero-one law pioneered the study of random graphs with tools of mathematical logic. This point of view has proved to be doubly beneficial, teaching us about the properties of the underlying random graph, and also about the expressive power of logical languages. It is therefore considered an important part of finite model theory.

The GKLT-Fagin zero-one law deals with first-order properties. However, many graph properties which are considered natural – including connectivity, Hamiltonicity and k-colorability – are not first-order. On the other extreme, the class $\mathcal{SO}$ of *second-order* graph properties contains all the properties listed above, but fails to obey the zero-one law. For example, as noted by Fagin [11, p. 55], the property of having an even number of vertices is second-order, but clearly has no limiting probability.

[1] We shall often identify logical sentences with the properties they describe.

It is therefore natural to ask for extensions of first-order logic which have a stronger expressive power on the one hand, but still obey the zero-one law on the other hand. This question was posed by Blass and Harary [2, Section 5]. In their discussion, they suggest several guiding questions:

1. Is there an extension of first order logic which is strong enough to express Hamiltonicity and rigidity (asymmetry), but still obeys the zero-one law?
2. What about monadic second-order logic? It cannot express Hamiltonicity, but it is still an important extension of first-order logic – does it obey the zero-one law?
3. Can something be done with "more exotic languages", for example with equicardinality quantifiers?

These questions have been studied in many papers. We dedicate the following sections to explain the precise meaning of these questions in more detail and review previous results. Our review is not exhaustive; for a broader survey of results in this field, we refer the reader to [4] and [29]. For a more focused discussion of results directly related to our work, see [6] and [16].

Lindström Extensions

Suppose we are interested in an extension of $\mathcal{FO}$ which includes a certain graph property P. One option is to simply take the union $\mathcal{FO} \cup \{P\}$. However, this set of properties clearly lacks a basic notion of closure. To avoid such trivialities, we focus on *regular* extensions. A regular logic is a logic that is closed under negation, conjunction, existential quantification, relativization and substitution (see [7] for more details). We then have the following definition.

▶ **Definition 3.** *Let P be a graph property. The* Lindström extension *of $\mathcal{FO}$ by P, denoted $\mathcal{FO}[P]$, is the minimal regular extension of $\mathcal{FO}$ that includes P.*

The term *Lindström extension* comes from the fact that $\mathcal{FO}[P]$ can be constructed by adjoining the Lindström quantifier of P, denoted Q_P and defined as follows [26, 27, 7].

▶ **Definition 4.** *Let P be a graph property. Its Lindström quantifier Q_P is defined as follows.*

- *Syntactically, given a formula $\varphi_V(x, \vec{z})$ with $x, \vec{z}$ as free variables and a formula $\varphi_E(x, y, \vec{z})$ with $x, y, \vec{z}$ as free variables, it returns the formula $Q_P x, y\,(\varphi_V(x, \vec{z}), \varphi_E(x, y, \vec{z}))$ in which x, y are quantified and $\vec{z}$ are free.*
- *Semantically, the truth value of this formula is defined as follows. Let $G = (V, E)$ be a graph and let $\vec{a}$ be a vector of vertices, of the same length as $\vec{z}$. Let $V_0 = \{v \in V : G \models \varphi_V(v, \vec{a})\}$ and let E_0 be the set of pairs $\{u, v\}$ with $u, v \in V_0$ such that $G \models \varphi(u, v, \vec{a})$ or $G \models \varphi(v, u, \vec{a})$. Then*

$$G \models_{\vec{z}=\vec{a}} Q_P x, y\,(\varphi_V(x, \vec{z}), \varphi_E(x, y, \vec{z})) \iff G_0 = (V_0, E_0) \text{ satisfies } P.$$

It can be shown that $\mathcal{FO}[P]$ is the same as the closure of $\mathcal{FO}$ under quantification with Q_P (see [8] for more details).

We can now suggest a more precise formulation of the first question of Blass and Harary: do the Lindström extensions of $\mathcal{FO}$ by Hamiltonicity and by rigidity obey the zero-one law? The answer was given by Dawar and Grädel ([6], also in [5]).

▶ **Theorem 5** (Dawar-Grädel)**.** *Fix $p \in (0, 1)$.*

1. *The Lindström extension $\mathcal{FO}$[Rigidity] obeys the zero-one law in $G(n, p)$.*
2. *The Lindström extension $\mathcal{FO}$[Hamiltonicity] does not obey the zero-one law in $G(n, p)$.*

The latter part of the theorem was demonstrated by encoding Parity – the property of having an even number of vertices. It is worth noting, however, that Parity still allows for the possibility of a modular limit law, as shown by Kolaitis and Kopparty [25]. A far more extreme violation of the zero-one law occurs through the interpretation of a segment of arithmetic, a concept we will elucidate in the subsequent section.

Arithmetization

The second question of Blass and Harary, regarding monadic second order logic $\mathcal{MSO}$, was answered much earlier than the first. The answer was given by Kaufmann and Shelah [23], who proved that $\mathcal{MSO}$ disobeys the zero-one law in a very strong sense. Their main result is that $\mathcal{MSO}$ *can interpret arithmetic* in $G(n,p)$, which roughly means that there are sentences in $\mathcal{MSO}$ that define an arithmetic structure on (a subset of) the vertex set. *If a language $\mathcal{L}$ can interpret a segment of arithmetic then it is, in a sense, the farthest possible from obeying the zero-one law.* Indeed, in such a case, it is possible to construct sentences $\varphi \in \mathcal{L}$ whose probability sequence $\{\mathbb{P}(G(n,p) \models \varphi)\}_{n=1}^{\infty}$ exhibits different kinds of complex behaviors. We shall demonstrate this fact shortly by constructing a sentence $\varphi \in \mathcal{L}$ whose probability sequence alternates between near-zero values and near-one values, and hence has no limit.

The definition of interpreting arithmetic given below is somewhat weaker than what Kaufmann and Shelah prove for $\mathcal{MSO}$, but describes a more general and more common type of arithmetization results (see review below). In particular, it includes the main result of this paper, Theorem 14).

Recall that for $n \in \mathbb{N}$ we denote $[n] = \{1, 2, \ldots, n\}$. We begin by defining the language $\mathcal{SO}$[Arith] as the second-order language of arithmetic, where:

- Addition $+$ and multiplication $\times$ are defined as ternary relations (so, for example, we write $+(a,b,c)$ instead of $a+b=c$). This is a convenient choice when working with finite models such as $[n]$, where addition and multiplication are restricted.
- Second order quantification is done only over unary and binary relations.

▶ **Example 6.** We can construct a formula in $\mathcal{SO}$[Arith] with free variables x, y expressing the property $y = 2^x$ by the following steps.

- Begin by asserting the existence of a binary relation: $\exists \mathrm{Exp}^2$.
- Require that it is single-valued:

$$\forall a \forall b \forall b' \left(\mathrm{Exp}(a,b) \wedge \mathrm{Exp}(a,b') \to b = b'\right).$$

- Define the relation inductively:

$$\mathrm{Exp}(0,1) \wedge \forall a \forall b \forall a' \forall b' \left(\mathrm{Exp}(a,b) \wedge +(a,1,a') \wedge \times(b,2,b') \to \mathrm{Exp}(a',b')\right).$$

- Finally, require $\mathrm{Exp}(x,y)$.

For every $n \in \mathbb{N}$, the set $[n]$ admits an arithmetic structure with the standard addition and multiplication (restricted to $[n]$). Given an interval $[a,b] \subseteq \mathbb{R}$ and a sentence $\varphi \in \mathcal{SO}$[Arith], we say that:

1. φ *holds in* $[a,b]$ if $[n] \models \varphi$ for every $n \in [a,b] \cap \mathbb{N}$.
2. φ is *constant on* $[a,b]$ if φ holds in $[a,b]$ or $\neg\varphi$ holds in $[a,b]$.

Finally, let us say that a function $f\colon \mathbb{N} \to \mathbb{N}$ is *finite-to-one* if for every $m \in \mathbb{N}$, the inverse image $f^{-1}(m)$ is a non-empty finite set. Note that if f is finite-to-one then f is onto and $\lim_{n\to\infty} f(n) = \infty$. Examples include $f(n) = \lfloor \sqrt{n} \rfloor$ and $f(n) = \lfloor \log\log n \rfloor$.

▶ **Definition 7** (Arithmetization). *Let $\mathcal{L}$ be a logical language whose signature includes the binary relation symbol $\sim$, representing adjacency. Fix a parameter $p \in (0,1)$, constants $0 < c_1 \leq c_2$ and a finite-to-one function $f\colon \mathbb{N} \to \mathbb{N}$. We say that $\mathcal{L}$* can interpret a segment of arithmetic *in $G(n,p)$ with constants c_1, c_2 and a scaling function $f(n)$ if the following holds. For every sentence $\varphi \in \mathcal{SO}$[Arith] there exists a sentence $\varphi^* \in \mathcal{L}$ such that, given a sequence $\{n_k\}_{k=1}^{\infty}$ with $\lim_{k\to\infty} n_k = \infty$ such that φ is constant on $[c_1 f(n_k), c_2 f(n_k)]$ for every k, we have*

$$\lim_{k\to\infty} \mathbb{P}\left(G(n_k,p) \models \varphi^* \iff \varphi \text{ holds in } [c_1 f(n_k), c_2 f(n_k)]\right) = 1.$$

To motivate the definition, we remark that it reflects a general strategy of encoding arithmetic in random graphs, which can be roughly summarized as follows:

- Restrict to a certain $\mathcal{L}$-definable subset $S \subseteq [n]$ of size $|S| \in [c_1 f(n), c_2 f(n)]$.
- Use the structure of the random graph to encode unary and binary relations and an arithmetic structure on S.
- Given a sentence $\varphi \in \mathcal{SO}$[Arith], use the encoded structure to convert it into a sentence $\varphi^* \in \mathcal{L}$ asserting that φ is satisfied in S.

▶ **Proposition 8.** *Let $\mathcal{L}$ be a language that can interpret a segment of arithmetic in $G(n,p)$. Then there exists a sentence $\psi \in \mathcal{L}$ such that the limit $\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \models \psi\right)$ does not exist. In particular, $\mathcal{L}$ disobeys the zero-one law.*

Proof. Let $0 < c_1 \leq c_2$ and $f(n)$ be the constants and the scaling function for $\mathcal{L}$, as in Definition 7. Let $N = \lceil \max\{|\log_2 c_1|, |\log_2 c_2|\} \rceil + 1$. It is straightforward to construct a sentence $\varphi \in \mathcal{SO}$[Arith] such that for every $m \in \mathbb{N}$, $[m] \models \varphi$ if and only if

$$\lfloor \log_2 m \rfloor \equiv k \mod 8N \quad \text{for } k \in \{-N, -N+1, \ldots, N-1, N\}.$$

Let $\{n_k\}_{k=1}^{\infty}$ be a sequence satisfying $f(n_k) = 2^{4Nk}$ for every k. Such a sequence exists and approaches ∞, because f is finite-to-one. We claim that φ is constant on each interval $[c_1 f(n_k), c_2 f(n_k)]$. Indeed, for every $m \in [c_1 f(n_k), c_2 f(n_k)] \cap \mathbb{N}$ we have

$$\begin{aligned} \log_2 m &\in [\log_2 c_1 + 4Nk, \log_2 c_2 + 4Nk], \\ \lfloor \log_2 m \rfloor &\in [4Nk - N, 4Nk + N] \end{aligned} \tag{1}$$

where (1) follows from our choice of N. When k is even, (1) implies $[m] \models \varphi$. When k is odd, (1) implies $[m] \not\models \varphi$.

Now let $\varphi^* \in \mathcal{L}$ as in Definition 7. Then the probabilities sequence $\{\mathbb{P}\left(G(n_k,p) \models \varphi^*\right)\}_{k=1}^{\infty}$ converges to 1 on even values of k and converges to 0 on odd values of k. This implies that the limit $\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \models \varphi^*\right)$ does not exist. ◀

Going back to Kaufmann and Shelah, we can now formulate the following consequence of [23] (which follows from Theorem 1 and the closing remark).

▶ **Theorem 9** (Kaufmann-Shelah). *Fix $p \in (0,1)$. Then $\mathcal{MSO}$ can interpret a segment of arithmetic in $G(n,p)$ (with constants $c_1 = c_2 = 1$ and scaling function $f(n) = \lfloor \sqrt{n} \rfloor$).*

As explained, this result provides a strongly negative answer to the second question of Blass and Harary.

In [16], Haber and Shelah prove an arithmetization result for the Lindström extension of Hamiltonicity, thus strengthening Part 2 of Theorem 5.

▶ **Theorem 10** (Haber-Shelah). *Fix $p \in (0,1)$. Then $\mathcal{FO}$*[Hamiltonicity] *can interpret a segment of arithmetic in $G(n,p)$ (with scaling function $f(n) = \Omega(\log\log\log n)$).*

As for other graph properties, Haber and Shelah also proved in [16] that the zero-one law holds for the Lindström extensions $\mathcal{FO}$[Connectivity] and $\mathcal{FO}[k-\text{colorability}]$ for every fixed k. These results also follow from a more general theorem by Dawar and Grädel [6], which also implies that the zero-one law holds for $\mathcal{FO}$[Planarity]. On the other hand, there are additional graph properties P for which it is known that $\mathcal{FO}[P]$ can interpret a segment of arithmetic. These include regularity, the existence of a perfect matching [15] and the existence of a C_4-factor [14]. It is noteworthy that Haber and Shelah [16] employed a strategy to encode the Rescher plurality quantifier [28], resulting in a more expressive logic. In contrast, our finding that equicardinality alone suffices to interpret a segment of arithmetic is unexpected and significantly stronger.

Equicardinality Quantifiers

Common to all the Lindström extensions of $\mathcal{FO}$ which are known to be able to interpret a segment of arithmetic is the ability to express the *equicardinality quantifier*, also known as the *Härtig quantifier* [17], which we denote by $Q_=$. This quantifier allows for testing if two definable sets are of the same size.

▶ **Definition 11.** *The Härtig quantifier $Q_=$ is defined as follows.*

- *Syntactically, given formulas $\varphi(x,\vec{z})$ and $\psi(x,\vec{z})$ with free variables $x,\vec{z}$, it returns a formula $Q_=x\,(\varphi(x,\vec{z}),\psi(x,\vec{z}))$ in which x is quantified and $\vec{z}$ are free.*
- *Semantically, the truth value of this formula is defined as follows. Let $G=(V,E)$ be a finite graph and let $\vec{a}$ be a vector of vertices, of the same length as $\vec{z}$. Then*

$$G \models_{\vec{z}=\vec{a}} Q_=x\,(\varphi(x,\vec{z}),\psi(x,\vec{z})) \iff |\{v \in V : G \models \varphi(v,\vec{a})\}| = |\{v \in V : G \models \psi(v,\vec{a})\}|\,.$$

The following proposition shows that $Q_=$ is indeed expressible in $\mathcal{FO}$[Hamiltonicity]. Similar arguments show that $Q_=$ is also expressible in the other Lindström extensions of $\mathcal{FO}$ listed above that are known to be able to interpret a segment of arithmetic.

▶ **Proposition 12.** *Let $\varphi(x,\vec{z})$ and $\psi(x,\vec{z})$ be formulas in $\mathcal{FO}$*[Hamiltonicity] *with free variables $x,\vec{z}$. Then the formula $Q_=x\,(\varphi(x,\vec{z}),\psi(x,\vec{z}))$ is also expressible in $\mathcal{FO}$*[Hamiltonicity].

Proof. Fix a graph $G=(V,E)$ and a vector $\vec{a}$ of vertices. Let $A = \{x \in V : G \models \varphi(x,\vec{a})\}$ and $B = \{x \in V : G \models \psi(x,\vec{a})\}$. Also let $\varphi'(x,\vec{z}) = \varphi(x,\vec{z}) \wedge \neg\psi(x,\vec{z})$ (which defines the set $A \setminus B$) and $\psi'(x,\vec{z}) = \psi(x,\vec{z}) \wedge \neg\varphi(x,\vec{z})$ (which defines the set $B \setminus A$). Define $\varphi_V(x,\vec{z}) = \varphi'(x,\vec{z}) \vee \psi'(x,\vec{z})$ and $\varphi_E(x,y,\vec{z}) = \varphi'(x,\vec{z}) \wedge \psi'(y,\vec{z})$. Consider the graph $G_0=(V_0,E_0)$ defined by these formulas, as in Definition 4. Note that G_0 is the complete bipartite graph with sides $A \setminus B$ and $B \setminus A$. Recall that a complete bipartite graph is Hamiltonian if and only if its sides are of the same size. Therefore

$$\begin{aligned} G \models_{\vec{z}=\vec{a}} Q_{\mathrm{Ham}}x,y\,(\varphi_V(x,\vec{z}),\varphi_E(x,y,\vec{z})) &\iff |A \setminus B| = |B \setminus A| \\ &\iff |A| = |B| \iff G \models_{\vec{z}=\vec{a}} Q_=x\,(\varphi(x,\vec{z}),\psi(x,\vec{z}))\,. \end{aligned}$$

◀

Let $\mathcal{FO}[Q_=]$ denote the closure of $\mathcal{FO}$ under quantification with $Q_=$. This is a natural extension of first-order logic which has been studied quite extensively. For a survey on equicardinality quantifiers in the context of general model theory and abstract logic, see [18].

Equicardinality quantifiers have also been studied in the context of zero-one laws and convergence laws. In [12], Fayolle, Grumbach and Tollu studied zero-one laws for first-order logic enriched by generalized quantifiers, including Härtig quantifiers (equicardinality quantifiers) and Rescher quantifiers (expressing inequalities of cardinalities). Their results show that the zero-one law *does* hold for $\mathcal{FO}^*[Q_=]$, defined in the same way as $\mathcal{FO}[Q_=]$ but with the restriction that free variables are not allowed in the scope of $Q_=$-quantification.

▶ Remark 13. A simple argument, also mentioned in [12], proves that $\mathcal{FO}[Q_=]$ can express parity in the special case of $G\left(n, \frac{1}{2}\right)$. Indeed, consider the sentence $\exists z Q_= x\,(x \sim z, \neg(x \sim z))$, asserting the existence of a vertex with the same number of neighbors as non-neighbors. This sentence holds in $G\left(n, \frac{1}{2}\right)$ with probability 0 when n is even, and with probability approaching 1 when n is odd. In particular, $\mathcal{FO}[Q_=]$ does not satisfy the zero-one law in $G\left(n, \frac{1}{2}\right)$. As we shall soon see, much more can be said: $\mathcal{FO}[Q_=]$ can express not only parity, but a segment of arithmetic, and for every $p \in (0,1)$ (Theorem 14).

Finally, we mention that in addition to Lindström quantifiers and equicardinality quantifiers, other generalized quantifiers have also been studied in the context of zero-one laws and convergence laws. In a sequence of three papers [20, 21, 22], Kalia studied almost sure quantifier elimination, providing a method for proving convergence laws for logics with generalized quantifiers. In [24], Keisler and Lotfallah proved almost sure quantifier elimination logics with probability quantifiers.

1.2 Our Results

The main result of the paper is the following theorem.

▶ **Theorem 14** (Main Theorem). *Fix $p \in (0,1)$. Then $\mathcal{FO}[Q_=]$ can interpret a segment of arithmetic in $G(n,p)$ (with scaling function $f(n) = \left\lfloor \sqrt{\ln n} \right\rfloor$).*

In particular, from Proposition 8 we get the following corollary.

▶ **Corollary 15.** *Fix $p \in (0,1)$. Then there exists a sentence $\psi \in \mathcal{FO}[Q_=]$ such that the limit $\lim_{n\to\infty} \mathbb{P}\left(G(n,p) \models \psi\right)$ does not exist.*

This answers the third question of Blass and Harary [2, Section 5], and provides a general result which immediately implies all known cases of Lindström extensions of $\mathcal{FO}$ which are able to express arithmetic. As mentioned above, these include the Lindström quantifier of Hamiltonicity, regularity, the existence of a perfect matching and the existence of a C_4-factor.

The rest of the paper is dedicated to the proof of Theorem 14. The proof strategy is roughly as follows. Given a sentence $\varphi \in \mathcal{SO}[\mathrm{Arith}]$, first apply Theorem 9 to convert it into a sentence $\varphi^* \in \mathcal{MSO}$ which expresses φ on a set of size $\lfloor\sqrt{n}\rfloor$. Then, the crux of the proof is to convert a sentence $\varphi^* \in \mathcal{MSO}$ into a sentence $\psi \in \mathcal{FO}[Q_=]$ which expresses φ^* on a set of size $\Theta(\ln n)$. To do that, we need to show that $\mathcal{FO}[Q_=]$ can define subsets of logarithmic size, and can also interpret monadic second-order logic on such sets. The proof is divided between Sections 2 and 3. In Section 2 we develop the necessary probabilistic tools, and in Section 3 we put them together in order to complete the proof.

Notation and Conventions

We denote $\mathcal{FO}_= := \mathcal{FO}[Q_=]$ for short. Given a list of variable symbols $x_1, \ldots, x_n$, let $\mathcal{FO}(x_1, \ldots, x_n)$ denote the set of first-order formulas (in the language of graphs) with $x_1, \ldots, x_n$ as free variables. Similarly define $\mathcal{FO}_=(x_1, \ldots, x_n)$ and $\mathcal{MSO}(x_1, \ldots, x_n)$.

Throughout the text we maintain the convention of denoting random variables with a boldface font.

For $n \in \mathbb{N}$ and $p \in (0,1)$, we write $\mathbf{G}_n \sim G(n,p)$ to indicate that $\mathbf{G}_n$ is a random graph with distribution $G(n,p)$. For two vertices $u, v \in [n]$, let $u \sim v$ denote that they are adjacent in $\mathbf{G}_n$. For a subset $S \subseteq [n]$, let $\mathbf{G}_n[S]$ denote the subgraph of $\mathbf{G}_n$ induced by S.

We shall use the following notions of asymptotic probabilities. Let $(E_n)_{n=1}^{\infty}$ be a sequence of events, taken from a sequence of probability spaces.

1. We say that E_n holds *with high probability* (as $n \to \infty$) if

$$\mathbb{P}(E_n) = 1 - o(1).$$

2. We say that E_n holds *with exponentially high probability* (as $n \to \infty$) if

$$\mathbb{P}(E_n) = 1 - \exp\left(-n^{\Omega(1)}\right).$$

In addition, let $(\mathbf{X}_n)_{n=1}^{\infty}$, $(\mathbf{Y}_n)_{n=1}^{\infty}$ be two sequences of positive random variables. We say that $\mathbf{X}_n = (1+o(1))\,\mathbf{Y}_n$ with (exponentially) high probability if there exists a sequence $\varepsilon_n = o(1)$ such that the event $|\mathbf{X}_n/\mathbf{Y}_n - 1| \le \varepsilon_n$ holds with (exponentially) high probability.

For notational convenience, we sometimes omit dependency on n from our notation. The underlying assumption throughout the text is that all quantities implicitly depend on n (unless it is explicitly stated that they are constant or fixed) and $n \to \infty$. We explicitly refer to the dependency on n in cases where this convention may cause ambiguity.

Finally, recall the following tail bounds on binomial and Poisson variables, following from Chernoff's inequality (e.g. see [1, Appendix A]).

- Let $\mathbf{X} \sim \mathrm{Bin}(n,p)$ and $\mu = \mathbb{E}\mathbf{X}$. Then for every $0 < \delta < 1$,

$$\mathbb{P}(|\mathbf{X} - \mu| \ge \delta\mu) \le 2\exp\left(-\frac{\delta^2}{3}\mu\right). \tag{2}$$

- Let $\mathbf{X} \sim \mathrm{Pois}(\lambda)$ and $\mu = \mathbb{E}\mathbf{X}$. Then for every $0 < \delta < 1$,

$$\mathbb{P}(|\mathbf{X} - \mu| \ge \delta\mu) \le 2\exp\left(-\frac{\delta^2}{4}\mu\right). \tag{3}$$

2 Some Probabilistic Results

From now fix a constant $p \in (0,1)$ and consider a binomially distributed random graph $\mathbf{G}_n \sim G(n,p)$.

We begin by fixing, for every n, two arbitrary vertices $u_1, u_2 \in [n]$. Let $V' = [n] \setminus \{u_1, u_2\}$. Define the following (random) vertex sets:

$$\begin{aligned}
\mathbf{A} &= \{v \in V' : v \sim u_1 \wedge v \sim u_2\},\\
\mathbf{B} &= \{v \in V' : v \sim u_1 \wedge v \not\sim u_2\},\\
\mathbf{C} &= \{v \in V' : v \not\sim u_1 \wedge v \sim u_2\}.
\end{aligned}$$

Note that the statements $v \in \mathbf{A}, v \in \mathbf{B}, v \in \mathbf{C}$ are all expressible as formulas in $\mathcal{FO}(u_1, u_2, v)$.

From (2), with exponentially high probability we have

$$\begin{aligned}
|\mathbf{A}| &= (1+o(1))p^2 n,\\
|\mathbf{B}|, |\mathbf{C}| &= (1+o(1))p(1-p)n.
\end{aligned}$$

That is, there exists a sequence $\delta_n = o(1)$ such that the event (which we denote by $\mathcal{Q}$)

$$\frac{|\mathbf{A}|}{p^2 n}, \frac{|\mathbf{B}|}{p(1-p)n}, \frac{|\mathbf{C}|}{p(1-p)n} \in [1-\delta_n, 1+\delta_n] \tag{4}$$

holds with exponentially high probability. It will be convenient to condition on the values of the variables $\mathbf{A}, \mathbf{B}, \mathbf{C}$; that is, to condition on an event of the form $Q_{A,B,C} = \{\mathbf{A} = A, \mathbf{B} = B, \mathbf{C} = C\}$ where A, B, C are possible values of $\mathbf{A}, \mathbf{B}, \mathbf{C}$. Note that conditioning on $Q_{A,B,C}$ does not affect the distribution of $\mathbf{G}_n[V']$.

The rest of the section is as follows. In Section 2.1 we show how to define sets $S \subseteq [n]$ of logarithmic size in $\mathcal{FO}[Q_=]$. In Section 2.2 we show how to express unary relations (subsets) on such sets S in $\mathcal{FO}[Q_=]$. In both sections, all the probabilities and expected values are assumed to be conditioned on $Q_{A,B,C}$, where we assume that A, B, C satisfy Equation (4) (that is, we assume $Q_{A,B,C} \subseteq \mathcal{Q}$). Finally, in Section 2.3 we apply the law of total probability to obtain non-conditioned results.

2.1 Defining Sets of Logarithmic Size

Recall that A, B are the fixed values of the random sets $\mathbf{A}$, $\mathbf{B}$ defined above. We construct subsets of A in terms of the edges between A and B.

▶ **Definition 16.**

1. *For every vertex $x \in A$, let $\mathbf{d}_B(x)$ denote the B-degree of x, which is the number of edges between x and B.*
2. *For every $0 \le k \le |B|$, let $\mathbf{S}_k = \{v \in A : \mathbf{d}_B(v) = k\}$. That is, $\mathbf{S}_k$ is the set of vertices from A with B-degree k.*
3. *For every $x \in A$, let $\mathbf{S}[x] = \mathbf{S}_{\mathbf{d}_B(x)} = \{v \in A : \mathbf{d}_B(v) = \mathbf{d}_B(x)\}$. That is, $\mathbf{S}[x]$ is the set of vertices from A with the same B-degree as x.*

▶ Remark 17. Given a vertex $x \in A$, the statement $v \in \mathbf{S}[x]$ is expressible as a formula in $\mathcal{FO}_= (u_1, u_2, x, v)$:

$$v \in A \wedge Q_= y \, (y \in B \wedge y \sim v, y \in B \wedge y \sim x)$$

where $x \in A$ means $x \sim u_1 \wedge x \sim u_2$ and $y \in B$ means $y \sim u_1 \wedge \neg(y \sim u_2)$.

Importantly, note that the B-degrees $(\mathbf{d}_B(x))_{x \in A}$ are i.i.d. with distribution $\mathrm{Bin}\,(|B|\,, p)$.

▶ **Theorem 18.** *Let $c > 0$ be a constant. Then, with exponentially high probability, there exists $k = k(n)$ such that $0 \le k \le |B|$ and $|\mathbf{S}_k| = (1 + o(1))c \ln n$.*

We can reformulate the statement of theorem more explicitly by recalling the definition of exponentially high probability (see *Notation and conventions* above). Given a positive constant c, the statement is that there exists a sequence $(\varepsilon_n)_{n=1}^{\infty}$ such that, as $n \to \infty$, we have $\varepsilon_n = o(1)$ and

$$\mathbb{P}\left(\exists 0 \le k \le |B| : \left|\frac{|\mathbf{S}_k|}{c \ln n} - 1\right| \ge \varepsilon_n\right) = \exp\left(-n^{\Omega(1)}\right).$$

For the proof of Theorem 18 we use the following normal approximations of binomial probabilities (see [3, Theorems 1.2 and 1.5]).

▷ Claim 19. Let $p \in (0,1)$ and $n \in \mathbb{N}$. Let $\mu = np$ and $\sigma = \sqrt{p(1-p)n}$ be the mean and standard deviation of the binomial distribution $\mathrm{Bin}(n,p)$. Let $0 \le k \le n$ be an integer and let $b(k; n, p) = \mathbb{P}\,(\mathrm{Bin}\,(n,p) = k)$. Write $k = \mu + h$.

1. Assume $\mu \geq 1$ and $\frac{h(1-p)n}{3} \geq 1$. Then
$$b(k;n,p) \leq \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{h^2}{2\sigma^2} + \frac{h}{(1-p)n} + \frac{h^3}{p^2n^2}\right).$$
2. Assume $\mu \geq 1$, $h > 0$ and $k < n$. Then
$$b(k;n,p) \geq \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{h^2}{2\sigma^2} - \frac{h^3}{2(1-p)^2n^2} - \frac{h^4}{3p^3n^3} - \frac{h}{2pn} - \frac{1}{12k} - \frac{1}{12(n-k)}\right).$$

In the proof of Theorem 18 we shall use the following notation:
1. $n_A = |A|$ and $n_B = |B|$.
2. $\mu = pn_B$ and $\sigma = \sqrt{p(1-p)n_B}$.
3. $p_k = \mathbb{P}\left(\mathrm{Bin}\left(n_B, p\right) = k\right)$ for every $0 \leq k \leq n_B$.

▶ **Lemma 20.** *Let $c > 0$ be a constant. For every $n \in \mathbb{N}$ let $t_0 \in \mathbb{R}$ be the unique positive solution of*
$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{t_0^2}{2}\right) = \frac{c \ln n}{n}$$
and let $k_0 = \mu + t_0\sigma$. Then, for every integer $k \in \left[k_0 - n^{1/4}, k_0 + n^{1/4}\right]$ we have $p_k = (1 + o(1)) \frac{c \ln n}{n}$ (where the asymptotic term $o(1)$ is uniform with respect to k).

Proof of Lemma 20. First note that $n_B = \Theta(n)$, $\mu = \Theta(n)$, $\sigma = \Theta\left(n^{1/2}\right)$ and $t_0 = (1 + o(1))\sqrt{\ln n}$. For a given integer $k \in \left[k_0 - n^{1/4}, k_0 + n^{1/4}\right]$, we can write $k = \mu + t\sigma$ for $t = t_0 + O(n^{-1/4})$. Applying Part 1 of Claim 19 (with $h = t\sigma$ and $n = n_B$),
$$\begin{aligned} p_k &\leq \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{t^2}{2}\right) \cdot \exp\left(\frac{t\sigma}{(1-p)n_B} + \frac{t^3\sigma^3}{p^2n_B^2}\right) \\ &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{t_0^2}{2}\right) \cdot \exp\left(O(t_0 n^{-1/4})\right) \cdot \exp\left(O(t_0 n^{-1/2})\right) \\ &= (1 + o(1))\frac{c \ln n}{n}. \end{aligned}$$
Applying Part 2 of Claim 19 (with $h = t\sigma$ and $n = n_B$),
$$\begin{aligned} p_k &\geq \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{t^2}{2}\right) \\ &\quad \cdot \exp\left(-\frac{t^3\sigma^3}{2(1-p)^2b^2} - \frac{t^4\sigma^4}{3p^3b^3} - \frac{t\sigma}{2pb} - \frac{1}{12k} - \frac{1}{12(n-k)}\right) \\ &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{t_0^2}{2}\right) \cdot \exp\left(O(t_0 n^{-1/4})\right) \cdot \exp\left(O(t_0 n^{-1/2})\right) \\ &= (1 + o(1))c\frac{\ln n}{n}. \end{aligned}$$
Overall we have $p_k = (1+o(1))c\frac{\ln n}{n}$, where the $o(1)$ term can be taken to be $O\left((\ln n)^{1/2}n^{-1/4}\right)$ and uniform with respect to k. ◀

Proof of Theorem 18. Note that $\mathbf{s}_k := |\mathbf{S}_k| \sim \mathrm{Bin}\left(n_A, p_k\right)$ for every $0 \leq k \leq n_B$. The variables $\{\mathbf{s}_k\}_{k=0}^{n_B}$ are not independent, since $\sum_{k=0}^{n_B} \mathbf{s}_k = n_A$. However, we can replace them with independent variables by introducing a Poisson process.

Let $\{\mathbf{d}_i\}_{i=1}^{\infty}$ be i.i.d. variables with distribution $\mathrm{Bin}\left(n_B, p\right)$ and let $\mathbf{N} \sim \mathrm{Pois}(n_A)$ be independent of $\{\mathbf{d}_i\}_{i=1}^{\infty}$. These variables define the Poisson process $\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_{\mathbf{N}}$. For every $0 \leq k \leq n_B$ let $\tilde{\mathbf{s}}_k$ count the number of times the value k appears in the process; that is, $\tilde{\mathbf{s}}_k = |\{0 \leq i \leq \mathbf{N} : \mathbf{d}_i = k\}|$. Then the variables $\{\tilde{\mathbf{s}}_k\}_{k=0}^{n_B}$ satisfy the following two properties:

1. The distribution of $\{\tilde{\mathbf{s}}_k\}_{k=0}^{n_B}$ given $\mathbf{N} = n_A$ is identical to the distribution of $\{\mathbf{s}_k\}_{k=0}^{n_B}$.
2. $\{\tilde{\mathbf{s}}_k\}_{k=0}^{n_B}$ are independent and $\tilde{\mathbf{s}}_k \sim \mathrm{Pois}(n_A p_k)$ for every k.

We now apply Lemma 20 with $\frac{c}{p^2}$ as the constant. For every integer $k \in \left[k_0 - n^{1/4}, k_0 + n^{1/4}\right]$ we then have

$$\mathbb{E}\left(\tilde{\mathbf{s}}_k\right) = n_A p_k = (1+o(1))p^2 n \cdot \frac{c}{p^2}\ln n = (1+o(1))c\ln n.$$

From Equation (3) we deduce that there exists a sequence $\varepsilon_n = o(1)$ such that for every integer $k \in \left[k_0 - n^{1/4}, k_0 + n^{1/4}\right]$,

$$\mathbb{P}\left(\tilde{\mathbf{s}}_k \notin [(1-\varepsilon_n)c\ln n, (1+\varepsilon_n)c\ln n]\right) \leq \frac{1}{2}.$$

Write

$$\begin{aligned} I &= [(1-\varepsilon_n)c\ln n, (1+\varepsilon_n)c\ln n], \\ K &= \left[k_0 - n^{1/4}, k_0 + n^{1/4}\right] \cap \mathbb{Z} \end{aligned}$$

for short. Then, from independence,

$$\mathbb{P}\left(\tilde{\mathbf{s}}_k \notin I\ \forall k \in K\right) \leq \left(\frac{1}{2}\right)^{|K|} = \exp\left(-\Theta(n^{1/4})\right).$$

Therefore there exists k such that $\tilde{\mathbf{s}}_k \in I$ with exponentially high probability.

Finally, we condition on the event $\mathbf{N} = n_A$. By Stirling's approximation, $\mathbb{P}\left(\mathbf{N} = n_A\right) = \Theta\left(n_A^{-1/2}\right) = \Theta\left(n^{-1/2}\right)$. Overall

$$\mathbb{P}\left(\mathbf{s}_k \notin I\ \forall k \in K\right) \leq \frac{\mathbb{P}\left(\tilde{\mathbf{s}}_k \notin I\ \forall k \in K\right)}{\mathbb{P}\left(\mathbf{N} = n_A\right)} = \frac{\exp\left(-\Theta(n^{1/4})\right)}{\Theta\left(n^{-1/2}\right)} = \exp\left(-\Theta(n^{1/4})\right).$$

We conclude that, with exponentially high probability, there exists k such that $\mathbf{s}_k \in I$, and so $\mathbf{s}_k = (1+o(1))c\ln n$ as we wanted. ◀

▶ **Corollary 21.** *Let $c > 0$ be a constant. Then, with exponentially high probability, there exists $x \in A$ such that $|\mathbf{S}[x]| = (1+o(1))c\ln n$.*

Proof. Given k such that $|\mathbf{S}_k| = (1+o(1))c\ln n$, pick any $x \in \mathbf{S}_k$ and then $\mathbf{S}_k = \mathbf{S}[x]$. ◀

2.2 Expressing Unary Relations

To express subsets of a given set $S \subseteq A$, we use the edges between S and C.

▶ **Definition 22.** *For a set $S \subseteq A$ and a vertex $z \in C$ let $S_z = \{s \in S : s \sim z\}$. We say that S_z is the subset of S* defined by *z.*

▶ **Proposition 23.** *There exists a positive constant $c_1 < \frac{1}{2}$ such that the following holds with exponentially high probability. For every $x \in A$, if $|\mathbf{S}[x]| \leq 2c_1 \ln n$ then for every subset $T \subseteq \mathbf{S}[x]$ there exists $z \in C$ such that $T = \mathbf{S}[x]_z$.*

The purpose of the condition $c_1 < \frac{1}{2}$ will become apparent in Section 3 (see Lemma 30).

Proof. Let $p_1 = \min\{p, 1-p\}$ and choose c_1 to be a sufficiently small constant such that $c_1 < \frac{1}{2}$ and $\gamma_1 := -2c_1 \ln p_1 < 1$. For this proof only, let us say that a subset $S \subseteq A$ is *good* if for every subset $T \subseteq S$ there exists $z \in C$ such that $T = S_z$.

First, fix $S \subseteq A$ of size $|S| \leq 2c_1 \ln n$ and a subset $T \subseteq S$. For every $z \in C$,

$$\mathbb{P}(T = S_z) = p^{|T|}(1-p)^{|S|-|T|} \geq p_1^{|S|} \geq p_1^{2c_1 \ln n} = n^{-\gamma_1}.$$

Crucially, the subsets of S defined by different vertices $z \in C$ are independently distributed. Thus

$$\mathbb{P}(\forall z \in C.\ T \neq S_z) = \left(1 - p^{|T|}(1-p)^{|S|-|T|}\right)^{|C|} \leq \left(1 - n^{-\gamma}\right)^{|C|} = \exp\left(-\Theta(n^{1-\gamma_1})\right).$$

Taking a union bound over $2^{|S|} = 2^{\Theta(\ln n)}$ possible choices of the subset T, we deduce $\mathbb{P}(S \text{ is not good}) = \exp\left(-n^{\Omega(1)}\right)$.

Finally, for every $x \in A$ we can apply the law of total probability with respect to the possible values of $\mathbf{S}[x]$, and deduce that, with exponentially high probability, $|\mathbf{S}[x]| \leq 2c_1 \ln n$ implies that $\mathbf{S}[x]$ is good. Taking a union bound over $\Theta(n)$ possible choices of x, we get the desired result. ◀

We will also need the following analogous proposition, which will be used to control the upper bound on the size of the definable sets.

▶ **Proposition 24.** *There exists a positive constant c_2 such that $c_2 \geq 2c_1$ and the following holds with probability $1 - o\left(n^{-2}\right)$. For every $x \in A$, if $|\mathbf{S}[x]| \geq c_2 \log_2 n$ then for every $z_1, z_2 \in C$, if $z_1 \neq z_2$ then $\mathbf{S}[x]_{z_1} \neq \mathbf{S}[x]_{z_2}$.*

Again, the purpose of the condition $c_2 \geq 2c_1$ will become apparent in Section 3.

Proof. Let $p_2 = \max\{p, 1-p\}$ and choose c_2 to be a sufficiently large constant such that $c_2 \geq 2c_1$ and $\gamma_2 := -c_2 \ln p_1 > 5$. For this proof only, let us say that a subset $S \subseteq A$ is *good* if for every $z_1, z_2 \in C$, if $z_1 \neq z_2$ then $S_{z_1} \neq S_{z_2}$.

First, fix $S \subseteq A$ of size $|S| \geq c_2 \ln n$. For every pair of distinct vertices $z_1, z_2 \in C$,

$$\mathbb{P}(S_{z_1} = S_{z_2}) \leq p_2^{|S|} \leq p_2^{c_2 \ln n} = n^{c_2 \ln p_2} = n^{-\gamma_2}.$$

Taking a union bound over $\Theta(n^2)$ choices of $z_1 \neq z_2$, we get

$$\mathbb{P}(S \text{ is not good}) = O(n^{2-\gamma_2}).$$

Finally, for every $x \in A$ we can apply the law of total probability with respect to the possible values of $\mathbf{S}[x]$ and deduce that, with probability $1 - O(n^{2-\gamma_2})$, $|\mathbf{S}[x]| \geq c_2 \ln n$ implies that $\mathbf{S}[x]$ is good. Taking a union bound over $\Theta(n)$ possible choices of x, and recalling that $n^{3-\gamma_2} = o(n^{-2})$ by definition, we get the desired result. ◀

2.3 Non-Conditioned Results

Finally, we lose the conditioning on the events $Q_{A,B,C}$ which fix the values of $\mathbf{A}, \mathbf{B}, \mathbf{C}$. Note that we still have dependency on the choice of u_1, u_2; we will quantify over u_1, u_2 in the next section. The following theorem summarizes all the probabilistic results proved in this section.

▶ **Theorem 25.** *There exist positive constants c_1, c_2 with $c_1 < \frac{1}{2}$ and $c_2 \geq 2c_1$ and sequences $\delta_n = o(1)$ and $\varepsilon_n = o(1)$ such that $\mathbb{P}(\Gamma(u_1, u_2)) = 1 - o(n^{-2})$, where $\Gamma(u_1, u_2)$ is the event that:*

1. $\frac{|\mathbf{A}|}{p^2 n}, \frac{|\mathbf{B}|}{p(1-p)n}, \frac{|\mathbf{C}|}{p(1-p)n} \in [1-\delta_n, 1+\delta_n]$ *(we denote this event by $\mathcal{Q}$);*
2. *There exists $x \in \mathbf{A}$ such that*

$$|\mathbf{S}[x]| \in \left[(1-\varepsilon_n)\frac{3}{2}c_1 \ln n, (1+\varepsilon_n)\frac{3}{2}c_1 \ln n\right];$$

3. *For every $x \in \mathbf{A}$, if $|\mathbf{S}[x]| \leq 2c_1 \ln n$ then for every $T \subseteq \mathbf{S}[x]$ there exists $z \in \mathbf{C}$ such that $T = \mathbf{S}[x]_z$; and*
4. *For every $x \in \mathbf{A}$, if $|\mathbf{S}[x]| \geq c_2 \ln n$ then for every $z_1, z_2 \in \mathbf{C}$, if $z_1 \neq z_2$ then $\mathbf{S}[x]_{z_1} \neq \mathbf{S}[x]_{z_2}$.*

Proof. Let $\delta_n = o(1)$ be the sequence from Equation (4). In the previous sections, we conditioned all probabilities on $Q_{A,B,C}$ where $Q_{A,B,C} \subseteq \mathcal{Q}$. However, note that the proofs of Theorem 18 and Propositions 23, 24 do not depend on the specific values A, B, C, but only on the fact that they satisfy Equation (4). Therefore, they also hold when the conditioning is on the event $\mathcal{Q}$.

Now, let c_1, c_2 be the constants from Propositions 23 and 24 (respectively) and let $(\varepsilon_n)_{n=1}^{\infty}$ be the sequence from Theorem 18 for $c = \frac{3}{2}c_1$. Define $\Gamma(u_1, u_2)$ as above. Then

$$\begin{aligned}\mathbb{P}\left(\Gamma(u_1,u_2)\right) &= \mathbb{P}\left(\Gamma(u_1,u_2) \mid \mathcal{Q}\right)\mathbb{P}\left(\mathcal{Q}\right) \\ &\geq \left(1-o(n^{-2})\right)\left(1-\exp\left(-n^{\Omega(1)}\right)\right) = 1-o(n^{-2}).\end{aligned}$$

That concludes the proof. ◀

▶ Remark 26. Note that, due to symmetry considerations, $\mathbb{P}\left(\Gamma(u_1, u_2)\right)$ does not depend on the choice of u_1, u_2.

3 Proof of the Main Theorem

In this section we complete the proof of Theorem 14. We begin with a sequence of short lemmas, which build upon our previous results. Once again, we fix a constant $p \in (0,1)$ and consider a binomial random graph $\mathbf{G}_n \sim G(n,p)$. Probabilities are now non-conditioned.

We begin with a refinement of Theorem 9.

▶ **Lemma 27** (Kaufmann-Shelah)**.** *There exists a sentence* $\text{Encode}^* \in \mathcal{MSO}$ *such that:*
1. $\lim_{n\to\infty} \mathbb{P}\left(\mathbf{G}_n \models \text{Encode}^*\right) = 1$.
2. *For every sentence $\varphi \in \mathcal{SO}[\text{Arith}]$ there exists a sentence $\varphi^* \in \mathcal{MSO}$ such that, for every $n \in \mathbb{N}$ and graph $G = ([n], E)$ with $G \models \text{Encode}^*$, we have $G \models \varphi^* \iff [\sqrt{n}] \models \varphi$.*

Proof. This follows from Theorem 1 and the closing remark of [23]. ◀

Intuitively, the sentence Encode^* asserts the existence of $\mathcal{MSO}$-formulas expressing a structure of addition and multiplication on the vertices, a necessary ingredient for converting any $\varphi \in \mathcal{SO}[\text{Arith}]$ into $\varphi^* \in \mathcal{MSO}$. The basic structure of the sentence Encode^* is given in Theorem 1 of [23].

▶ **Lemma 28.** *For every sentence $\varphi^* \in \mathcal{MSO}$ there exists a formula $\varphi^{**}(u_1, u_2, x) \in \mathcal{FO}_=(u_1, u_2, x)$ such that the following holds. Given the event $\Gamma(u_1, u_2)$, for every $x \in \mathbf{A}$ with $|\mathbf{S}[x]| \leq 2c_1 \ln n$ we have*

$$\mathbf{G}_n \models \varphi^{**}(u_1, u_2, x) \iff \mathbf{G}_n[\mathbf{S}[x]] \models \varphi^*.$$

Proof. Given φ^*, define $\varphi^{**}(u_1, u_2, x)$ as follows:

- Restrict quantification to $\mathbf{S}[x]$: replace every $\forall v\,(\theta)$ with $\forall v\,(v \in \mathbf{S}[x] \to \theta)$ and every $\exists v\,(\theta)$ with $\exists v\,(v \in \mathbf{S}[x] \wedge \theta)$. Recall that the statement $v \in \mathbf{S}[x]$ is expressible as a formula in $\mathcal{FO}_=(u_1, u_2, x, v)$ (see Remark 17).
- Convert unary relations: for every unary relation R introduced by ψ, replace $\exists R\,(\theta)$ with $\exists z_R(z_R \in \mathbf{C} \wedge \theta)$ where z_R is a new variable symbol, and also replace every $R(v)$ with $v \sim z_R$. Similarly handle $\forall R\,(\theta)$. Recall that the statement $z \in \mathbf{C}$ is expressible the formula $z \not\sim u_1 \wedge z \sim u_2$, which is in $\mathcal{FO}(u_1, u_2, z)$

Given the event $\Gamma(u_1, u_2)$, for every $x \in \mathbf{A}$ with $|\mathbf{S}[x]| \leq 2c_1 \ln n$, we know that every subset of $\mathbf{S}[x]$ is defined by some $z \in \mathbf{C}$ (see Part 3 of Theorem 25). Therefore

$$\mathbf{G}_n \models \varphi^{**}(u_1, u_2, x) \iff \mathbf{G}_n[\mathbf{S}[x]] \models \varphi^*$$

as we wanted. ◀

Next, we introduce a formula for upper-bounding the size of definable sets.

▶ **Definition 29.** *Given a choice of $u_1, u_2 \in [n]$ and $x \in \mathbf{A}$, we say that $\mathbf{S}[x]$ is* pseudo-logarithmic *if there exist $z_1, z_2 \in \mathbf{C}$ such that $z_1 \neq z_2$ but $\mathbf{S}[x]_{z_1} = \mathbf{S}[x]_{z_2}$.*

Note that this property is expressible as a formula in $\mathcal{FO}_=(u_1, u_2, x)$, given by

$$\exists z_1 \exists z_2\,(z_1 \in \mathbf{C} \wedge z_2 \in \mathbf{C} \wedge z_1 \neq z_2 \wedge \mathbf{S}[x]_{z_1} = \mathbf{S}[x]_{z_2})\,,$$

where $\mathbf{S}[x]_{z_1} = \mathbf{S}[x]_{z_2}$ is the formula $\forall y\,(y \in \mathbf{S}[x] \to (y \sim z_1 \leftrightarrow y \sim z_2))$.

▶ **Lemma 30.** *Given the event $\Gamma(u_1, u_2)$, for every $x \in \mathbf{A}$,*

1. *If $\mathbf{S}[x]$ is pseudo-logarithmic then $|\mathbf{S}[x]| \leq c_2 \ln n$.*
2. *If $|\mathbf{S}[x]| \leq 2c_1 \ln n$ then $\mathbf{S}[x]$ is pseudo-logarithmic.*

Proof. Part 1 follows directly from the definition of $\Gamma(u_1, u_2)$ (see part 4 of Theorem 25). Part 2 follows from the pigeonhole principle. Indeed, let $S = \mathbf{S}[x]$ and assume $|S| \leq 2c_1 \ln n$. Then the number of subsets of S is $2^{|S|} \leq 2^{2c_1 \ln n} = n^{2c_1 \ln 2}$. Recall that $c_1 < \frac{1}{2}$, so $2^{|S|} \leq n^{\ln 2} = o(n)$. However, given $\Gamma(u_1, u_2)$ we have $|\mathbf{C}| = \Theta(n)$. From the pigeonhole principle there must exist $z_1, z_2 \in \mathbf{C}$ such that $z_1 \neq z_2$ but $S_{z_1} = S_{z_2}$, hence S is pseudo-logarithmic. ◀

Finally, we introduce a formula for comparing sizes of definable sets.

▶ **Definition 31.** *Given a choice of $u_1, u_2 \in [n]$ and two vertices $x, x' \in \mathbf{A}$, we say that $\mathbf{S}[x]$ is* pseudo-smaller *than $\mathbf{S}[x']$ if there exists $z \in \mathbf{C}$ such that $|\mathbf{S}[x']_z| = |\mathbf{S}[x]|$.*

Note that this property is expressible as a formula in $FO_=(u_1, u_2, x, x')$, since belonging to $\mathbf{S}[x]$ and to $\mathbf{S}[x']_z$ and the equicardinality condition $|\mathbf{S}[x']_z| = |\mathbf{S}[x]|$ are all expressible in $\mathcal{FO}_=$.

▶ **Lemma 32.** *Given the event $\Gamma(u_1, u_2)$, for every $x, x' \in \mathbf{A}$,*

1. *If $\mathbf{S}[x]$ is pseudo-smaller than $\mathbf{S}[x']$ then $|\mathbf{S}[x]| \leq |\mathbf{S}[x']|$.*
2. *If $|\mathbf{S}[x]| \leq |\mathbf{S}[x']| \leq 2c_1 \ln n$ then $\mathbf{S}[x]$ is pseudo-smaller than $\mathbf{S}[x']$.*

Proof. Part 1 follows from the definition of pseudo-smaller (in fact, it is true deterministically). Part 2 follows from the definition of $\Gamma(u_1, u_2)$ (see Part 3 of Theorem 25). ◀

We are now ready to prove the main theorem. In the proof, we use the notation $\mathbf{E}(X, Y)$ for the set of edges in $\mathbf{G}_n$ between two disjoint sets of vertices $X, Y \subseteq [n]$.

Proof of Theorem 14. We prove that $\mathcal{FO}_=$ can interpret a segment of arithmetic with constants $\sqrt{c_1}, \sqrt{c_2}$ and scaling function $f(n) = \left\lfloor \sqrt{\ln n} \right\rfloor$, where c_1, c_2 are the constants from Theorem 25. That is, for every sentence $\varphi \in \mathcal{SO}[\text{Arith}]$ we construct a sentence $\psi \in \mathcal{FO}_=$ such that the following holds. Given a sequence $\{n_k\}_{k=1}^{\infty}$ with $\lim_{k\to\infty} n_k = \infty$ such that φ is constant on $\left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$ for every k, we have

$$\lim_{k\to\infty} \mathbb{P}\left(G(n_k,p) \models \psi \iff \varphi \text{ holds in } \left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]\right) = 1. \tag{5}$$

From now on we fix a sentence $\varphi \in \mathcal{SO}[\text{Arith}]$. First, we apply Lemma 27 to obtain a sentence $\varphi^* \in \mathcal{MSO}$ such that, for every graph $G = ([n], E)$ with $G \models \text{Encode}^*$, we have $G \models \varphi^* \iff [\sqrt{n}] \models \varphi$. Second, we apply Lemma 28 to the $\mathcal{MSO}$-sentences φ^* and Encode^* (from Lemma 27) to obtain formulas $\varphi^{**}(u_1,u_2,x), \text{Encode}^{**}(u_1,u_2,x) \in \mathcal{FO}_=(u_1,u_2,x)$ such that, given the event $\Gamma(u_1,u_2)$, for every $x \in \mathbf{A}$ with $|\mathbf{S}[x]| \le 2c_1 \ln n$ we have

$$\mathbf{G}_n \models \varphi^{**}(u_1,u_2,x) \iff \mathbf{G}_n[\mathbf{S}[x]] \models \varphi^*, \tag{6}$$

$$\mathbf{G}_n \models \text{Encode}^{**}(u_1,u_2,x) \iff \mathbf{G}_n[\mathbf{S}[x]] \models \text{Encode}^*. \tag{7}$$

Now define ψ as the sentence claiming the existence of two vertices u_1, u_2 and a vertex $x \in \mathbf{A}$ such that:

1. $\mathbf{S}[x]$ is pseudo-logarithmic and $\mathbf{G}_n[\mathbf{S}[x]] \models \text{Encode}^*$.
2. If $x' \in \mathbf{A}$ is another vertex such that $\mathbf{S}[x']$ is pseudo-logarithmic and $\mathbf{G}_n[\mathbf{S}[x]] \models \text{Encode}^*$, then $\mathbf{S}[x]$ is not pseudo-smaller than $\mathbf{S}[x']$.
3. $\mathbf{G}_n[\mathbf{S}[x]] \models \varphi^*$.

From Lemmas 30, 32 and (6), (7) above, ψ is indeed expressible as a sentence in $\mathcal{FO}_=$. It remains to verify (5).

Let $\{n_k\}_{k=1}^{\infty}$ with $\lim_{k\to\infty} n_k = \infty$ such that φ is constant on $\left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$ for every k. There are two cases to consider.

Case 1: φ holds in $\left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$. Fix two vertices u_1, u_2 arbitrarily. We show that, with high probability, there exists a vertex $x \in \mathbf{A}$ which satisfies the three parts of ψ (along with u_1, u_2).

First, we know that $\mathbb{P}(\Gamma(u_1,u_2)) = 1 - o(1)$, so from now on we may assume that $\Gamma(u_1,u_2)$ holds. Recall that the event $\Gamma(u_1,u_2)$ guarantees a vertex $x' \in \mathbf{A}$ such that

$$|\mathbf{S}[x']| \in \left[(1-\varepsilon_n)\frac{3}{2}c_1 \ln n_k, (1+\varepsilon_n)\frac{3}{2}c_1 \ln n_k\right].$$

Fix such a vertex x'. Since $c_2 \ge 2c_1 \ge (1+\varepsilon_n)\frac{3}{2}c_1$, Lemma 32 implies that $\mathbf{S}[x']$ is pseudo-logarithmic. Let us show that $\mathbf{G}_{n_k}[\mathbf{S}[x']] \models \text{Encode}^*$ with high probability.

Condition on the values A, B, C of the sets $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and on the edge sets $\mathbf{E}(A,B)$ and $\mathbf{E}(A,C)$. These values determine the value S of the set $\mathbf{S}[x']$. Crucially, the induced subgraph $\mathbf{G}_{n_k}[S]$ depends only on the edge set $\mathbf{E}(A)$, and so, given the last conditioning, $\mathbf{G}_{n_k}[S]$ is still binomially distributed with vertex set S and edge probability p. From Lemma 27 we get that, given the last conditioning, $\mathbf{G}_{n_k}[S] \models \text{Encode}^*$ with high probability. Now apply the law of total probability over the possible values of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{E}(A,B), \mathbf{E}(A,C)$ to conclude that $\mathbf{G}_{n_k}[\mathbf{S}[x]] \models \varphi^*$ with high probability.

Next, among all vertices $x \in \mathbf{A}$ such that $\mathbf{S}[x]$ is pseudo-logarithmic and $\mathbf{G}_{n_k}[\mathbf{S}[x]] \models \text{Encode}^*$, pick x such that $|\mathbf{S}[x]|$ is maximal. By definition, x satisfies Part 1 and Part 2 of ψ. We show that it also satisfies Part 3. Since $\mathbf{S}[x]$ is pseudo-logarithmic, Lemma 30 implies $|\mathbf{S}[x]| \le c_2 \ln n_k$. We also know that

$$|\mathbf{S}[x]| \ge |\mathbf{S}[x']| \ge (1-\varepsilon_n)\frac{3}{2}c_1 \ln n_k \ge c_1 \ln n_k + 1.$$

Letting $\mathbf{s} = |\mathbf{S}[x]|$, we deduce $\lfloor\sqrt{\mathbf{s}}\rfloor \in \left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$. From the assumption of Case 1, $[\lfloor\sqrt{\mathbf{s}}\rfloor] \models \varphi$. Since $\mathbf{G}_{n_k}[\mathbf{S}[x']] \models \text{Encode}^*$, Lemma 27 implies $\mathbf{G}_{n_k}[\mathbf{S}[x']] \models \varphi^*$ as we wanted.

Case 2: $\neg\varphi$ holds in $\left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$. We need to prove that with high probability, for every u_1, u_2, there is no vertex x that satisfies all three parts of ψ. Let $\Gamma = \bigcap_{u_1,u_2} \Gamma(u_1, u_2)$. Theorem 25 shows that $\mathbb{P}(\Gamma(u_1, u_2)) = 1 - o(n^{-2})$ for every u_1, u_2. Taking a union bound the over $\Theta(n^2)$ pairs u_1, u_2 we get $\mathbb{P}(\Gamma) = 1 - o(1)$. So from now on we may assume that Γ holds.

Assume that u_1, u_2, x are vertices such that Part 1 and Part 2 of ψ hold and let us show that Part 3 does not hold. Again, Γ guarantees a vertex $x' \in \mathbf{A}$ such that

$$|\mathbf{S}[x']| \in \left[(1-\varepsilon_n)\frac{3}{2}c_1 \ln n_k, (1+\varepsilon_n)\frac{3}{2}c_1 \ln n_k\right].$$

We prove $|\mathbf{S}[x]| \geq |\mathbf{S}[x']|$ by contradiction. Indeed, otherwise we have

$$|\mathbf{S}[x]| \leq |\mathbf{S}[x']| \leq (1+\varepsilon_n)\frac{3}{2}c_1 \ln n_k \leq 2c_1 \ln n_k,$$

and from Lemma 30 we get that $\mathbf{S}[x]$ is pseudo-smaller than $\mathbf{S}[x']$. But that contradicts Part 2 of ψ. Therefore

$$|\mathbf{S}[x]| \geq |\mathbf{S}[x']| \geq (1-\varepsilon_n)\frac{3}{2}c_1 \ln n_k \geq c_1 \ln n_k + 1.$$

Moreover, $\mathbf{S}[x]$ is pseudo-logarithmic, so Lemma 30 implies $|\mathbf{S}[x]| \leq c_2 \ln n$. As before, letting $\mathbf{s} = |\mathbf{S}[x]|$, we get $\lfloor\sqrt{\mathbf{s}}\rfloor \in \left[\sqrt{c_1 \ln n_k}, \sqrt{c_2 \ln n_k}\right]$. From the assumption of Case 2, $[\lfloor\sqrt{\mathbf{s}}\rfloor] \not\models \varphi$. Since $\mathbf{G}_{n_k}[\mathbf{S}[x']] \models \text{Encode}^*$, Lemma 27 implies $\mathbf{G}_{n_k}[\mathbf{S}[x']] \not\models \varphi^*$, as we wanted. ◀

References

1 Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Wiley Publishing, 4th edition, 2016.

2 Andreas Blass and Frank Harary. Properties of almost all graphs and complexes. *Journal of Graph Theory*, 3(3):225–240, 1979. doi:10.1002/JGT.3190030305.

3 Béla Bollobás. *Random Graphs*. Cambridge University Press, 2001.

4 Kevin J. Compton. 0-1 laws in logic and combinatorics. In Rival Ivan, editor, *Algorithms and Order*, volume 255 of *Advanced Study Institute Series C: Mathematical and Physical Sciences*, pages 353–383. Kluwer Academic Publishers, Dordrecht, 1989.

5 Anuj Dawar and Erich Grädel. Generalized quantifiers and 0-1 laws. In *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science (LICS 1995)*, pages 54–64. IEEE Computer Society Press, June 1995. doi:10.1109/LICS.1995.523244.

6 Anuj Dawar and Erich Grädel. Properties of almost all graphs and generalized quantifiers. *Fundam. Inform.*, 98(4):351–372, 2010. doi:10.3233/FI-2010-232.

7 Heinz-Dieter Ebbinghaus. Extended logics: The general framework. In Jon Barwise and Solomon Feferman, editors, *Model-Theoretic Logics*, chapter II, pages 25–76. Association for Symbolic Logic, September 1985.

8 Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite Model Theory*. Springer Berlin, Heidelberg, second edition, 2006.

9 Paul Erdős and Alfréd Rényi. On random graphs, I. *Publicationes Mathematicae*, 6:290–297, 1959.

10 Paul Erdős and Alfréd Rényi. On the evolution of random graphs. In *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, number 5 in Acta Math. Acad. Sci. Hungar., pages 17–61, 1960.

11 Ronald Fagin. Probabilities on finite models. *The Journal of Symbolic Logic*, 41(1):50–58, March 1976. doi:10.1017/S0022481200051756.

12 Guy Fayolle, Stéphane Grumbach, and Christophe Tollu. Asymptotic probabilities of languages with generalized quantifiers. In *Proceedings Eighth Annual IEEE Symposium on Logic in Computer Science*, pages 199–207, 1993. doi:10.1109/LICS.1993.287587.

13 Yu. V. Glebskiĭ, D. I. Kogan, M. I. Liogon'kiĭ, and V. A. Talanov. Range and degree of realizability of formulas in the restricted predicate calculus. *Cybernetics and Systems Analysis*, 5(2):142–154, March 1969. (Russian original: Kibernetika 5(2):17–27, March-April 1969).

14 Simi Haber. Generalized quantifiers for simple graph properties in random graphs. , Preprint.

15 Simi Haber. Arithmetization for first order logic augmented with perfect matching. , Submitted.

16 Simi Haber and Saharon Shelah. Random graphs and Lindström quantifiers for natural graph properties. *Accepted, Annales Univ. Sci. Budapest., Sect. Math*, 2024+. HbSh:986 in Shelah's archive.

17 Klaus Härtig. Über einen quantifikator mit zwei wirk ungsbereichen. In Laszlo Kalmar, editor, *Colloquium on the Foundations of Mathematics, Mathematical Machines and their Applications*, pages 31–36. Akademiai Kiado, Budapest, September 1962.

18 Heinrich Herre, Michał Krynicki, Alexandr Pinus, and Jouko Väänänen. The Härtig quantifier: a survey. *Journal of Symbolic Logic*, 56(4):1153–1183, December 1991. doi:10.2307/2275466.

19 Svante Janson, Tomasz Łuczak, and Andrzej Ruciński. *Random Graphs*. John Wiley & Sons, 2000.

20 Risto Kaila. On probabilistic elimination of generalized quantifiers. *Random Struct. Algorithms*, 19(1):1–36, 2001. doi:10.1002/RSA.1016.

21 Risto Kaila. Convergence laws for very sparse random structures with generalized quantifiers. *Math. Log. Q.*, 48(2):301–320, 2002. doi:10.1002/1521-3870(200202)48:2\%3C301::AID-MALQ301\%3E3.0.CO;2-Z.

22 Risto Kaila. On almost sure elimination of numerical quantifiers. *J. Log. Comput.*, 13(2):273–285, 2003. doi:10.1093/LOGCOM/13.2.273.

23 Matt Kaufmann and Saharon Shelah. On random models of finite power and monadic logic. *Discrete Mathematics*, 54(3):285–293, 1985. doi:10.1016/0012-365X(85)90112-8.

24 H. Jerome Keisler and Wafik Boulos Lotfallah. Almost everywhere elimination of probability quantifiers. *Journal of Symbolic Logic*, 74(4):1121–1142, 2009. doi:10.2178/JSL/1254748683.

25 Phokion G. Kolaitis and Swastik Kopparty. Random graphs and the parity quantifier. *Journal of the Association for Computing Machinery*, 60(5):1–34, October 2013. doi:10.1145/2528402.

26 Per Lindström. First order predicate logic with generalized quantifiers. *Theoria*, 32(3):186–195, 1966.

27 Per Lindström. On extensions of elementary logic. *Theoria*, 35(1):1–11, 1969.

28 M. H. Löb. Meeting of the association for symbolic logic, leeds 1962. *The Journal of Symbolic Logic*, 27(3):373–382, 1962. First abstract: Plurality-quantification by Nicholas Rescher. doi:10.1017/S0022481200118742.

29 Peter Winkler. Random structures and zero-one laws. In Sauer N. W., Woodrow R. E., and Sands B., editors, *Finite and Infinite Combinatorics in Sets and Logic*, Advanced Science Institutes, pages 399–420. Kluwer Academic Publishers, Dordrecht, 1993.

Computational Complexity of the Weisfeiler-Leman Dimension

Moritz Lichter
RWTH Aachen University, Germany

Simon Raßmann
TU Darmstadt, Germany

Pascal Schweitzer
TU Darmstadt, Germany

Abstract

The Weisfeiler-Leman dimension of a graph G is the least number k such that the k-dimensional Weisfeiler-Leman algorithm distinguishes G from every other non-isomorphic graph, or equivalently, the least k such that G is definable in $(k+1)$-variable first-order logic with counting. The dimension is a standard measure of the descriptive or structural complexity of a graph and recently finds various applications in particular in the context of machine learning. This paper studies the complexity of computing the Weisfeiler-Leman dimension. We observe that deciding whether the Weisfeiler-Leman dimension of G is at most k is NP-hard, even if G is restricted to have 4-bounded color classes. For each fixed $k \geq 2$, we give a polynomial-time algorithm that decides whether the Weisfeiler-Leman dimension of a given graph with 5-bounded color classes is at most k. Moreover, we show that for these bounds on the color classes, this is optimal because the problem is P-hard under logspace-uniform AC_0-reductions. Furthermore, for each larger bound c on the color classes and each fixed $k \geq 2$, we provide a polynomial-time decision algorithm for the abelian case, that is, for structures of which each color class has an abelian automorphism group.

While the graph classes we consider may seem quite restrictive, graphs with 4-bounded abelian colors include CFI-graphs and multipedes, which form the basis of almost all known hard instances and lower bounds related to the Weisfeiler-Leman algorithm.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Problems, reductions and completeness; Theory of computation → Complexity theory and logic

Keywords and phrases Weisfeiler-Leman algorithm, dimension, complexity, coherent configurations

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.13

Related Version *Full Version*: `https://arxiv.org/abs/2402.11531` [35]

Funding The research leading to these results has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (EngageS: grant agreement No. 820148).
Moritz Lichter: The research of this author has received further funding by the European Union (ERC, SymSim, 101054974).

1 Introduction

The *Weisfeiler-Leman algorithm* is a simple combinatorial procedure studied in the context of the graph isomorphism problem. For every $k \geq 1$, the algorithm has a k-dimensional variant, k-WL for short, that colors k-tuples of vertices according to how they structurally sit inside the whole graph: if two tuples get different colors, they cannot be mapped onto each other by an automorphism of the graph (while the converse is not always true). The 1-dimensional algorithm, which is also known as *color refinement*, starts by coloring each

vertex according to its degree, and then repeatedly refines this coloring by including into each vertex color the multisets of colors of its neighbors. The k-dimensional variant generalizes this idea and colors k-tuples of vertices instead of single vertices [43, 11].

The Weisfeiler-Leman algorithm plays an important role in both theoretic and practical approaches to the graph isomorphism problem, but is also related to a plethora of seemingly unrelated areas: to finite model theory and descriptive complexity via the correspondence of k-WL to $(k+1)$-variable first-order logic with counting [11, 27], to machine learning via a correspondence to the expressive power of (higher-dimensional) graph neural networks [37], to the Sherali-Adams hierarchy in combinatorial optimization [3, 25], and to homomorphism counts from treewidth-k graphs [14]. On the side of practical graph isomorphism, the color refinement procedure is a basic building block of the so-called *individualization-refinement framework*, which is the basis of almost every modern practical solver for the graph isomorphism problem [36, 28, 29, 1]. On the side of theoretical graph isomorphism, Babai's quasipolynomial-time algorithm for the graph isomorphism problem [4] uses a combination of group-theoretic techniques and a logarithmic-dimensional Weisfeiler-Leman algorithm.

The Weisfeiler-Leman algorithm is a powerful algorithm for distinguishing non-isomorphic graphs on its own. For every k, k-WL can be used as an incomplete polynomial-time isomorphism test: if the multiset of colors of k-tuples of two graphs G and H differ, then G and H cannot be isomorphic. In this case, k-WL distinguishes G and H, otherwise G and H are k-WL-equivalent. For a given graph G, we say that the k-dimensional Weisfeiler-Leman algorithm k-WL *identifies* G if it distinguishes G from every non-isomorphic graph. The smallest such k is known as the *Weisfeiler-Leman dimension of* G [21].

It is known that almost all graphs have Weisfeiler-Leman dimension 1 [5]. However, color refinement fails spectacularly on regular graphs, where it always returns the monochromatic coloring. For these, it is known that 2-WL identifies almost all regular graphs [10, 32]. In contrast to these positive results, for every k there is some graph G (even of order linear in k, of maximum degree 3, and with 4-bounded abelian color classes, i.e., such that no more than 4 vertices can share the same vertex-color, and every color class induces a graph with abelian automorphism group) that is not identified by k-WL [11]. These so-called CFI-graphs have high Weisfeiler-Leman dimension and are thus hard instances for combinatorial approaches to the graph isomorphism problem.

The situation changes for restricted classes of graphs. If the Weisfeiler-Leman dimension over some class of graphs is bounded by k, then the k-WL correctly decides isomorphism over this class. And since k-WL can be implemented in polynomial time $O(n^{k+1}\log n)$ [27], this puts graph isomorphism over such classes into polynomial time. Examples of graph classes with bounded Weisfeiler-Leman dimension include graphs of bounded tree-width [23], graphs of bounded rank-width [24], graphs with 3-bounded color classes [27], planar graphs [30], and more generally every non-trivial minor-closed graph class [20].

In this paper, we study the computational complexity of computing the Weisfeiler-Leman dimension. We call the problem of deciding whether the Weisfeiler-Leman dimension of a given graph is at most k the k-WL-*identification problem*. For upper complexity bounds, non-identification of a graph G can be witnessed by providing a graph H that is not distinguished from G by k-WL but is also not isomorphic to G. As the latter can be checked in co-NP, this places the identification problem into the class Π_2^{P} of the polynomial hierarchy. If the graph isomorphism problem is solvable in polynomial time, this complexity bound collapses to co-NP. However, there is no apparent reason why the identification problem should not be polynomial-time decidable.

On the side of lower complexity bounds, the 1-WL-identification problem is complete for polynomial time under uniform reductions in the circuit complexity class AC_0 [31, 2]. Hardness of the 1-WL-identification problem does, however, not easily imply any hardness results for the k-WL-identification problem for higher values of k. Indeed, no hardness results are known for $k \geq 2$. The 2-WL-identification problem in particular includes the problem of deciding whether a given strongly regular graph is determined up to isomorphism by its parameters, which is a baffling problem from classic combinatorics far beyond our current knowledge. To understand the difficulties of the k-WL-identification problem better, we can again consider classes of graphs. On every class of graphs with bounded color classes, graph isomorphism is solvable in polynomial time [6, 16], which puts the identification problem over this class into $\mathsf{co\text{-}NP}$ for every $k \geq 2$. Graphs with 3-bounded color classes are identified by 2-WL [27], which makes their identification problem trivial. As shown by the CFI-graphs [11], this is no longer true for graphs with 4-bounded color classes. Nevertheless, as shown by Fuhlbrück, Köbler, and Verbitsky, identification of graphs with 5-bounded color classes by 2-WL is efficiently decidable [15]. For higher dimensions or bounds on the color classes essentially nothing is known.

Contribution. We extend the results of [15] from 2-WL to k-WL and give a polynomial-time algorithm deciding whether a graph with 5-bounded color classes is identified by k-WL:

▶ **Theorem 1.** *For every k, there is an algorithm that decides the k-WL-identification problem for vertex- and edge-colored, directed graphs with 5-bounded color classes in time $O_k(n^{O(k)})$. If such a graph G is not identified by k-WL, the algorithm provides a witness for this, i.e., a graph H that is not isomorphic to G and not distinguished from G by k-WL.*

Via the correspondence of k-WL to $(k+1)$-variable counting logic, Theorem 1 implies that definability of graphs with 5-bounded color classes in this logic is decidable in polynomial time. While the restriction to 5-bounded color classes may seem stark, almost all known hardness results and lower bounds for the Weisfeiler-Leman algorithm remain true for graphs with bounded color classes and in most cases even 4-bounded color classes suffice [19, 13, 40, 39, 38].

Towards generalizing Theorem 1 to arbitrary relational structures and larger color classes, we consider structures with abelian color classes, i.e., structures of which each color class induces a structure with an abelian automorphism group. Such structures were previously considered in the context of descriptive complexity theory [44], and include both CFI-graphs [11] and multipedes [39, 38] over ordered base graphs, which form the basis of all known constructions of graphs with high Weisfeiler-Leman dimension. For many cases in descriptive complexity theory, restricting to 4-bounded abelian color classes is sufficient, but in some cases larger (but still abelian) color classes are required [26, 18, 33, 34]. For such structures, we obtain a polynomial-time algorithm as before:

▶ **Theorem 2.** *For every $k \in \mathbb{N}$ and $c, r \leq k$, there is an algorithm that decides the k-WL-identification problem for r-ary relational structures with c-bounded abelian color classes in time $O_k(n^{O(k)})$. If such a structure $\mathfrak{A}$ is not identified by k-WL, the algorithm provides a witness for this, i.e., a second structure $\mathfrak{B}$ that is not isomorphic to $\mathfrak{A}$ and not distinguished from $\mathfrak{A}$ by k-WL.*

On the side of hardness results, we first prove that when the dimension k is part of the input, the identification problem is NP-hard. Note that a similar result was recently independently observed by Seppelt [42].

▶ **Theorem 3.** *The problem of deciding, given a graph G and a natural number k, whether the Weisfeiler-Leman dimension of G is at most k is NP-hard, both over uncolored simple graphs, and over simple graphs with 4-bounded color classes.*

Furthermore, we extend the P-hardness results for 1-WL [2] to arbitrary k and prove that, when k is fixed, the k-WL-identification problem is hard for polynomial time:

▶ **Theorem 4.** *For every $k \geq 1$, the k-WL-identification problem is P-hard under uniform AC_0-reductions over both uncolored simple graphs, and simple graphs with 4-bounded abelian color classes.*

Techniques. To prove Theorem 1, we exploit the close connection between the coloring computed by k-WL and certain combinatorial structures called k-ary coherent configurations. These structures come with two notions of isomorphisms, algebraic ones and combinatorial ones. Similarly to [15], we reduce the k-WL-identification problem to the separability problem for k-ary coherent configurations, that is, to decide whether algebraic and combinatorial isomorphisms for a given k-ary coherent configuration coincide. We make two crucial observations: First, we show that the k-ary coherent configurations obtained from graphs are fully determined by their underlying 2-ary configurations. We call such configurations 2-induced. Second, we reduce the separability problem for arbitrary k-ary coherent configurations to the same problem on the structurally simpler class of star-free k-ary coherent configurations. Combining both observations, we show that two 2-induced, star-free k-ary coherent configurations obtained from k-WL-equivalent graphs must be isomorphic. Given such a k-ary coherent configuration obtained from a graph, it thus suffices to decide whether there is another non-isomorphic graph yielding the same configuration. Finally, we solve this problem by encoding it into the graph isomorphism problem for structures with bounded color classes, which is polynomial-time solvable [6, 16].

The main obstacle to generalize Theorem 1 to larger color classes or relational structures of higher arity is the existence of k-WL-equivalent structures that yield non-isomorphic star-free k-ary coherent configurations, which greatly increases the space of possibly equivalent but non-isomorphic structures. To make up for this, we consider structures with abelian color classes. Using both the bijective pebble game [26] and ideas from the theory of coherent configurations, we provide structural insights for the class of k-ary coherent configurations with abelian fibers. This allows us to finally prove that in the abelian case, it does suffice to consider other relational structures yielding the same k-ary coherent configuration.

NP-hardness in Theorem 3 is proved by combining the known relationship between the Weisfeiler-Leman dimension of CFI-graphs [11] and the tree-width of the underlying base graphs with the recent result that computing the tree-width of cubic graphs is NP-hard [9]. With the same techniques, we can also prove that deciding k-WL-equivalence of graphs is co-NP-hard when the dimension k is considered part of the input.

For the P-hardness result of the k-WL-identification problem in Theorem 4, we adapt a construction by Grohe [19] that he used to prove P-hardness of the k-WL-equivalence problem. The construction encodes monotone boolean circuits into graphs using different types of gadgets. This simultaneously reduces the monotone circuit value problem, which is known to be hard for polynomial time, to the k-WL-equivalence and the k-WL-identification problem. The main difficulty was to show identification of Grohe's gadgets, specifically his so-called *one-way switches*. We give an alternative construction of these one-way switches based on the CFI-construction. This construction simplifies proofs and more importantly yields graphs with 4-bounded color classes for every k. This shows hardness for the k-WL-equivalence and k-WL-identification problems even for graphs with 4-bounded abelian color classes.

Full proofs of all statements can be found in the full version of this paper [35].

2 The Weifeiler-Leman Algorithm and Coherent Configurations

Preliminaries. For $n \in \mathbb{N}$, we set $[n] \coloneqq \{1, \dots, n\}$. For a set A, the set of all k-element subsets of A is denoted by $\binom{A}{k}$. For two runtime-bounding functions f and g with parameters including κ, we write $f \in O_\kappa(g)$ if f/g is bounded by a function of κ. A simple graph is a pair $G = (V(G), E(G))$ of a set $V(G)$ of *vertices* and a set $E(G) \subseteq \binom{V(G)}{2}$ of undirected edges. For a directed graph, we allow $E(G) \subseteq V(G)^2 \setminus \{(v,v) \colon v \in V(G)\}$. For either graph type, we write uv for the edge $\{u,v\}$ or (u,v) respectively. For a simple or directed graph G, a *vertex-coloring* of G is a map $\chi \colon V(G) \to C$ for some finite, ordered set C of colors. Similarly, an *edge-coloring* is a map $\eta \colon E(G) \to C$. A (vertex-)color class is a set $\chi^{-1}(c)$ for some vertex color $c \in C$. If all color classes have order at most q, we say that the colored graph (G, χ) has *q-bounded color classes*.

Relational structures are a higher-arity analogue of graphs. Formally, a *k-ary relational structure* $\mathfrak{A}$ is a tuple $(V(\mathfrak{A}), R_1, \dots, R_\ell)$ of vertices $V(\mathfrak{A})$ and relations $R_i \subseteq V(\mathfrak{A})^{r_i}$ with $r_i \leq k$. The number r_i is the *arity* of the relation R_i. We again allow relational structures to come with a vertex-coloring and define q-bounded color classes as before.

An *isomorphism* between graphs G and H is a bijection $\varphi \colon V(G) \to V(H)$ such that $uv \in E(G)$ if and only if $\varphi(u)\varphi(v) \in E(H)$. In this case G and H are *isomorphic* and we write $G \cong H$. An isomorphism between edge- or vertex-colored graphs must also preserve the vertex- and edge-colors. Similarly, an isomorphism between (vertex-colored) relational structures is a (color-preserving) bijection between the vertex sets that preserves all relations and their complements. An automorphism is an isomorphism from a structure to itself. We say that a graph or relational structure $\mathfrak{A}$ has *abelian color classes* if for every color class C, the induced substructure $\mathfrak{A}[C]$ has an abelian automorphism group.

Bounded Variable Counting Logics. First-order counting logic C is the extension of first-order logic by the *counting quantifiers* $\exists^{\geq k}$ for all natural numbers k, which state that there exist at least k distinct elements satisfying the formula that follows. But because first-order logic has the ability to simulate the counting quantifier $\exists^{\geq k}$ by a sequence of k usual existential quantifiers, adding counting quantifiers does not actually increase the expressive power of first-order logic. This situation changes when we restrict the number of variables. For a natural number $k \geq 2$, we define k-variable counting logic C^k to be the fragment of C which only uses the variables $x_1, \dots, x_k$. In order to not restrict the expressive power of these logics too much, we do, however, allow *requantifications*, that is, quantifications over a variable within the scope of another quantification over the same variable. As an example, the following is a C^2-formula

$$\forall x_1 \exists x_2 \left(E x_1 x_2 \wedge \left(\exists^{\geq 5} x_1 E x_2 x_1 \right) \wedge \neg \exists^{\geq 6} x_1 E x_2 x_1 \right),$$

which states that every vertex is adjacent to a vertex of degree 5.

A relational structure $\mathfrak{A}$ is *definable* in C^k if there exists some formula $\varphi \in \mathrm{C}^k$ which is satisfied by a structure if and only if it is isomorphic to $\mathfrak{A}$.

The Weisfeiler-Leman Algorithm. The distinguishing power of bounded variable counting logics has another characterization in terms of the Weisfeiler-Leman algorithm. For every $k \geq 2$, the k-dimensional Weisfeiler-Leman algorithm (k-WL) computes an isomorphism-invariant coloring of k-tuples of vertices of a given graph G via an iterative refinement process. Initially, the algorithm colors each k-tuple according to its *isomorphism type*, i.e., $\mathbf{x} = (x_1, \dots, x_k), \mathbf{y} = (y_1, \dots, y_k) \in V(G)^k$ get the same color if and only if mapping

$x_i \mapsto y_i$ for every $i \in [k]$ is an isomorphism of the induced subgraphs $G[\{x_1, \dots, x_k\}]$ and $G[\{y_1, \dots, y_k\}]$. In each iteration, this coloring is refined as follows: if $\chi_r^G \colon V(G)^k \to C_r$ is the coloring obtained after r refinement rounds, the coloring $\chi_{r+1}^G \colon V(G)^k \to C_{r+1}$ is defined as $\chi_{r+1}^G(\mathbf{x}) \coloneqq (\chi_r^G(\mathbf{x}), M_{\mathbf{x}}^r)$, where

$$M_{\mathbf{x}}^r = \left\{\!\!\left\{ \left(\chi_r^G\big(\mathbf{x}\tfrac{y}{1}\big), \dots, \chi_r^G\big(\mathbf{x}\tfrac{y}{k}\big) \right) \colon y \in V(G) \right\}\!\!\right\}$$

and $\mathbf{x}\frac{y}{i}$ denotes the tuple obtained from $\mathbf{x}$ by replacing the i-th entry by y. If χ_{r+1}^G does not induce a finer color partition on $V(G)^k$ than χ_r^G, then the algorithm terminates and returns the *stable coloring* $\chi_\infty^G \coloneqq \chi_{r+1}^G$. This must happen before the n^k-th refinement round.

We say that k-WL *distinguishes two k-tuples* $\mathbf{x}, \mathbf{y} \in V(G)^k$ if $\chi_\infty^G(\mathbf{x}) \neq \chi_\infty^G(\mathbf{y})$ and that k-WL *distinguishes two ℓ-tuples* $\mathbf{x}, \mathbf{y} \in V(G)^\ell$ for $\ell < k$ if k-WL distinguishes the two k-tuples we get by repeating the last entries of $\mathbf{x}$ respectively $\mathbf{y}$. In either case, we write $(G, \mathbf{x}) \not\equiv_{k\text{-WL}} (G, \mathbf{y})$. Finally, k-WL *distinguishes two graphs* G and H if the multisets of stable colors computed for the k-tuples of vertices over the two graphs disagree. Otherwise, G and H are k-WL-*equivalent* and we write $G \equiv_{k\text{-WL}} H$. A graph G is *identified* by k-WL if k-WL distinguishes G from every other non-isomorphic graph. Every n-vertex graph is identified by n-WL, and the least number k such that k-WL identifies G is called the *Weisfeiler-Leman dimension* of G, denoted by $\text{WL-dim}(G)$.

k-WL is at least as powerful in distinguishing graphs as $(k-1)$-WL and this hierarchy does not collapse [11]. Completely analogously, k-WL can be applied to relational structures.

▶ **Lemma 5** ([11, 26]). *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two relational structures of arity at most k, and $\mathbf{a} \in V(\mathfrak{A})^k$ and $\mathbf{b} \in V(\mathfrak{B})^k$ two tuples of vertices. Then the following are equivalent:*

- **(i)** *For every C^{k+1}-formula $\varphi(x_1, \dots, x_k)$, we have $(\mathfrak{A}, \mathbf{a}) \models \varphi$ if and only if $(\mathfrak{B}, \mathbf{b}) \models \varphi$, and*
- **(ii)** *the stable colors computed by k-WL for the tuples $\mathbf{a}$ and $\mathbf{b}$ agree.*

Further, every stable color class is definable by a single C^{k+1}-formula.

In particular, the Weisfeiler-Leman dimension of a structure is precisely one less than the number of variables needed to define the structure in first-order counting logic.

Coherent Configurations. For an introduction to (2-ary) coherent configurations and their connection to the Weisfeiler-Leman algorithm we refer to [15]. For $k \geq 2$, a *k-ary rainbow* is a pair $(V, \mathcal{R})$ of a finite set of vertices V and a partition $\mathcal{R}$ of V^k, whose elements are called *basis relations*, that satisfies the following two conditions:

- **(R1)** For every basis relations $R \in \mathcal{R}$, all tuples $\mathbf{x}, \mathbf{y} \in R$ have the same *equality type*, i.e., $x_i = x_j$ if and only if $y_i = y_j$. We also call this the equality type of the relation R.
- **(R2)** $\mathcal{R}$ is closed under permuting indices: For all basis relations $R \in \mathcal{R}$ and permutations σ of $[k]$, the set $R^\sigma \coloneqq \{(x_{\sigma(1)}, \dots, x_{\sigma(k)}) \colon (x_1, \dots, x_k) \in R\}$ is a basis relation.

Because the vertex set V is determined by the partition $\mathcal{R}$, we write $\mathcal{R}$ to denote the rainbow $(V, \mathcal{R})$ and in this case write $V(\mathcal{R})$ for its vertex set V. A *k-ary coherent configuration* is a k-ary rainbow $\mathcal{C}$ that is stable under k-WL-refinement. Formally, this means that

- **(C)** for all basis relations $R, R_1, \dots, R_k \in \mathcal{C}$, the *intersection number*

$$p\,(R; R_1, \dots, R_k) \coloneqq \left| \left\{ y \in V(\mathcal{C}) \colon \mathbf{x}\tfrac{y}{i} \in R_i \text{ for all } i \in [k] \right\} \right|$$

is the same for all choices of $\mathbf{x} \in R$ and is thus well-defined.

For $\ell \leq k$, the partition of k-vertex tuples of an ℓ-ary relational structure according to their isomorphism type always yields a k-ary rainbow. The connection of k-WL and k-ary coherent configurations is that the partition of k-vertex tuples of a graph according to their k-WL-colors always forms a k-ary coherent configuration.

Induced Configurations. If $\mathcal{R}$ is an ℓ-ary rainbow for $\ell \leq k$, we can interpret $\mathcal{R}$ as the k-ary rainbow $\mathcal{R}|^k$ by partitioning k-tuples according to the basis relations of the ℓ-subtuples they contain. Formally, let $\sim_{\mathcal{R}}$ be the equivalence relation on $V(\mathcal{R})^\ell$ whose equivalence classes are the basis relations of $\mathcal{R}$. We define the equivalence relation $\sim_{\mathcal{R}}^k$ on $V(\mathcal{R})^k$ by writing $\mathbf{x} \sim_k \mathbf{y}$ if and only if for all $I \in \binom{[k]}{\ell}$ we have $\mathbf{x}|_I \sim_{\mathcal{R}} \mathbf{y}|_I$, where $\mathbf{x}|_I$ is the subtuple of $\mathbf{x}$ for which all indices not in I are deleted. The basis relations of $\mathcal{R}|^k$ are the equivalence classes of $\sim_{\mathcal{R}}^k$.

For every k-ary rainbow $\mathcal{R}$, there is a unique coarsest k-ary coherent configuration $\mathrm{WL}_k(\mathcal{R})$ that is at least as fine as $\mathcal{R}$ and is called the *k-ary coherent closure* of $\mathcal{R}$. For an $\ell \leq k$ and an ℓ-ary rainbow $\mathcal{R}$, we also write $\mathrm{WL}_k(\mathcal{R})$ for $\mathrm{WL}_k(\mathcal{R}|^k)$. Similarly, for an ℓ-ary relational structure $\mathfrak{A}$, we write $\mathrm{WL}_k(\mathfrak{A})$ for the partition of $V(\mathfrak{A})^k$ into k-WL-color classes.

Every k-ary coherent configuration $\mathcal{C}$ induces the ℓ-ary coherent configuration $\mathcal{C}|_\ell$ for every $\ell \leq k$ by considering the partition of tuples of the form $(x_1, \dots, x_\ell, \dots, x_\ell) \in V(\mathcal{C})^k$. This ℓ-ary coherent configuration is called the *ℓ-skeleton of $\mathcal{C}$*. For every basis relation $R \in \mathcal{C}$ and every subset $I \in \binom{[k]}{\ell}$ of the indices, the set $R_I \coloneqq \{\mathbf{x}|_I : \mathbf{x} \in R\}$ is a basis relation of $\mathcal{C}|_\ell$ and called the *I-face* of R. The 1-skeleton yields a partition of $V(\mathcal{C})$, whose partition classes are called *fibers*. We denote the set of fibers by $F(\mathcal{C})$. $\mathcal{C}$ has *c-bounded fibers* if all fibers of $\mathcal{C}$ have order at most c. If $W \subseteq V(\mathcal{C})$ is a union of fibers, the induced structure $\mathcal{C}[W]$ is again a k-ary coherent configuration. Between two fibers X and Y, the induced configuration $\mathcal{C}|_2$ further induces a partition $\mathcal{C}|_2[X, Y]$ of $X \times Y$, called an *interspace*.

A k-ary coherent configuration $\mathcal{C}$ is *ℓ-induced* if it is the coherent closure of its ℓ-skeleton, i.e., if $\mathcal{C} = \mathrm{WL}_k(\mathcal{C}|_\ell)$. This is equivalent to $\mathcal{C}$ being the coherent closure of some ℓ-ary rainbow. In particular, the k-ary coherent closure of a (directed, colored) graph is 2-induced and the k-ary coherent closure of an ℓ-ary relational structure is ℓ-induced for every $k \geq \ell$.

For a k-ary rainbow $\mathcal{R} = (V, \{R_1, \dots, R_\ell\})$, the vertex-colored k-ary relational structure $(V, R_1, \dots, R_\ell, \chi)$ where χ maps every vertex to its fiber is a *colored variant* of $\mathcal{R}$. Note that this requires choosing an ordering of the basis relations; colored variants are thus not unique.

Algebraic and Combinatorial Isomorphisms. There are two notions of isomorphism for two k-ary coherent configurations $\mathcal{C}$ and $\mathcal{D}$. First, a *combinatorial isomorphism* is a bijection $\varphi\colon V(\mathcal{C}) \to V(\mathcal{D})$ that preserves the partition into basis relations, i.e., for every basis relation $R \in \mathcal{C}$, the mapped set $R^\varphi \coloneqq \{(\varphi(x_1), \dots, \varphi(x_k)) : (x_1, \dots, x_k) \in R\}$ is a basis relation of $\mathcal{D}$. Combinatorial isomorphisms are thus isomorphisms between certain colored variants of $\mathcal{C}$ and $\mathcal{D}$ and the notion also applies to rainbows.

Second, an *algebraic isomorphism* is a map $f\colon \mathcal{C} \to \mathcal{D}$ between the two partitions that preserves the intersection numbers. More formally, we require that

(A1) for all $R \in \mathcal{C}$, the relations R and $f(R)$ have the same equality type,

(A2) for all $R \in \mathcal{C}$ and permutations σ of $[k]$, we have $f(R^\sigma) = f(R)^\sigma$, and

(A3) for all $R, T_1, \dots, T_k \in \mathcal{C}$, we have $p\,(R; T_1, \dots, T_k) = p\,(f(R); f(T_1), \dots, f(T_k))$,

but Property (A3) already implies the former two. Algebraic isomorphisms can be thought of as maps preserving the Weisfeiler-Leman colors and thus as a functional perspective on Weisfeiler-Leman equivalence. More formally, if for k-ary relational structures $\mathfrak{A}$ and $\mathfrak{B}$, $f\colon \mathrm{WL}_k(\mathfrak{A}) \to \mathrm{WL}_k(\mathfrak{B})$ is an algebraic isomorphism that preserves the relations of $\mathfrak{A}$ and

$\mathfrak{B}$, then f is the unique map that maps every color class of the stable coloring computed by k-WL on $\mathfrak{A}$ to the corresponding color class of the stable coloring computed by k-WL on $\mathfrak{B}$. In particular, we get $\mathfrak{A} \equiv_{k\text{-WL}} \mathfrak{B}$ in this case.

If $f\colon \mathcal{C} \to \mathcal{D}$ is an algebraic isomorphism, then f induces an algebraic isomorphism $f|_\ell\colon \mathcal{C}|_\ell \to \mathcal{D}|_\ell$ for every $\ell \leq k$. If $\mathcal{C} = \mathrm{WL}_k(\mathcal{R})$ for some rainbow $\mathcal{R}$, f induces a map $f|_\mathcal{R}\colon \mathcal{R} \to \mathcal{R}^f$ for some rainbow $\mathcal{R}^f$ by sending each basis relation of $\mathcal{R}$, which is a union of basis relations of $\mathcal{C}$, to the union of f-images of these basis relations of $\mathcal{C}$. A combinatorial (respectively algebraic) *automorphism of* $\mathcal{C}$ is a combinatorial (respectively algebraic) isomorphism from $\mathcal{C}$ to itself. Every combinatorial isomorphism induces an algebraic isomorphism, but the converse is not true. Algebraic isomorphisms behave nicely with coherent closures as seen in the next lemma (the proof is analogue to the $k = 2$ case [15, Lemma 2.4]):

▶ **Lemma 6.** *For all k-ary rainbows $\mathcal{R}$, algebraic isomorphisms $f\colon \mathcal{C} \to \mathcal{D}$, and $\mathcal{C} = \mathrm{WL}_k(\mathcal{R})$*
1. *$\mathcal{D} = \mathrm{WL}_k(\mathcal{R}^f)$, in particular, if $\mathcal{C}$ is ℓ-induced, then so is $\mathcal{D}$,*
2. *f is fully determined by its action on basis relations in $\mathcal{R}$, and*
3. *if $f|_\mathcal{R}$ is induced by a combinatorial isomorphism φ, then φ induces f.*

A k-ary coherent configuration $\mathcal{C}$ is called *separable* if every algebraic isomorphism $f\colon \mathcal{C} \to \mathcal{D}$ from $\mathcal{C}$ is induced by a combinatorial one. There is a close relation to the power of the Weisfeiler-Leman algorithm (the proof is analogue to the $k = 2$ case [15, Theorem 2.5]):

▶ **Lemma 7.** *Let $\ell \leq k$ and $\mathfrak{A}$ be an ℓ-ary relational structure. Then $\mathfrak{A}$ is identified by the k-dimensional Weisfeiler-Leman algorithm if and only if $\mathrm{WL}_k(\mathfrak{A})$ is separable.*

3 Deciding Identification for Graphs With 5-Bounded Color Classes

As recently shown [15], identification of graphs with 5-bounded color classes by 2-WL is polynomial-time decidable. We extend this result to arbitrary dimensions of the Weisfeiler-Leman algorithm. We adapt the approach of [15] and solve the separability problem for 2-induced k-ary coherent configurations with 5-bounded fibers. We generalize the elimination of interspaces containing a matching and of interspaces of type $2K_{1,2}$: we reduce to *star-free* k-ary coherent configurations. By characterizing separability using certain automorphism groups, we provide a new reduction of the separability problem for such configurations to graph isomorphism for bounded color classes, which can be solved in polynomial time.

Disjoint Unions of Stars. Let $\mathcal{C}$ be a k-ary coherent configuration and $X, Y \in F(\mathcal{C})$ two distinct fibers. A *disjoint union of stars* between X and Y is a basis relation $S \in \mathcal{C}|_2[X, Y]$ such that every vertex in Y is incident to exactly one edge in S. If no interspace of $\mathcal{C}$ contains a disjoint union of stars, then $\mathcal{C}$ is called *star-free*. We show that the separability problem of (2-induced) k-ary coherent configurations reduces to that of star-free ones.

▶ **Lemma 8.** *Let $\mathcal{C}$ be a k-ary coherent configuration, $X, Y \in F(\mathcal{C})$ two distinct fibers, and $S \in \mathcal{C}|_2[X, Y]$ a disjoint union of stars between X and Y. Then $\mathcal{C}$ is separable if and only if $\mathcal{C} \setminus X := \mathcal{C}[V(\mathcal{C}) \setminus X]$ is separable. Furthermore, if $\mathcal{C}$ is 2-induced, then so is $\mathcal{C} \setminus X$.*

Proof sketch. Let Eq_S be the set of pairs of vertices in Y that have a common S-neighbor in X. We show that the k-ary coherent configuration $\mathcal{C}$ is uniquely determined by the configuration $\mathcal{C} \setminus X$ and the relation Eq_S. For this, consider the function $\nu_S\colon Y \to X$ that maps each vertex in Y to its unique neighbor in X. When we apply this map to some of the Y-components of a basis relation $R \in \mathcal{C}$, the resulting set is again a basis relation, and we can obtain every basis relation of $R \in \mathcal{C}$ from basis relations in $\mathcal{C} \setminus X$ in this way.

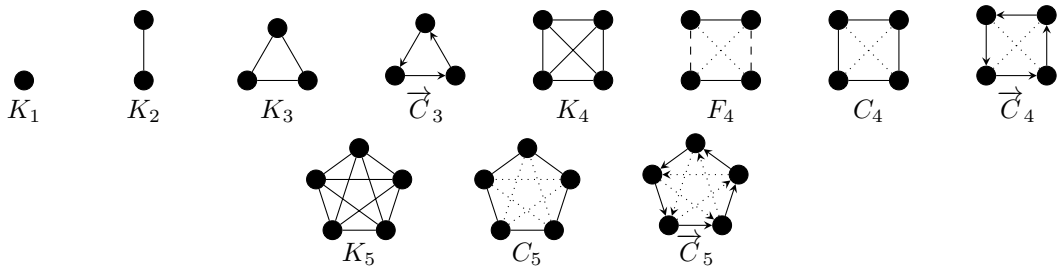


Figure 1 The complete list of 2-ary coherent configurations on a single fiber of order up to 4 from [15], and the three 2-ary coherent configurations on a single fiber of order 5 from [41].

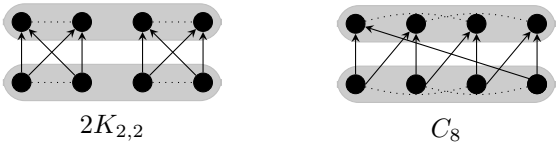


Figure 2 All non-uniform and star-free interspace types between two fibers of order up to 5. In each case, there are at least two basis relations in each fiber, including the drawn matchings, and two basis relations between the fibers: the drawn one and its complement.

We show that both algebraic and combinatorial isomorphisms can detect the uniqueness of this extension in the following sense: every algebraic or combinatorial isomorphism $\mathcal{C} \setminus X \to \mathcal{D}$ uniquely extends to an algebraic or combinatorial isomorphism $\mathcal{C} \to \mathcal{D}^*$, for some uniquely determined extension $\mathcal{D}^* \supseteq \mathcal{D}$ by a single fiber. Hence, $\mathcal{C}$ is separable if and only if $\mathcal{C} \setminus X$ is. By analyzing this unique extension, it is moreover clear that it does not affect 2-inducedness. ◀

Lemma 8 allows us to remove fibers that are incident to a disjoint unions of stars without affecting the separability. This simultaneously generalizes the elimination of interspaces containing a matching and the elimination of fibers of size 2 from [15].

5-Bounded Fibers. In order to structurally understand 2-induced k-ary coherent configurations, it mostly suffices to understand their 2-skeletons. The possible isomorphism types of 2-ary coherent configurations on a single fiber of order at most 5 are known [15, 41], see Figure 1. Further, the possible interspaces between fibers of order up to 4 are also known [15]. For fibers $X, Y \in F(\mathcal{C})$, it is always possible that the interspace $\mathcal{C}[X, Y]$ is *uniform*, meaning that it consists of only a single basis relation $X \times Y$. Every interspace between a fiber of size 5 and a fiber of size at most 4 is uniform [15, Lemma 3.1], and every interspace between two fibers of size 5 is either uniform or contains a matching [15, Section 13]. Now, only two possible non-uniform, star-free interspaces remain, which are depicted in Figure 2.

We call an algebraic automorphism f of a k-ary coherent configuration $\mathcal{C}$ *strict* if it fixes every fiber, i.e., it satisfies $f(X) = X$ for every $X \in F(\mathcal{C})$. The strict algebraic automorphisms of $\mathcal{C}$ form a group, which we denote by $\mathbb{A}(\mathcal{C})$. Using the enumeration of fiber and interspace types, we obtain the following reformulation of separability as in [15, Lemma 7.2].

▶ **Lemma 9.** *A star-free 2-induced k-ary coherent configuration with 5-bounded fibers is separable if and only if every strict algebraic automorphism is induced by a combinatorial one.*

Proof sketch. The forward implication is immediate. For the backward implication, consider an algebraic isomorphism $f\colon \mathcal{C} \to \mathcal{D}$. Because every coherent configuration of order at most 8 is separable [15], f is induced by a combinatorial isomorphism on every union of two fibers. We pick such a combinatorial isomorphism inducing $f|_2$ for every interspace of type C_8 and everywhere else, we pick combinatorial isomorphisms inducing $f|_2$ just on each fiber. Using the structure of fibers of order at most 5 and their interspaces, we can show that these isomorphisms combine to a combinatorial isomorphism φ inducing $f|_2$ on every fiber. But then, $\varphi^{-1} \circ f|_2$ is a strict algebraic automorphism and is thus induced by a combinatorial automorphism θ. But then, $\varphi \circ \theta$ induces $f|_2$ and thus also f by Lemma 6. ◀

Strict Algebraic Automorphisms. We now sketch a polynomial-time algorithm that decides whether every strict algebraic automorphism of a k-ary coherent configuration is induced by a combinatorial automorphism. We will heavily use that the graph isomorphism problem is polynomial-time solvable for graphs with bounded color classes.

▶ **Lemma 10** ([6, 16]). *Isomorphism of k-ary relational structures of order n and c-bounded color classes is decidable in time $O_{k,c}(n^{O(k)})$. A generating set of the automorphism group of these structures is computable in time $O_{k,c}(n^{O(k)})$.*

By encoding the algebraic structure of a coherent configuration $\mathcal{C}$ into a relational structure, we obtain the following:

▶ **Lemma 11.** *There is an algorithm running in time $O_{k,c}(n^{O(k)})$ that, given a k-ary coherent configuration $\mathcal{C}$ of order n with c-bounded fibers, computes a generating set of $\mathbb{A}(\mathcal{C})$.*

Proof. We construct a $(k+1)$-ary relational structure $\mathfrak{A}_{\mathcal{C}}$ with c^k-bounded color classes such that $\mathrm{Aut}(\mathfrak{A}_{\mathcal{C}}) \cong \mathbb{A}(\mathcal{C})$. The vertices of our constructed structure are the basis relations of $\mathcal{C}$, and we color each $(k+1)$-tuple $(R, R_1, \dots, R_k)$ of basis relations using the color $p(R; R_1, \dots, R_k)$. Further, we color each basis relation by the tuple of fibers of its components. Because every fiber is c-bounded, at most c^k basis relations can share a color. Furthermore, it is immediate that the automorphisms of the structure constructed so far naturally correspond to strict algebraic automorphisms of $\mathcal{C}$. Thus, we can compute a generating set for the group of strict algebraic automorphisms in the required time using Lemma 10. ◀

Similarly, we can reduce the question whether a strict algebraic automorphism is induced by a combinatorial one to the graph isomorphism problem.

▶ **Lemma 12.** *There is an algorithm running in time $O_{k,c}(n^{O(k)})$ that, given a k-ary coherent configuration $\mathcal{C}$ with c-bounded fibers and a strict algebraic automorphism $f \in \mathbb{A}(\mathcal{C})$, decides whether f is induced by a combinatorial automorphism.*

Proof sketch. Let $\mathfrak{C}$ be an arbitrary colored variant of $\mathcal{C}$ and $\mathfrak{C}^f$ another colored variant such that f is a color-preserving map between the color classes of $\mathfrak{C}$ and $\mathfrak{C}^f$. Combinatorial automorphisms inducing f correspond to isomorphisms between $\mathfrak{C}$ and $\mathfrak{C}^f$, meaning that f is induced by a combinatorial automorphism if and only if $\mathfrak{C} \cong \mathfrak{C}^f$. As $\mathcal{C}$ has bounded fibers, so does $\mathfrak{C}$. Thus, we can decide the latter in the required time by Lemma 10. ◀

▶ **Corollary 13.** *There is an algorithm running in time $O_{k,c}(n^{O(k)})$ that, given a k-ary coherent configuration $\mathcal{C}$ with c-bounded fibers, decides whether every strict algebraic automorphism of $\mathcal{C}$ is induced by a combinatorial automorphism.*

Proof. Those strict algebraic automorphisms that are induced by combinatorial ones form a subgroup of the group of all strict algebraic automorphisms. Hence, it suffices to compute a generating set of the whole group via Lemma 11 and to decide whether all elements of it are induced by combinatorial automorphisms using Lemma 12. ◀

Finally, we are ready to prove our first theorem.

▶ **Theorem 1.** *For every k, there is an algorithm that decides the k-WL-identification problem for vertex- and edge-colored, directed graphs with 5-bounded color classes in time $O_k(n^{O(k)})$. If such a graph G is not identified by k-WL, the algorithm provides a witness for this, i.e., a graph H that is not isomorphic to G and not distinguished from G by k-WL.*

Proof. In a first step, we run k-WL on G to get the 2-induced configuration $\mathcal{C} \coloneqq \mathrm{WL}_k(G)$. By Lemma 7, it remains to decide whether $\mathcal{C}$ is separable. Now, we eliminate disjoint unions of stars using Lemma 8, while maintaining 2-inducedness of $\mathcal{C}$. By Lemma 9, it remains to decide whether every strict algebraic automorphism is induced by a combinatorial one. This can be achieved using Corollary 13.

If this is the case, the input structure is identified by k-WL. Otherwise, the algorithm actually finds a strict algebraic automorphism f which is not induced by a combinatorial automorphism. By adding back all interspaces containing a disjoint union of stars, we can extend f to an algebraic isomorphism $\widehat{f}\colon \mathrm{WL}_k(G) \to \mathcal{D}$ which is not induced by a combinatorial isomorphism. But then, we can obtain a witnessing graph H from G by replacing its edge set by its $\widehat{f}|_2$-image and similarly translating vertex- and edge-colors along $\widehat{f}$. ◀

4 Identification for Structures With Bounded Abelian Color Classes

The approach we used in Section 3 to decide the k-WL-identification problem for graphs with 5-bounded color classes does not easily generalize to larger bounds on the color classes or to relational structures of higher arity. In particular, Lemma 9 was crucial in the reduction of k-WL-identification to a statement on certain automorphisms which could be handled using group-theoretic techniques. The proof of the lemma was based on an explicit case distinction on the possible isomorphism types of interspaces, and fails for graphs with larger color classes. In this section, we show that Lemma 9 remains true in the special case of relational structures with bounded abelian color classes, i.e., structures for which the automorphism group of the structure induced on each color class is abelian. Such structures were already considered in the context of descriptive complexity theory [44] and include both CFI-graphs [11] and multipedes [39, 38] over ordered base graphs.

Coherent Configurations With Abelian Fibers. To start, we translate the concept of abelian color classes to the corresponding concept of abelian fibers for k-ary coherent configurations. A combinatorial automorphism φ of a k-ary coherent configuration $\mathcal{D}$ is *color-preserving* if φ fixes every basis relation of $\mathcal{D}$. This is equivalent to φ being an automorphism of every colored variant of $\mathcal{D}$ or to the algebraic automorphism induced by φ being the identity (recall that combinatorial automorphisms are not required to fix every basis relation, but only the partition of $V(\mathcal{D})^k$ into basis relations). We say that a coherent configuration $\mathcal{C}$ has *abelian fibers* if, for each fiber $X \in F(\mathcal{C})$, the group of color-preserving combinatorial automorphisms of $\mathcal{C}[X]$ is abelian.

▶ **Lemma 14.** *Let $\mathfrak{A}$ be a relational structure of arity at most k. If $\mathfrak{A}$ has abelian color classes, then $\mathrm{WL}_k(\mathfrak{A})$ has abelian fibers.*

We start with a structural lemma, which states that small abelian fibers are always *thin*. For a fiber $X \in F(\mathcal{C})$, a binary basis relation $S \in \mathcal{C}|_2[X]$ is called *thin* if every vertex in X is incident to exactly one ingoing and exactly one outgoing S-edge, that is, if S is either a matching or a union of directed cycles. The fiber X is called *thin* if all basis relations $R \in \mathcal{C}|_2[X]$ are thin and if this is true for all fibers of $\mathcal{C}$, we say that $\mathcal{C}$ has *thin fibers*.

▶ **Lemma 15.** *Let $\mathcal{C}$ be a k-ary coherent configuration. Then every abelian fiber of order at most k is thin.*

Proof. Let $X \in F(\mathcal{C})$ be an abelian fiber of order at most k. Then $\mathcal{C}|_2[X]$ is the partition of X^2 into orbits under the natural action of the group of color-preserving automorphisms.

Now, assume that some binary basis relation $S \in \mathcal{C}|_2[X]$ contains two pairs xy and xy' for $x, y, y' \in X$. This implies that there is a color-preserving automorphism φ of $\mathcal{C}[X]$ that maps xy to xy'. But as the group of color-preserving automorphism of $\mathcal{C}[X]$ is abelian and acts transitvely on the vertices of X, its point-stabilizers are trivial. Because $\varphi(x) = x$, this implies $\varphi = \mathrm{id}_X$ and thus $y' = \varphi(y) = y$. Thus, the basis relation S is thin. ◀

Next, we need one well-known lemma on the structure of thin fibers, which essentially states that thin fibers correspond to Cayley graphs of their automorphism groups.

▶ **Lemma 16** ([12, Section 2.1.4])**.** *Let $\mathcal{C}$ be a 2-ary coherent configuration on a single thin fiber. Then the basis relations of $\mathcal{C}$ are precisely those of the form $S_\varphi := \{x\varphi(x) \colon x \in V(\mathcal{C})\}$ for color-preserving combinatorial automorphisms φ of $\mathcal{C}$.*

Separability of Configurations With Bounded Thin Fibers. Next, we show that k-ary coherent configurations with few, thin fibers are separable:

▶ **Lemma 17.** *Let $\mathcal{C}$ be a k-ary coherent configuration with at most k fibers. If $\mathcal{C}$ has thin fibers, then $\mathcal{C}$ is separable.*

Proof sketch. Let $\mathbf{x} \in V(\mathcal{C})^k$ be a k-tuple of vertices which contains a vertex from every fiber of $\mathcal{C}$. Then every vertex of $\mathcal{C}$ is the unique outgoing neighbor of some vertex in $\mathbf{x}$ with respect to some thin basis relation. Thus, all vertices of $\mathcal{C}$ are fixed relative to $\mathbf{x}$.

Now, let $\mathfrak{C}$ be a colored variant of $\mathcal{C}$. By Lemma 7, $\mathcal{C}$ is separable if and only if $\mathfrak{C}$ is identified by k-WL. We show the latter using the bijective $(k+1)$-pebble game, which is an Ehrenfeucht-Fraïssé-type game capturing k-WL-equivalence [26]. Indeed, if $\mathfrak{C} \equiv_{k\text{-WL}} \mathfrak{D}$, this corresponds to Duplicator having a winning strategy in this game. But because we can fix every vertex of $\mathfrak{C}$ by only fixing one vertex per color class, we can easily extract an isomorphism $\mathfrak{C} \to \mathfrak{D}$ from such a winning strategy. ◀

Finally, we are ready to once again reduce the question of separability to only strict algebraic automorphisms, which we can again deal with using Corollary 13.

▶ **Lemma 18.** *Let $\mathcal{C}$ be a k-ary coherent configurations with thin fibers. Then $\mathcal{C}$ is separable if and only if every strict algebraic automorphism of $\mathcal{C}$ is induced by a combinatorial automorphism.*

Proof sketch. The proof is similar to that of Lemma 9, where instead of using that every 2-ary coherent configuration of order at most 8 is separable, we apply Lemma 17 to get separability of sufficiently small configurations. Afterwards, we use the structure of configurations with thin fibers to show that the local bijections we picked are compatible with the partition in the k-ary interspaces. ◀

▶ **Theorem 2.** *For every $k \in \mathbb{N}$ and $c, r \leq k$, there is an algorithm that decides the k-WL-identification problem for r-ary relational structures with c-bounded abelian color classes in time $O_k(n^{O(k)})$. If such a structure $\mathfrak{A}$ is not identified by k-WL, the algorithm provides a witness for this, i.e., a second structure $\mathfrak{B}$ that is not isomorphic to $\mathfrak{A}$ and not distinguished from $\mathfrak{A}$ by k-WL.*

Proof. Let $\mathfrak{A}$ be a relational structure of arity r. Then $\mathfrak{A}$ is identified by k-WL if and only if $\mathrm{WL}_k(\mathfrak{A})$ is separable. Because the k-ary coherent configuration $\mathrm{WL}_k(\mathfrak{A})$ has c-bounded thin fibers by Lemmas 14 and 15, Lemma 18 implies that separability of $\mathrm{WL}_k(\mathfrak{A})$ is equivalent to every strict algebraic automorphism of $\mathrm{WL}_k(\mathfrak{A})$ being induced by a combinatorial automorphism. This can be checked in the given time using Corollary 13, and in case of a negative answer, we can construct a non-isomorphic but non-distinguished structure from the strict algebraic automorphism not induced by a combinatorial one as in Theorem 1. ◀

Note that the restriction to relational structures of arity at most k is insubstantial, because the standard variant of the Weisfeiler-Leman algorithm given in Section 2 does not identify any relational structure of arity larger than k, simply because it does not consider tuples of length larger than k and thus cannot even detect whether a relation of arity larger than k is empty. While there are variants of k-WL which identify some $(k+1)$-ary relational structures, these variants can be treated similarly to decide identification by those algorithms.

5 Hardness

We now prove hardness results that complement the positive results in the previous two sections. In the case that the dimension k is part of the input, the k-WL-equivalence problem and the k-WL-identification problem are co-NP-hard and NP-hard, respectively. We use that deciding whether a cubic graph has tree-width k is NP-hard [9]. The CFI-construction [11] assigns to a graph G two CFI-graphs, which are distinguished by k-WL if and only if G has tree-width at most k [8, 22]. If G is cubic, then the CFI-graphs are polynomial-time computable and thus we reduced to k-WL-equivalence. To show hardness of k-WL-identification, we show that the CFI-graphs are actually identified by k-WL if the tree-width of G is at most k. Hardness of the k-WL-equivalence problem was independently observed by Seppelt [42].

▶ **Theorem 3.** *The problem of deciding, given a graph G and a natural number k, whether the Weisfeiler-Leman dimension of G is at most k is NP-hard, both over uncolored simple graphs, and over simple graphs with 4-bounded color classes.*

▶ **Theorem 19.** *The problem of deciding, for a given pair of graphs G and H and a natural number $k \geq 1$, whether $G \equiv_{k\text{-WL}} H$ is co-NP-hard, both over uncolored simple graphs, and over simple graphs with 4-bounded abelian color classes.*

P-Hardness for Fixed Dimension. We again turn to the k-WL-identification problem for a fixed dimension $k \geq 2$, and show that both over uncolored simple graphs and over simple graphs with 4-bounded abelian color classes, the problem is P-hard under logspace-uniform AC_0-reductions. We reduce from the P-hard monotone circuit value problem MCVP [17]. Our construction of a graph from a monotone circuit closely resembles the reductions of Grohe [19] to show P-hardness of the k-WL-equivalence problem. A similar reduction was also used to prove P-hardness of the identification problem for the color refinement algorithm (1-WL) [2].

The reduction is based on so-called *one-way switches*, which were introduced by Grohe [19]. These graph gadgets allow color information computed by the Weisfeiler-Leman algorithm to pass in one direction, but block it from passing in the other. And while Grohe provides one-way switches for every dimension of the Weisfeiler-Leman algorithm, his gadgets have large color classes and are difficult to analyze. Instead, we give a new construction of such gadgets with 4-bounded color classes. We then use these one-way switches to construct a graph from an instance of the monotone circuit value problem from the identification of which we can read off the answer to the initial MCVP-query.

One-Way Switches. Fix a dimension $k \geq 2$ of the Weisfeiler-Leman algorithm. A *k-one-way switch* is a graph gadget with a pair of *input vertices* $\{y_1, y_2\}$ and a pair of *output vertices* $\{x_1, x_2\}$, which each form a color class of size 2. A pair of vertices is *split* if the two vertices are colored differently and k-WL *splits* a pair if the coloring computed by k-WL splits the pair. The crucial property of a k-one-way switch is the following: if the input pair $\{y_1, y_2\}$ of the one-way switch is split, then k-WL also splits the output pair, but not the other way around. One-way switches thus only allow one-way flow of k-WL-color information. In contrast to Grohe's gadgets, our one-way switches are based on the CFI-construction [11]. We give only a brief sketch of the properties of our one-way switches here, and postpone their precise construction and properties to Appendix A.

▶ **Lemma 20** (simplified). *For every $k \geq 2$, there is a colored graph O^k with 4-bounded abelian color classes, called k-*one-way switch*, with an input pair $\{y_1, y_2\}$ and an output pair $\{x_1, x_2\}$, neither of which is split by k-WL, such that*

1. *the graph O^k_{split} obtained by splitting the input pair $\{y_1, y_2\}$ is identified by k-WL,*
2. *k-WL splits the output pair $\{x_1, x_2\}$ of O^k_{split}, and*
3. *if we split the output pair $\{x_1, x_2\}$, k-WL still does not split the input pair $\{y_1, y_2\}$.*

From Monotone Circuits to Graphs. We reduce the monotone circuit value problem to the k-WL-identification problem. A monotone circuit M is a circuit consisting of input nodes with values True or False, inner nodes, which are either AND- or OR-nodes with two inputs each, and a distinguished output node. We write $V(M)$ for the set of nodes of M. With a monotone circuit M, we associate the evaluation function $\text{val}_M \colon V(M) \to \{\text{True}, \text{False}\}$, which is defined in the expected way. The monotone circuit value problem asks whether the output node of a given monotone circuit evaluates to True and is P-hard by [17].

Let M be a monotone circuit. We construct a colored graph G_M such that for every node $a \in V(M)$, there is a vertex pair $\{a_1, a_2\}$ in G_M that will be split by k-WL if and only if $\text{val}_M(a) = \text{False}$. Up to the construction of the one-way switches, the construction of G_M is similar to constructions employed in [19] and [2]. We use two graphs G_{OR} and G_{AND} (see Figure 3) as gadgets to simulate logic gates. These gadgets both have two input pairs and one output pair such that exchanging the two output vertices by an automorphism requires the two vertices of one (for G_{OR}) or both (for G_{AND}) input pairs to also be exchanged.

The formal construction of G_M is depicted in Figure 4. For every node a of M, we add a pair of vertices $\{a_1, a_2\}$ forming a color class of G_M. To encode the input values of the circuit, we split every pair $\{a_1, a_2\}$ corresponding to an input node a of value False. For every AND-node $a \in V(M)$ with input nodes b and b', we add a freshly colored copy of the gadget G_{AND} from Figure 3 and identify its output pair with the pair $\{a_1, a_2\}$. Next, we connect its two input pairs via freshly colored one-way switches O^k_{ba} and $O^k_{b'a}$ to the pairs $\{b_1, b_2\}$ and $\{b'_1, b'_2\}$ respectively. More precisely, we identify the input pairs of these

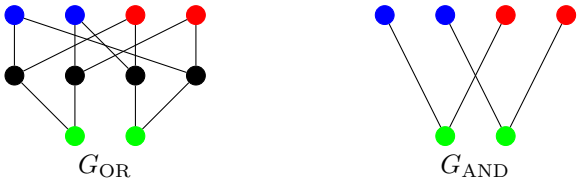


Figure 3 The gadgets G_{OR} and G_{AND} encoding OR- and AND-gates respectively. The two vertex pairs at the top are their *input pairs* and the bottom pair is their *output pair*.

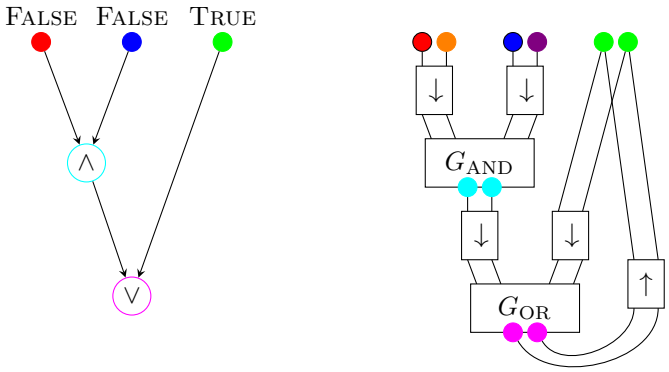


Figure 4 A simple monotone circuit M and the graph G_M obtained from it.

one-way switches with $\{b_1, b_2\}$ or $\{b'_1, b'_2\}$ respectively and identify their output pair with the respective input pair of the copy is identif of G_{AND}. Analogously, we add a copy of G_{OR} for every OR-node $a \in V(M)$ and connect it to its input via one-way switches as before.

This concludes the translation of the circuit itself, but for our reduction to the identification problem we need one more step: we connect all input pairs $\{a_1, a_2\}$ with $\text{val}_M(a) = \text{True}$ to the output pair $\{c_1, c_2\}$ via additional one-way switches O^k_{ca}, whose input pair we identify with $\{c_1, c_2\}$ and whose output pair we identify with $\{a_1, a_2\}$. Let G_M be the resulting graph. Because we color different gadgets using distinct colors, and every gadget has 4-bounded color classes, the resulting graph G_M also has 4-bounded color classes and indeed, these color classes could also be made abelian by introducing colored edges within the gadgets. The following lemma is proven similar to [19, Section 5.4] and [2, Theorem 7.11] by using the properties of the one-way switches to bound the distinguishing power of k-WL on G_M in terms of its distinguishing power on the individual gadgets.

Recall that G_M contains, for every node a of M, a vertex pair $\{a_1, a_2\}$. The essential property of the encoding of monotone circuits as graphs is the following:

▶ **Lemma 21.** *For every monotone circuit M with output node c and every node a of M, we have $(G_M, a_1) \equiv_{k\text{-WL}} (G_M, a_2)$ if and only if $\text{val}_M(a) = \text{val}_M(c) = \text{True}$.*

▶ **Corollary 22.** *The k-WL-equivalence problem for vertices is P-hard under uniform AC_0-reductions, both over simple graphs with 4-bounded abelian color classes, and over uncolored simple graphs.*

Consider now the modified graph G^*_M that we get by adding another freshly colored one-way switch O^k_* whose input pair is $\{c_1, c_2\}$, i.e., the vertex pair corresponding to the output node of the circuit M. Furthermore, we split the output pair of O^k_*. Note that when splitting the

output pair, we can choose which of the two vertices to give a fresh color to. We show that these two choices lead to non-isomorphic graphs which are distinguished by k-WL if and only if the circuit evaluates to FALSE.

▶ **Lemma 23.** *For every monotone circuit M, the graph G^*_M is identified by k-WL if and only if* $\mathrm{val}_M(c) = \textsc{False}$.

Proof sketch. If $\mathrm{val}_M(c) = \textsc{False}$, then all input and output pairs in G^*_M are split. Because all gadgets in G^*_M are identified when their input and output pairs are split, and different gadgets only interact at these split pairs, the whole graph G^*_M is identified.

Conversely, assume $\mathrm{val}_M(c) = \textsc{True}$, and let $(G^*_M)'$ be the graph constructed just like G^*_M, but with the colors of the two output vertices of the one-way switch O^k_* exchanged. Because the pair $\{c_1, c_2\}$ is not split in G_M, k-WL cannot distinguish the two output vertices of O^k_*, which means that it cannot distinguish the non-isomorphic graphs G^*_M and $(G^*_M)'$. ◀

▶ **Theorem 4.** *For every $k \geq 1$, the k-WL-identification problem is P-hard under uniform AC_0-reductions over both uncolored simple graphs, and simple graphs with 4-bounded abelian color classes.*

6 Conclusion

We have shown on the one hand that when the dimension k is part of the input, the k-WL-equivalence problem and the k-WL-identification problem are co-NP-hard and NP-hard, respectively.

On the other hand, when the dimension k is fixed, the equivalence problem is trivially solvable in polynomial time, and we have shown that the identification problem is solvable in polynomial time over graphs with 5-bounded color classes and on relational structures with k-bounded abelian color classes. Still, the identification problem is P-hard in both cases. As an immediate corollary, we obtain the same polynomial-time solvability and hardness results for definability and equivalence in the bounded-variable logic with counting C^k.

It would be interesting to know whether the k-WL-identification problem can be solved in polynomial time for larger color classes or indeed on general graphs when k is fixed. Indeed, our NP-hardness reduction was based on whether the tree-width of a given graph is at most k, which can be solved in linear time for every fixed k [7], and thus does not even yield a super-linear lower bound when k is fixed. Still, we would expect that neither the identification nor the equivalence problem can be solved in time $n^{o(k)}$. It might be fruitful to study these problems from the lens of parameterized complexity or provide lower complexity bounds based on the (strong) exponential time hypothesis.

References

1 Markus Anders and Pascal Schweitzer. Parallel computation of combinatorial symmetries. In *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pages 6:1–6:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ESA.2021.6.

2 Vikraman Arvind, Johannes Köbler, Gaurav Rattan, and Oleg Verbitsky. Graph isomorphism, color refinement, and compactness. *Comput. Complex.*, 26(3):627–685, 2017. doi:10.1007/S00037-016-0147-6.

3 Albert Atserias and Elitza N. Maneva. Sherali-Adams relaxations and indistinguishability in counting logics. *SIAM J. Comput.*, 42(1):112–137, 2013. doi:10.1137/120867834.

4 László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 684–697. ACM, 2016. doi:10.1145/2897518.2897542.

5 László Babai and Ludek Kucera. Canonical labelling of graphs in linear average time. In *20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29-31 October 1979*, pages 39–46. IEEE Computer Society, 1979. doi:10.1109/SFCS.1979.8.

6 László Babai. Monte-Carlo algorithms in graph isomorphism testing. Technical Report 79-10, Université de Montréal, 1979.

7 Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996. doi:10.1137/S0097539793251219.

8 Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

9 Hans L. Bodlaender, Édouard Bonnet, Lars Jaffke, Dušan Knop, Paloma T. Lima, Martin Milanič, Sebastian Ordyniak, Sukanya Pandey, and Ondřej Suchý. Treewidth is NP-complete on cubic graphs. In *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPIcs*, pages 7:1–7:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.IPEC.2023.7.

10 Béla Bollobás. Distinguishing vertices of random graphs. *North-holland Mathematics Studies*, 62:33–49, 1982. doi:10.1016/S0304-0208(08)73545-X.

11 Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

12 G. Chen and I. Ponomarenko. *Lectures on Coherent Configurations*. Central China Normal University Press, 2019. A draft is available at https://www.pdmi.ras.ru/~inp/.

13 Anuj Dawar and David Richerby. The power of counting logics on restricted classes of finite structures. In *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 84–98. Springer, 2007. doi:10.1007/978-3-540-74915-8_10.

14 Zdenek Dvorák. On recognizing graphs by numbers of homomorphisms. *J. Graph Theory*, 64(4):330–342, 2010. doi:10.1002/JGT.20461.

15 Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. Identifiability of graphs with small color classes by the Weisfeiler-Leman algorithm. *SIAM J. Discret. Math.*, 35(3):1792–1853, 2021. doi:10.1137/20M1327550.

16 Merrick Furst, John Hopcroft, and Eugene M. Luks. A subexponential algorithm for trivalent graph isomorphism. Technical report, Cornell University, USA, 1980.

17 Leslie M. Goldschlager. The monotone and planar circuit value problems are log space complete for P. *SIGACT News*, 9(2):25–29, 1977. doi:10.1145/1008354.1008356.

18 Erich Grädel and Wied Pakusa. Rank logic is dead, long live rank logic! *J. Symb. Log.*, 84(1):54–87, 2019. doi:10.1017/jsl.2018.33.

19 Martin Grohe. Equivalence in finite-variable logics is complete for polynomial time. *Comb.*, 19(4):507–532, 1999. doi:10.1007/S004939970004.

20 Martin Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. *J. ACM*, 59(5):27:1–27:64, 2012. doi:10.1145/2371656.2371662.

21 Martin Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory*, volume 47 of *Lecture Notes in Logic*. Cambridge University Press, 2017. doi:10.1017/9781139028868.

22 Martin Grohe, Moritz Lichter, Daniel Neuen, and Pascal Schweitzer. Compressing CFI graphs and lower bounds for the Weisfeiler-Leman refinements. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 798–809. IEEE, 2023. doi:10.1109/FOCS57990.2023.00052.

23 Martin Grohe and Julian Mariño. Definability and descriptive complexity on databases of bounded tree-width. In *Database Theory - ICDT '99, 7th International Conference, Jerusalem, Israel, January 10-12, 1999, Proceedings*, volume 1540 of *Lecture Notes in Computer Science*, pages 70–82. Springer, 1999. doi:10.1007/3-540-49257-7_6.

24 Martin Grohe and Daniel Neuen. Canonisation and definability for graphs of bounded rank width. *ACM Trans. Comput. Log.*, 24(1):6:1–6:31, 2023. doi:10.1145/3568025.

25 Martin Grohe and Martin Otto. Pebble games and linear equations. *J. Symb. Log.*, 80(3):797–844, 2015. doi:10.1017/JSL.2015.28.

26 Lauri Hella. Logical hierarchies in PTIME. *Inf. Comput.*, 129(1):1–19, 1996. doi:10.1006/INCO.1996.0070.

27 Neil Immerman and Eric S. Lander. *Describing Graphs: A First-Order Approach to Graph Canonization*, pages 59–81. Springer New York, New York, NY, 1990. doi:10.1007/978-1-4612-4478-3_5.

28 Tommi A. Junttila and Petteri Kaski. Engineering an efficient canonical labeling tool for large and sparse graphs. In *Proceedings of the Nine Workshop on Algorithm Engineering and Experiments, ALENEX 2007, New Orleans, Louisiana, USA, January 6, 2007*. SIAM, 2007. doi:10.1137/1.9781611972870.13.

29 Tommi A. Junttila and Petteri Kaski. Conflict propagation and component recursion for canonical labeling. In *Theory and Practice of Algorithms in (Computer) Systems - First International ICST Conference, TAPAS 2011, Rome, Italy, April 18-20, 2011. Proceedings*, volume 6595 of *Lecture Notes in Computer Science*, pages 151–162. Springer, 2011. doi:10.1007/978-3-642-19754-3_16.

30 Sandra Kiefer, Ilia Ponomarenko, and Pascal Schweitzer. The Weisfeiler-Leman dimension of planar graphs is at most 3. *J. ACM*, 66(6):44:1–44:31, 2019. doi:10.1145/3333003.

31 Sandra Kiefer, Pascal Schweitzer, and Erkal Selman. Graphs identified by logics with counting. *ACM Trans. Comput. Log.*, 23(1):1:1–1:31, 2022. doi:10.1145/3417515.

32 Ludek Kucera. Canonical labeling of regular graphs in linear average time. In *28th Annuqal Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 271–279. IEEE Computer Society, 1987. doi:10.1109/SFCS.1987.11.

33 Moritz Lichter. Separating rank logic from polynomial time. *J. ACM*, 70(2), March 2023. doi:10.1145/3572918.

34 Moritz Lichter. Witnessed symmetric choice and interpretations in fixed-point logic with counting. In *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *LIPIcs*, pages 133:1–133:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.133.

35 Moritz Lichter, Simon Raßmann, and Pascal Schweitzer. Computational complexity of the Weisfeiler-Leman dimension. *CoRR*, abs/2402.11531, 2024. doi:10.48550/arXiv.2402.11531.

36 Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *J. Symb. Comput.*, 60:94–112, 2014. doi:10.1016/J.JSC.2013.09.003.

37 Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4602–4609. AAAI Press, 2019. doi:10.1609/AAAI.V33I01.33014602.

38 Daniel Neuen and Pascal Schweitzer. Benchmark graphs for practical graph isomorphism. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 60:1–60:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ESA.2017.60.

39 Daniel Neuen and Pascal Schweitzer. An exponential lower bound for individualization-refinement algorithms for graph isomorphism. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 138–150. ACM, 2018. doi:10.1145/3188745.3188900.

40 Thomas Schneider and Pascal Schweitzer. An upper bound on the Weisfeiler-Leman dimension, 2024. arXiv:2403.12581, doi:10.48550/arXiv.2403.12581.

41 Kyoungah See and Sung Y. Song. Association schemes of small order. *Journal of Statistical Planning and Inference*, 73(1):225–271, 1998. doi:10.1016/S0378-3758(98)00064-0.

42 Tim Seppelt. An Algorithmic Meta Theorem for Homomorphism Indistinguishability. In *49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024)*, volume 306 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 82:1–82:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2024.82.

43 B. Weisfeiler and A. Leman. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia, Seriya 2*, 9:12–16, 1968. An english translation due to Grigory Ryabov is available at https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf.

44 Faried Abu Zaid, Erich Grädel, Martin Grohe, and Wied Pakusa. Choiceless polynomial time on structures with small abelian colour classes. In *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part I*, volume 8634 of *Lecture Notes in Computer Science*, pages 50–62. Springer, 2014. doi:10.1007/978-3-662-44522-8_5.

A Construction of One-Way Switches

We construct our one-way switches based on the CFI-construction, and the proof of the properties heavily uses the bijective pebble game. We thus start with a short introduction to these.

The Bijective Pebble Game. The question whether C^{k+1} or k-WL can distinguish structures $\mathfrak{A}$ and $\mathfrak{B}$ has another characterization in terms of the so-called *bijective* $(k+1)$*-pebble game*. In this game, there are two players: Spoiler and Duplicator. Game positions are partial maps $\mathbf{g} \mapsto \mathbf{h}$ between G and H, where both tuples contain at most $k+1$ elements. We also sometimes identify such partial maps with the set $P = \{g_i \mapsto h_i \colon i \leq |\mathbf{g}|\}$.

We think of these maps as $k+1$ pairs of corresponding pebbles placed in the two graphs. If such a partial map is not a partial isomorphism, i.e., not an isomorphisms on the induced subgraphs, Spoiler wins immediately.

Otherwise, at the beginning of each turn, Spoiler picks up one pebble pair, either from the board if all $k+1$ pairs are placed, or from the side if there are pebble pairs left. Duplicator responds by giving a bijection $\varphi \colon V(G) \to V(H)$ between the two graphs. Spoiler then places the pebble pair they picked up on a pair $(g, \varphi(g))$ of vertices of their choice. The game then continues in the resulting new position.

We say that Spoiler wins if the graphs have differing cardinality or they can reach a position that is no longer a partial isomorphism (and thus win immediately). Duplicator wins the game if Duplicator can find responses to Spoiler's moves indefinitely.

Lemma 5 now has the following extension:

▶ **Lemma 24** ([11], [26]). *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two relational structures of arity at most k, and $\mathbf{a} \in V(\mathfrak{A})^k$ and $\mathbf{b} \in V(\mathfrak{B})^k$ two tuples of vertices. Then the following are equivalent:*

(i) *Duplicator has a winning strategy in position $\mathbf{a} \mapsto \mathbf{b}$ of the bijective $(k+1)$-pebble game between $\mathfrak{A}$ and $\mathfrak{B}$,*

(ii) *for every C^{k+1}-formula $\varphi(x_1, \ldots, x_k)$, we have $(G, \mathbf{g}) \models \varphi$ if and only if $(H, \mathbf{h}) \models \varphi$,*

(iii) *the stable colors computed by k-WL for the tuples $\mathbf{a}$ and $\mathbf{b}$ agree.*

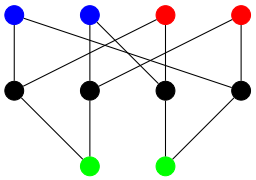


Figure 5 A CFI-gadget for a vertex of degree 3, consisting of four inner vertices and three outer pairs.

The CFI-Construction. CFI-graphs are certain graphs with high Weisfeiler-Leman dimension [11]. To construct them, we start with a *base graph* G, which is a connected simple graph, and a function $f\colon E(G) \to \mathbb{F}_2$. For a vertex $v \in V(G)$, we denote the set of edges incident to v by $E[v] \coloneqq \{uv\colon v \in N_G(v)\} \subseteq E(G)$. Now, to construct the CFI-graph $\mathrm{CFI}(G, f)$, we replace each vertex $v \in V(G)$ by a gadget X_v which consists of *inner vertices* $I_v \coloneqq \{v\} \times \{\mathbf{x} \in \mathbb{F}_2^{E[v]}\colon \sum \mathbf{x} = 0\}$ and *outer vertices* $\{v\} \times \{(e, i)\colon e \in E[v], i \in \mathbb{F}_2\}$. Inside each gadget, the inner and outer vertices each form an independent set, and an inner vertex $(v, \mathbf{x})$ and outer vertex (v, e, i) are connected by an edge if and only if $\mathbf{x}_e = i$. The resulting gadget for a vertex of degree 3 is depicted in Figure 5.

Next, we define the edge set between different gadgets. For every edge $e = uv \in E(G)$, we connect the outer vertices (u, e, i) and (v, e, j) if and only if $i + j = f(e)$ and add no further edges. Thus, corresponding outer vertex pairs $(u, e, \cdot)$ and $(v, e, \cdot)$ are always connected by a matching, which is either *untwisted* if $f(e) = 0$, or *twisted* if $f(e) = 1$.

Finally, we define a vertex coloring on this graph. For every vertex v, we turn the set I_v of inner vertices into a color class of size $2^{d(v)-1}$. Moreover, we turn each outer pair $\{(v, e, 0), (v, e, 1)\}$ into a color class of size 2. This finishes the construction of CFI-graphs.

It turns out that for two functions $f, g\colon E(G) \to \mathbb{F}_2$, we have $\mathrm{CFI}(G, f) \cong \mathrm{CFI}(G, g)$ if and only if $\sum f = \sum g$, meaning that every even number of twists cancels out. Thus, we also write $\mathrm{CFI}(G, 0)$ and $\mathrm{CFI}(G, 1)$ for the *untwisted* and *twisted* CFI-graphs over the base graph G.

To understand the power of the Weisfeiler-Leman algorithm on CFI-graphs, it is convenient to study *tree-width*, which is a graph parameter that intuitively measures how far a graph is from being a tree. In this work, we do not need the formal definition of tree-width, and refer to [8]. The power of the Weisfeiler-Leman algorithm to distinguish CFI-graphs can now conveniently be expressed in terms of the tree-width of the base graphs, see [22].

▶ **Lemma 25.** *For every base graph G of tree-width $\mathrm{tw}(G) \geq 2$, we have*

$$\text{WL-dim}(\mathrm{CFI}(G, 0)) = \text{WL-dim}(\mathrm{CFI}(G, 1)) = \mathrm{tw}(G).$$

Construction of the One-Way Switches. We start by defining a base graph. Consider a wall graph consisting of $k - 1$ rows of k bricks each. Then, we attach a new vertex v to the two upper corner vertices of the first row. The resulting graph B_k is depicted in Figure 6.

▶ **Lemma 26.** *The graph B_k has tree-width $k + 1$, while $B_k - v$ has tree-width k.*

Now, we are ready to construct our one-way switches. In our proofs, we actually need a more explicit version of Lemma 20 in order to precisely control the expressive power fo the Weisfeiler-Leman algorithm on the whole graph in terms of its expressive power on the individual gadgets:

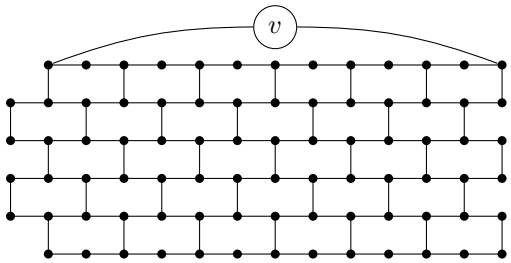


Figure 6 The base graph B_6 of the CFI-graphs underlying our one-way switches.

▶ **Lemma 27** (compare [19, Lemma 14]). *For every $k \geq 2$, there is a colored graph O^k with 4-bounded color classes, called k-*one-way switch*, with an input pair $\{y_1, y_2\}$ and an output pair $\{x_1, x_2\}$ satisfying the following properties:*

1. *The graph O^k_{split} obtained by splitting the input pair $\{y_1, y_2\}$ is identified by k-*WL*.*
2. *k-*WL *splits the output pair $\{x_1, x_2\}$ of O^k_{split}.*
3. *There is no automorphism of O^k exchanging the output vertices x_1 and x_2.*

Furthermore, there are sets of positions in the bijective $(k+1)$-pebble game between O^k and itself, called trapped *and* twisted *such that*

4. *every trapped or twisted position is a partial isomorphism,*
5. *Duplicator can avoid non-trapped positions from trapped ones and non-twisted positions from twisted ones,*
6. *for every trapped position $\mathbf{a} \mapsto \mathbf{b}$, the position $\mathbf{a}x_1 \mapsto \mathbf{b}x_1$ is also trapped,*[1]
7. *for every twisted position $\mathbf{a} \mapsto \mathbf{b}$, the position $\mathbf{a}x_1 \mapsto \mathbf{b}x_2$ is also twisted.*
8. *the positions $y_1y_2 \mapsto y_1y_2$ and $y_1y_2 \mapsto y_2y_1$ are both trapped and twisted,*
9. *every subposition of a trapped position is trapped, and every subposition of a twisted position is twisted*

Proof. Let O^k be the (untwisted) CFI-graph of B_k, but with a CFI-gadget of degree 3 added for the vertex v instead of a gadget of degree 2. This leaves one outer pair of this gadget free which we use as our output pair $\{x_1, x_2\}$. Furthermore, we use one of the other two outer pairs of this same CFI-gadget as the input pair $\{y_1, y_2\}$.

Now, if we fix the output pair $\{x_1, x_2\}$ by individualizing one of the two vertices, the resulting graph corresponds to the usual CFI-graph of H, while switching the pair $\{x_1, x_2\}$ corresponds to the twisted CFI-graph of H. In particular, as these graphs are not isomorphic, there is no automorphism of O^k switching the pair $\{x_1, x_2\}$, which proves Property 3.

Moreover, splitting the input pair $\{y_1, y_2\}$ has the same effect to the power of k-WL as removing one of the two edges incident to v in the base graph B_k has. When removing this edge in the base graph, the resulting graph is essentially equivalent to the CFI-graph of the $k \times (k+1)$-wall graph with one corner vertex replaced by a CFI-gadget of degree 3 instead of 2. Because exchanging the two vertices of the free outer pair of this degree-3 gadget interchanges the twisted and untwisted CFI-graphs over the base graph, and k-WL can distinguish CFI-graphs from all other graphs, the resulting graph is identified by k-WL. This proves Property 1.

[1] If the position $\mathbf{a}x_1 \mapsto \mathbf{b}x_1$ contains more than $k+1$ pebbles, this means that every subposition on at most $k+1$ pebbles is trapped.

To show Property 2, we start the bijective $(k+1)$-pebble game in position $x_1 \mapsto x_2$. Then, Spoiler uses the usual strategy of pebbling a wall which they then move from one side of the wall graph to the other. But because the game started in position $x \mapsto x'$, the two graphs the game is played on differ in a twist which will finally force Duplicator to lose.

Now, consider again the original graph O^k without splitting the input pair. On this graph, we can extend every winning position for Duplicator in the bijective k-pebble game between the untwisted CFI-graph $\mathrm{CFI}(B_k, 0)$ and the twisted CFI-graph $\mathrm{CFI}(B_k, 1)$ to a position in the bijective k-pebble game between O^k and itself which is compatible with $x_1 \mapsto x_2$. Similarly, we can extend every winning position for Duplicator in the bijective k-pebble game between the untwisted CFI-graph $\mathrm{CFI}(B_k, 0)$ and itself to a position between O^k and itself which is compatible with $x_1 \mapsto x_1$.

We call the former positions *twisted* and the latter positions *trapped*. Properties 4, 6, 7 and 9 are then immediate, and Property 5 follows from Lemma 25 together with Lemma 26.

Because v lies on a cycle in B_k, there exists an automorphism of $\mathrm{CFI}(B_k)$ which twists both outer pairs of the gadget corresponding to v. Lifting this automorphism to O^k yields an automorphism switching y_1 and y_2 whilst fixing x_1 and x_2. This proves Property 8. ◀

Finite Variable Counting Logics with Restricted Requantification

Simon Raßmann
TU Darmstadt, Germany

Georg Schindling
TU Darmstadt, Germany

Pascal Schweitzer
TU Darmstadt, Germany

Abstract

Counting logics with a bounded number of variables form one of the central concepts in descriptive complexity theory. Although they restrict the number of variables that a formula can contain, the variables can be nested within scopes of quantified occurrences of themselves. In other words, the variables can be requantified. We study the fragments obtained from counting logics by restricting requantification for some but not necessarily all the variables.

Similar to the logics without limitation on requantification, we develop tools to investigate the restricted variants. Specifically, we introduce a bijective pebble game in which certain pebbles can only be placed once and for all, and a corresponding two-parametric family of Weisfeiler-Leman algorithms. We show close correspondences between the three concepts.

By using a suitable cops-and-robber game and adaptations of the Cai-Fürer-Immerman construction, we completely clarify the relative expressive power of the new logics.

We show that the restriction of requantification has beneficial algorithmic implications in terms of graph identification. Indeed, we argue that with regard to space complexity, non-requantifiable variables only incur an additive polynomial factor when testing for equivalence. In contrast, for all we know, requantifiable variables incur a multiplicative linear factor.

Finally, we observe that graphs of bounded tree-depth and 3-connected planar graphs can be identified using no, respectively, only a very limited number of requantifiable variables.

2012 ACM Subject Classification Theory of computation → Finite Model Theory; Theory of computation → Complexity theory and logic

Keywords and phrases Requantification, Finite variable counting logics, Weisfeiler-Leman algorithm

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.14

Related Version *Full Version*: `https://arxiv.org/abs/2411.06944`

Funding The research of the second and third author leading to these results has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (EngageS: grant agreement No. 820148).

1 Introduction

Descriptive complexity is a branch of finite model theory that essentially aims at characterizing how difficult logical expressions need to be in order to capture particular complexity classes. While we are yet to find or rule out a logic capturing the languages in the complexity class P, there is an extensive body of work regarding the descriptive complexity of problems within P. Most notably, there is the work of Cai, Fürer, and Immerman [3] which studies a particular fragment of first-order logic. This is the fragment C^k in which counting quantifiers are introduced into the logic, but the number of variables is restricted to being at most k. The seminal result in [3] shows that this logic fails to define certain graphs up to isomorphism, which in turn proves that *inflationary fixed-point logic with counting* $\mathsf{IFP{+}C}$ fails to capture P.

Although the fragment C^k restricts the number of variables, it is common for variables to be reused within a single logical formula. In particular, variables can be nested within scopes of quantified occurrences of themselves. In other words, they can be requantified. In our work, we are interested in understanding what happens if we limit the ability to reuse variables through requantification. In fact, we may think of reusability as a resource (in the vein of time, space, communication, proof length, advice etc.) that should be employed economically.

It turns out that the ability to limit requantification provides us with a more detailed lens into the landscape of descriptive complexities within P, much in the fashion of fine-grained complexity theory.

Results and techniques. Let us denote by $\mathsf{C}^{(k_1,k_2)}$ the fragment of first-order logic with counting quantifiers in which the formulas have at most k_1 variables that may be requantified and at most k_2 variables that may not be requantified.

First, we show that many of the traditional techniques of treating counting logics can be adapted to the setting of limited requantification. Specifically, it is well known that there is a close ternary correspondence between the logic C^k, the combinatorial bijective k-pebble game, and the famous $(k-1)$-dimensional Weisfeiler-Leman algorithm [3, 22, 26]. We develop versions of the game and the algorithm that also have a limit on the reusability of resources. For the pebble game, a limit on requantification translates into pebbles that cannot be picked up anymore, once they have been placed. For the Weisfeiler-Leman algorithm, the limit on requantification translates into having some dimensions that "cannot be reused". In fact the translation to the algorithmic viewpoint is not as straightforward as one might hope at first. Indeed, we do not know how to define a restricted version of the classical Weisfeiler-Leman algorithm that corresponds to the logic $\mathsf{C}^{(k_1,k_2)}$. However, we circumvent this problem by employing the oblivious Weisfeiler-Leman algorithm (OWL). This variant is often used in the context of machine learning. In fact, Grohe [17] recently showed that $k+1$-dimensional OWL is in fact exactly as powerful as k-dimensional (classical) WL. We develop a resource-reuse restricted version of the oblivious algorithm and prove equivalence to our logic. Indeed, we formally prove precisely matching correspondences between the limited requantification, limited pebble reusability, and the limited reusable dimensions (Theorem 6).

Next, we conclusively clarify the relation between the logics within the two-parametric family $\mathsf{C}^{(k_1,k_2)}$. We show that in most cases limiting the requantifiability of a variable strictly reduces the power of the logic. We argue that no amount of requantification-restricted variables is sufficient to compensate the loss of an unrestricted variable. However, these statements are only true if at least some requantifiable variable remains. In fact, exceptionally, $\mathsf{C}^{(1,k_2)}$ is strictly less expressive than $\mathsf{C}^{(0,k_2')}$ whenever $k_2' > 2k_2$ (Theorem 13). To show the separation results, we adapt a construction of Fürer [13] and develop a cops-and-robber game similar to those in [12, 19]. In this version, some of the cops may repeatedly change their location, while others can only choose a location once and for all. Using another graph construction, we rule out various a priori tempting ideas concerning normal forms in $\mathsf{C}^{(k_1,k_2)}$. To this end we show that formulas in the logics can essentially become as complicated as possible, having to repeatedly requantify all of the requantifiable variables an unbounded number of times, before using a non-requantifiable variable (Corollary 16). In terms of the pebble game, it seems a priori unclear when an optimal strategy would employ the non-reusable pebbles. However, the corollary says that in general one has to conserve the non-reusable pebbles for possibly many moves until a favorable position calls for them.

Having gone through the technical challenges that come with the introduction of reusability, puts us into a position to discuss the implications. Indeed, as our main result, we argue that our finer grained view on counting logics through restricted requantification has beneficial algorithmic implications. Specifically, we show that equivalence with respect to the logic $\mathsf{C}^{(k_1,k_2)}$ can be decided in polynomial time with a space complexity of $O\big(n^{k_1}\log n\big)$, hiding quadratic factors depending only on k_1 and k_2 (Theorem 24). This shows that while the requantifiable variables each incur a multiplicative linear factor in required space, the restricted variables only incur an additive polynomial factor. In particular, equivalence with respect to the logics $\mathsf{C}^{(0,k_2)}$ can be decided in logarithmic space. To show these statements, we leverage the fact that, because non-requantifiable variables cannot simultaneously occur free and bound, the $\mathsf{C}^{(k_1,k_2)}$-type of a variable assignment does not depend on the $\mathsf{C}^{(k_1,k_2)}$-type of assignments which disagree regarding non-requantifiable variables. Moreover, we use ideas from an algorithm of Lindell, which computes isomorphism of trees in logarithmic space [30] to implement the iteration steps of our algorithm. Generally, we believe the new viewpoint may be of interest in particular for applications in machine learning, where the WL-hierarchy appears to be too coarse for actual applications with graph neural networks (see for example [1, 2, 31, 40]). In the process of the space complexity proof, we also show that the iteration number of the resource-restricted Weisfeiler-Leman algorithm described above is at most $(k_2+1)n^{k_1}-1$ (Corollary 19).

Justifying the new concepts of restricted reusability, we observe that there are interesting graph classes that are identified by the logics $\mathsf{C}^{(k_1,k_2)}$. We argue that $\mathsf{C}^{(0,d+1)}$ identifies all graphs of tree-depth at most d (Theorem 27) and that $\mathsf{C}^{(2,2)}$ identifies all 3-connected planar graphs (Theorem 33).

Outline of the paper. After briefly providing necessary preliminaries (Section 2) we formally introduce the logics $\mathsf{C}^{(k_1,k_2)}$, the pebble game with non-reusable pebbles, the (k_1,k_2)-dimensional oblivious Weisfeiler-Leman algorithm, and prove the correspondence theorem between them (Section 3). We then relate the power of the logics to each other and rule out certain normal forms (Section 4). We then analyze the space complexity (Section 5) and finally provide two classes of graphs that are identified by our logics (Section 6).

Further related work. In addition to the references above, let us mention related investigations. Over time, a large body of work on descriptive complexity has evolved. For insights into fundamental results regarding bounded variable logics, we refer to classic texts [25, 26, 33, 34]. However, highlighting the importance of the counting logics C^k, let us at least mention the Immerman-Vardi theorem [24, 39]. It says that on ordered structures, *least fixed-point logic* LFP captures P. Since LFP has the same expressive power as $\mathsf{IFP{+}C}$ on ordered structures, also $\mathsf{IFP{+}C}$, whose expressive power is closely related to the expressive power of the logics C^k, captures P. We should also mention the work of Hella [22] introducing the bijective k-pebble game which forms the basis for our resource restricted versions.

(Counting logics on graph classes) Because of the close correspondence between the logic C^k and the $(k-1)$-dimensional Weisfeiler-Leman algorithm, our investigations are closely related to the notion of the *Weisfeiler-Leman dimension* of a graph defined in [15]. Given a graph G this is the least number of variables k such that C^{k+1} identifies G. In particular, on every graph class of bounded Weisfeiler-Leman dimension, the corresponding finite variable counting logic captures isomorphism. Graph classes with bounded Weisfeiler-Leman dimension include graphs with a forbidden minor [14] and graphs of bounded rank-width (or equivalently clique width) [20], which in both cases is also shown to imply

that IFP+C captures P on these classes. For a comprehensive survey we refer to [27]. Our observations for planar graphs follow from techniques bounding the number of variables required for the identification of planar graphs [28]. Other recent classes not already captured by the results on excluded minors and rank-width include, for example, some polyhedral graphs [29], some strongly regular graphs [4], and permutation graphs [21].

(Logic and tree decompositions) In [9] and independently [8] it was shown that C^k-equivalence is characterized by homomorphism counts from graphs of tree-width at most $k-1$. Likewise, homomorphism counts from bounded tree-depth graphs characterize equivalence in counting logic of bounded quantifier-rank [16]. Recently, these results were unified to characterize logical equivalence in finite variable counting logics with bounded quantifier-rank in terms of homomorphism counts [12].

(Space complexity) Ideas underlying Lindell's logspace algorithm for tree isomorphism have also been used in the context of planar graphs [6] and more generally bounded genus graphs [10]. Similar results exist for classes of bounded tree-width [5, 11].

(Further recent results) Let us mention some quite recent results in the vicinity of our work that cannot be found in the surveys mentioned above. Regarding the quantifier-rank within counting logics, there is a recent superlinear lower bound [19] improving Fürer's linear lower bound construction [13]. Further, very recent work on logics with counting includes results on rooted unranked trees [23] and inapproximability of questions on unique games [35]. Finally, there has been a surge in research on descriptive complexity within the context of machine learning (see [17, 36, 37]).

2 Preliminaries

General notation. For $n \in \mathbb{N}_+$ we use $[n]$ to denote the n-element set $\{1, \dots, n\}$. We use the notation $\{\!\{v_1, \dots, v_n\}\!\}$ for *multisets*. For $k_1, k_2 \in \mathbb{N}_+$, we fix the variable sets $[x_{k_1}] \coloneqq \{x_1, \dots, x_{k_1}\}$, $[y_{k_2}] \coloneqq \{y_1, \dots, y_{k_2}\}$, and $[x_{k_1}, y_{k_2}] \coloneqq \{x_1, \dots, x_{k_1}, y_1, \dots, y_{k_2}\}$. Given a set V, a *partial function* $\alpha\colon [x_{k_1}, y_{k_2}] \rightharpoonup V$ assigns to every variable $z \in [x_{k_1}, y_{k_2}]$ at most one element $\alpha(z) \in V$. If α does not assign an element to z, we write $\alpha(z) = \bot$. Also, we write $\mathrm{im}(\alpha)$ for the *image* of α. With a finite set V and $\alpha\colon [x_{k_1}, y_{k_2}] \rightharpoonup V$ we associate the total function $\overline{\alpha}\colon [x_{k_1}, y_{k_2}] \to V \mathbin{\dot\cup} \{\bot\}$, which we also view as a $[x_{k_1}, y_{k_2}]$-indexed $(k_1 + k_2)$-tuple. For $z \in [x_{k_1}, y_{k_2}]$ and $v \in V$, the function $\alpha[z/v]$ is defined as α but with $\alpha(z)$ replaced by v.

Graphs. A graph is a pair $G = (V(G), E(G))$ consisting of a finite set $V(G)$ of *vertices* and a set $E(G) \subseteq \binom{V(G)}{2}$ of *edges*. We write $|G|$ for the number of vertices, called the *order* of G. For a vertex $v \in V(G)$ we define the *neighborhood* $N_G(v) \coloneqq \{w \in V(G) : \{v, w\} \in E(G)\}$ and the *degree* $d_G(v) \coloneqq |N_G(v)|$ of v in G. We call v *universal* in G if $N_G(v) = V(G) \setminus \{v\}$. A *colored graph* consists of a graph G and a coloring function $\chi\colon V(G) \to C$ with a finite, ordered set C of *colors*. For a colored graph G and vertices $v_1, \dots, v_n \in V(G)$ the graph $G_{(v_1,\dots,v_n)}$ is obtained by assigning new and distinct colors to the vertices $v_1, \dots, v_n$ in G. The vertices $v_1, \dots, v_n$ are then called *individualized*. An isomorphism of (colored) graphs G and H is a bijection $\varphi\colon V(G) \to V(H)$ that preserves edges, non-edges, and vertex-colors.

We denote the complete graph on n vertices by K_n, that is, the graph with vertex set $[n]$ and all possible edges included. A star of degree n is a graph consisting of one universal vertex of degree n and its neighbors of degree 1.

First-order logic with counting. First-order logic with counting C is an extension of first-order logic by counting quantifiers $\exists^{\geq k}$ for all $k \in \mathbb{N}$. These intuitively state that there exist at least k distinct vertices satisfying the formula that follows.

Over the language of colored graphs with variable set $\mathcal{V}$, formulas are inductively built up from atomic formulas (for stating equality or adjacency of vertices as well as for stating that a vertex has a given vertex color) via negation ($\neg$), conjunction ($\wedge$), disjunction ($\vee$), implication ($\rightarrow$), and quantification over vertices via $\forall$, $\exists$ and the counting quantifiers $\exists^{\geq k}$. For a colored graph G, a variable assignment $\alpha\colon \mathcal{V} \to V(G)$, and a formula $\varphi \in \mathsf{C}$, we write $G, \alpha \models \varphi$ if the graph G together with the variable assignment α *satisfies* the formula φ.

The *quantifier-rank* $\mathrm{qr}(\varphi)$ of a formula φ is the maximum depth of nested quantifiers in the formula. The set of *free variables* $\mathrm{free}(\varphi)$ of a formula φ is defined as the set of all variables that occur outside the scope of a corresponding quantifier in φ. If $\mathrm{free}(\varphi) = \emptyset$ the formula φ is called a *sentence*. The set of *bound variables* $\mathrm{bound}(\varphi)$ is the set of all variables that occur quantified in φ. If we restrict to a finite set of $k \in \mathbb{N}_+$ variables, the resulting logic is called *k-variable counting logic* and denoted by C^k. For $r \in \mathbb{N}$ the *quantifier-rank-r counting logic* C_r is obtained by restricting formulas in C to quantifier-rank at most r. The *k-variable quantifier-rank-r counting logic* is defined as $\mathsf{C}^k_r \coloneqq \mathsf{C}^k \cap \mathsf{C}_r$.

As a general reference on finite variable logics, we refer to [33].

The Weisfeiler-Leman algorithm. Let $k \geq 1$ and G be a colored graph. The *k-dimensional Weisfeiler-Leman algorithm* (short k-WL) iteratively computes a coloring of the k-tuples of vertices of G. We also view the k-tuples as total functions $\overline{\alpha}\colon \{x_1, \dots, x_k\} \to V(G)$. Initially, each tuple $\overline{\alpha} \in V(G)^k$ is colored by $\mathrm{wl}_k^{(0)}(G, \overline{\alpha}) \coloneqq \mathrm{atp}_k(G, \overline{\alpha})$. Here, $\mathrm{atp}_k(G, \overline{\alpha})$ is the *atomic type* of $\overline{\alpha}$ in G, i.e., the set of all atomic formulas with variables in $\{x_1, \dots, x_k\}$ satisfied by the colored graph $G[\mathrm{im}(\alpha)]$ together with the assignment α. For every $r \in \mathbb{N}$, we then inductively set

$$\mathrm{wl}_k^{(r+1)}(G, \overline{\alpha}) \coloneqq (\mathrm{wl}_k^{(r)}(G, \overline{\alpha}); \{\!\{(\mathrm{wl}_k^{(r)}(G, \overline{\alpha}[x_i/u]))_{i\in[k]} : u \in V(G)\}\!\})$$

whenever $k \geq 2$, but for the case $k = 1$ we set

$$\mathrm{wl}_k^{(r+1)}(G, \overline{\alpha}) \coloneqq (\mathrm{wl}_k^{(r)}(G, \overline{\alpha}); \{\!\{\mathrm{wl}_k^{(r)}(G, u) : u \in N_G(\overline{\alpha})\}\!\}).$$

We write $\mathrm{wl}_k^{(r)}(G)$ for the coloring of all k-tuples of vertices of G assigning $\mathrm{wl}_k^{(r+1)}(G, \overline{\alpha})$ to $\overline{\alpha}$. Since the definition of $\mathrm{wl}_k^{(r+1)}(G, \overline{\alpha})$ includes the color $\mathrm{wl}_k^{(r)}(G, \overline{\alpha})$ of the previous iteration, the coloring $\mathrm{wl}_k^{(r+1)}(G)$ *refines* the coloring $\mathrm{wl}_k^{(r)}(G)$. That is, whenever $\mathrm{wl}_k^{(r+1)}(G, \overline{\alpha_1}) = \mathrm{wl}_k^{(r+1)}(G, \overline{\alpha_2})$ for $\overline{\alpha_1}, \overline{\alpha_2} \in V(G)^k$, then we also have $\mathrm{wl}_k^{(r)}(G, \overline{\alpha_1}) = \mathrm{wl}_k^{(r)}(G, \overline{\alpha_2})$. Since there are exactly $|V(G)|^k$-many k-tuples of vertices, there exists an $r < |V(G)|^k$ such that $\mathrm{wl}_k^{(r)}(G)$ induces the same partition of color classes as $\mathrm{wl}_k^{(r+1)}(G)$. It follows from the definition of the refinement that this implies $\mathrm{wl}_k^{(r)}(G)$ induces the same partition as $\mathrm{wl}_k^{(r')}(G)$ for all $r' \geq r$. In this case we say that k-WL *stabilizes* after at most r iterations on the graph G. If r is minimal with this property, we write $\mathrm{wl}_k^{(\infty)}(G) \coloneqq \mathrm{wl}_k^{(r+1)}(G)$ and call $\mathrm{wl}_k^{(\infty)}(G)$ the *stable coloring*. For a second colored graph H, we say that k-WL *distinguishes* G and H after r iterations if there exists a color c such that $|\{\overline{\alpha} \in V(G)^k : \mathrm{wl}_k^{(r)}(G, \overline{\alpha}) = c\}| \neq |\{\overline{\beta} \in V(H)^k : \mathrm{wl}_k^{(r)}(H, \overline{\beta}) = c\}|$.

Besides the classical k-dimensional Weisfeiler-Leman algorithm, there also exists a variant, called the *$(k+1)$-dimensional oblivious Weisfeiler-Leman algorithm* (short $(k+1)$-OWL). This variant colors $(k+1)$-tuples of vertices, which we again view as total functions $\overline{\alpha}\colon \{x_1, \dots, x_{k+1}\} \to V(G)$. Initially, all tuples are colored by their atomic type: $\mathrm{owl}_{k+1}^{(0)}(G, \overline{\alpha}) \coloneqq \mathrm{atp}_{k+1}(G, \overline{\alpha})$. Then, this coloring is iteratively refined by setting

$$\mathrm{owl}_{k+1}^{(r+1)}(G, \overline{\alpha}) \coloneqq (\mathrm{owl}_{k+1}^{(r)}(G, \overline{\alpha}); \{\!\{\mathrm{owl}_{k+1}^{(r)}(G, \overline{\alpha}[x_i/u]) : u \in V(G)\}\!\}_{i\in[k+1]}).$$

The stable coloring $\mathrm{owl}_{k+1}^{(\infty)}(G)$ and the notion of distinguishing graphs is defined as for k-WL.

It turns out that k-WL and $(k+1)$-OWL have the same distinguishing power.

▶ **Lemma 1** ([17, Lemma A.1, Corollary V.7]). *Let G and H be graphs, $\overline{\alpha} \in V(G)^{k+1}$ and $\overline{\beta} \in V(H)^{k+1}$. Then the following are equivalent for every $r \in \mathbb{N}$:*

1. $\mathrm{owl}^{(r)}_{k+1}(G, \overline{\alpha}) = \mathrm{owl}^{(r)}_{k+1}(H, \overline{\beta})$,
2. $\mathrm{atp}_{k+1}(G, \overline{\alpha}) = \mathrm{atp}_{k+1}(H, \overline{\beta})$ *and for all* $i \in [k+1]$, *we have* $\mathrm{wl}^{(r)}_k(G, \overline{\alpha}_{\neq i}) = \mathrm{wl}^{(r)}_k(H, \overline{\beta}_{\neq i})$, *where* $\overline{\alpha}_{\neq i}$ *is the k-tuple obtained from α by deleting the i-th entry.*

Moreover, two graphs are distinguished by k-WL if and only if they are distinguished by $(k+1)$-OWL.

3 Finite Variable Counting Logics with Restricted Requantification

When working in the logic C^k, it is often necessary to *requantify* variables in order to express certain properties. We introduce finite variable first-order logic with counting quantifiers and *restricted requantification* to study this issue. We then define an Ehrenfeucht-Fraïssé-style game as an important tool for the analysis of the newly introduced logic by game-theoretic arguments. Finally, we devise a variant of k-OWL that precisely captures the expressive power of the logic and game and prove a characterization that closely ties the reusable and non-reusable resources among these objects. First, we give a precise definition of *requantification*.

▶ **Definition 2.** *Consider the counting logic C over a set of variables $\mathcal{V}$. A variable $x \in \mathcal{V}$ is said to be requantified in a formula $\varphi \in \mathsf{C}$ if either $x \in \mathrm{free}(\varphi) \cap \mathrm{bound}(\varphi)$ or if there exist a subformula $Qx\psi$ of φ and in turn a subformula $Q'x\chi$ of ψ with $Q, Q' \in \{\forall, \exists\} \cup \{\exists^{\geq n} : n \in \mathbb{N}\}$. We define the logic $\mathsf{C}^{(k_1,k_2)}$ as the fragment of C over the fixed variable set $\mathcal{V} = [x_{k_1}, y_{k_2}]$ consisting of those formulas in which the variables from $\{y_1, \dots, y_{k_2}\}$ are not requantified. The fragment of $\mathsf{C}^{(k_1,k_2)}$ with quantifier-rank at most $r \in \mathbb{N}$ is denoted by $\mathsf{C}^{(k_1,k_2)}_r$.*

▶ **Example 3.** Consider the following $\mathsf{C}^{(2,1)}_3$ formula:

$$\big(\exists y_1 \neg E(x_2, y_1)\big) \wedge \exists^{\geq 4} x_1 \Big(E(x_2, x_1) \wedge \exists y_1 \big(\neg E(x_1, y_1)\big) \wedge \forall x_2 \big(\neg E(x_2, x_1) \rightarrow \exists^{\geq 3} x_1 E(x_1, x_2)\big)\Big)$$

expressing that the vertex x_2 is not universal and has at least four non-universal neighbors such that every non-neighbor of those has degree at least three. The variable x_2 is requantified in this formula since it occurs free and bound. The variable x_1 is requantified because the subformula $\exists^{\geq 3} x_1 E(x_1, x_2)$ occurs within the scope of the outermost quantification $\exists^{\geq 4} x_1$. The variable y_1 however is not requantified since neither of its quantifications occurs in the scope of the other.

The central question we will investigate in the following is how the non-requantifiability restriction affects the expressive power of the logic $\mathsf{C}^{(k_1,k_2)}$. To this end, we use the notation $\mathsf{C}^{(k_1,k_2)} \preceq \mathsf{C}^{(k_1',k_2')}$ if every pair of graphs distinguished by $\mathsf{C}^{(k_1,k_2)}$ is also distinguished by $\mathsf{C}^{(k_1',k_2')}$. We also write $\mathsf{C}^{(k_1,k_2)} \equiv \mathsf{C}^{(k_1',k_2')}$ if the two logics distinguish exactly the same pairs of graphs and $\mathsf{C}^{(k_1,k_2)} \prec \mathsf{C}^{(k_1',k_2')}$ if the relation is strict. In the case of unrestricted requantification (i.e. $k_2 = 0$) it is clear that $\mathsf{C}^{(k_1,0)} \preceq \mathsf{C}^{(k_1+1,0)}$. In this terminology, the central result of [3] is that this relation is strict for all $k_1 \in \mathbb{N}$. For the case of restricted requantification we make the simple observation that having more variables is at least as expressive as having fewer variables (independent of their ability to be requantified). We also observe that requantifiable variables are at least as expressive as non-requantifiable variables. That is, for all $k_1, k_2 \in \mathbb{N}$ with $k_1 + k_2 \geq 1$ it holds that $\mathsf{C}^{(k_1,k_2)} \preceq \mathsf{C}^{(k_1,k_2+1)} \preceq \mathsf{C}^{(k_1+1,k_2)}$.

Also, observe that having only non-requantifiable variables (i.e. $k_1 = 0$) bounds the quantifier-rank to at most k_2 and in turn every sentence of quantifier-rank at most k_2 can be rewritten using at most k_2 non-requantifiable variables. More precisely, we have $\mathsf{C}^{(0,k_2)} \equiv \mathsf{C}_{k_2}$.

Next, we establish an Ehrenfeucht-Fraïssé-style game which closely corresponds to the power of the previously defined logics with respect to distinguishing graphs. The game is a variant of the bijective pebble game introduced in [22] with the additional restriction that some pebbles may not be picked up again once placed.

▶ **Definition 4.** *Suppose $k_1, k_2 \in \mathbb{N}$ and $k_1 + k_2 \geq 1$. For colored graphs G and H, we define the* bijective (k_1, k_2)-pebble game $\mathrm{BP}_{(k_1,k_2)}(G, H)$ *as follows:*

The game is played by the players Spoiler, denoted by (S), and Duplicator, denoted by (D), with one pair of pebbles for each variable in $[x_{k_1}, y_{k_2}]$. The pebble pairs in $[x_{k_1}]$ are called reusable *and the pebble pairs in $[y_{k_2}]$ are called* non-reusable*.*

*The game proceeds in rounds, each of which is associated with a pair of partial functions $\alpha\colon [x_{k_1}, y_{k_2}] \rightharpoonup V(G)$, $\beta\colon [x_{k_1}, y_{k_2}] \rightharpoonup V(H)$ with $\mathrm{dom}(\alpha) = \mathrm{dom}(\beta)$. We call such a pair of partial functions a (k_1, k_2)-*configuration *on the pair G, H. These functions indicate the placement of the pebble pairs on the graphs. For a pebble pair $z \in [x_{k_1}, y_{k_2}]$ and vertices $v \in V(G), w \in V(H)$ we have $\alpha(z) = v, \beta(z) = w$ whenever the two pebbles of the pair z are placed on v and w, respectively. If not specified otherwise, both games start from the empty configuration given by $\mathrm{dom}(\alpha) = \mathrm{dom}(\beta) = \emptyset$. One round of the game with current configuration (α, β) consists of the following steps:*

1. *(S) picks up a pebble pair $z \in [x_{k_1}, y_{k_2}]$ such that $z \in [x_{k_1}]$ or $\alpha(z)$ is undefined. If no such z exists, the winning condition is checked directly.*
2. *(D) chooses a bijection $f\colon V(G) \to V(H)$.*
3. *(S) chooses $w \in V(G)$ and $f(w) \in V(H)$ to be pebbled with the pair z.*
4. *The new configuration is given by $(\alpha[z/w], \beta[z/f(w)])$.*

The winning conditions are as follows:

- *(S) wins immediately, if the initial configuration (α, β) does not induce a partial isomorphism. That is, the function $h\colon \mathrm{im}(\alpha) \to \mathrm{im}(\beta), \alpha(z) \mapsto \beta(z)$ is not a graph isomorphism from $G[\mathrm{im}(\alpha)]$ to $H[\mathrm{im}(\beta)]$.*
- *(S) wins if (D) cannot choose a bijection f, i.e., if $|G| \neq |H|$.*
- *(S) wins after the current round if the configuration (α, β) does not induce a partial isomorphism. Otherwise, the game continues and (D) wins the game if (S) never wins a round.*

For $r \in \mathbb{N}_+$ we define the game variant $\mathrm{BP}^r_{(k_1,k_2)}$, which has the additional winning condition that (D) wins the game if (S) does not win after r rounds.

We now turn to devise an algorithmic counterpart of the logic $\mathsf{C}^{(k_1,k_2)}$ and the game $\mathrm{BP}_{(k_1,k_2)}$. It is an adaptation of the oblivious Weisfeiler-Leman algorithm k-OWL.

Indeed, to capture $\mathsf{C}^{(k_1,k_2)}$-equivalence, we iteratively color (partial) $(k_1 + k_2)$-tuples of vertices of a given graph with the previous color, and a sequence of multisets corresponding to variables as in k_1-OWL. We deviate from the classical oblivious Weisfeiler-Leman algorithm by treating some entries of the tuple as *non-reusable*: For $y \in [y_{k_2}]$ with $\alpha(y) \neq \bot$, the variable y is already assigned in the logic, respectively the non-reusable pebble is already placed in the game. Thus, the entry in α corresponding to this variable should not be replaced by other vertices, but be kept fixed. For this reason we utilize the advantage of oblivious Weisfeiler-Leman that each multiset corresponds to exactly one variable and pebble pair respectively.

Recall that we can view a variable assignment $\alpha\colon [x_{k_1}, y_{k_2}] \rightharpoonup V(G)$ for a graph G as a $[x_{k_1}, y_{k_2}]$-indexed (k_1+k_2)-tuple over $V(G) \dot\cup \{\bot\}$, which we denote by $\overline{\alpha}$. For a variable assignment α, we set $J(\alpha) \coloneqq \{j \in [k_2] : \overline{\alpha}(y_j) = \bot\}$.

▶ **Definition 5.** *Let G be a graph and $k_1, k_2 \in \mathbb{N}$ with $k_1 + k_2 \geq 1$. The* (k_1, k_2)-dimensional oblivious Weisfeiler-Leman algorithm *(short (k_1, k_2)-OWL) iteratively computes a coloring of $[x_{k_1}, y_{k_2}]$-indexed $(k_1 + k_2)$-tuples over $V(G) \dot\cup \{\bot\}$.*

Initially, each tuple $\overline{\alpha}$ is colored by its atomic type in G: $\mathrm{owl}^{(0)}_{(k_1,k_2)}(G, \overline{\alpha}) \coloneqq \mathrm{atp}_{k_1+k_2}(G, \overline{\alpha})$. *This coloring is then refined recursively: for every $r \in \mathbb{N}$, we define*

$$\mathrm{owl}^{(r+1)}_{(k_1,k_2)}(G, \overline{\alpha}) \coloneqq (\mathrm{owl}^{(r)}_{(k_1,k_2)}(G, \overline{\alpha}), \{\!\{\mathrm{owl}^{(r)}_{(k_1,k_2)}(G, \overline{\alpha}[x_i/w]) : w \in V(G)\}\!\}_{i\in[k_1]}, \{\!\{\mathrm{owl}^{(r)}_{(k_1,k_2)}(G, \overline{\alpha}[y_j/w]) : w \in V(G)\}\!\}_{j\in J(\alpha)})$$

Just as in the classical case, the coloring $\mathrm{owl}^{(r+1)}_{(k_1,k_2)}(G)$ refines $\mathrm{owl}^{(r)}_{(k_1,k_2)}(G)$ and eventually stabilizes. We denote the stable coloring by $\mathrm{owl}^{(\infty)}_{(k_1,k_2)}(G)$.

The correspondence of counting logic, pebble game, and algorithm for restricted reusability now is as follows:

▶ **Theorem 6.** *Let G, H be colored graphs and $k_1, k_2 \in \mathbb{N}$ with $k_1 + k_2 \geq 1$. Then for all (k_1, k_2)-configurations (α, β) and $r \in \mathbb{N}$ the following are equivalent:*

1. *For every $\varphi \in \mathsf{C}^{(k_1,k_2)}_r$ with $\mathrm{free}(\varphi) \subseteq \mathrm{dom}(\alpha)$ and $\mathrm{free}(\varphi) \cap [y_{k_2}] = \mathrm{dom}(\alpha) \cap [y_{k_2}]$ it holds that $G, \alpha \models \varphi \Leftrightarrow H, \beta \models \varphi$.*
2. *(D) has a winning strategy for $\mathrm{BP}^r_{(k_1,k_2)}(G, H)$ with initial configuration (α, β).*
3. *It holds that $\mathrm{owl}^{(r)}_{(k_1,k_2)}(G, \overline{\alpha}) = \mathrm{owl}^{(r)}_{(k_1,k_2)}(H, \overline{\beta})$.*

The theorem can be proved by carefully adapting the proof of [3, Theorem 5.2] by treating non-requantifiable variables separately.

4 The Role of Reusability

We investigate the interplay of requantifiable and non-requantifiable variables in $\mathsf{C}^{(k_1,k_2)}$ using the game-theoretic characterization provided by Theorem 6. To this end, we utilize the CFI construction from [3] in the variant employed in [13]. The construction starts from a so-called *base graph*, that is, a connected and colored graph such that every vertex receives a unique natural number as color. By our convention, the coloring induces a linear ordering on the vertices of the base graph. The vertices and edges of the base graph are called *base vertices* and *base edges*, respectively. From a base graph G the *CFI graph* $X(G)$ is constructed by replacing each base vertex in G by a *gadget* consisting of *gadget vertices* but not edges. Gadgets corresponding to adjacent base vertices are then connected by adding edges between gadget vertices. To *twist* a base edge $\{u, v\} \in E$ in $X(G)$ means to replace every edge between the corresponding gadgets by a non-edge and every non-edge by an edge. The *twisted CFI graph* $\widetilde{X}(G)$ is obtained by twisting an arbitrary base edge $e \in E(G)$ in $X(G)$.

We introduce a variant of the cops-and-robber game used in [19] to simulate the game $\mathrm{BP}_{(k_1,k_2)}$ on CFI graphs via a game played only on the base graph. Our variant involves non-reusable cops as a way of restricting reusability of resources.

▶ **Definition 7.** *The cops-and-robber game* $\mathrm{CR}_{(k_1,k_2)}(G)$ *is played on a base graph* G *between a group of* k_1+k_2 *cops and one robber. The cops are denoted by the elements of* $[x_{k_1}, y_{k_2}]$ *and a cop* x_i *is called reusable while a cop* y_j *is called non-reusable. Each round of the game is associated with a partial function* $\gamma\colon [x_{k_1}, y_{k_2}] \rightharpoonup V(G)$ *and an edge* $e \in E(G)$*. The function* γ *encodes the current positions of the cops while the edge* e *is the position of the robber. Initially, there are no cops on the vertices and the robber is placed on some edge of the base graph. One round of the game with current position* (γ, e) *consists of the following steps:*

1. *The cops choose* $z \in [x_{k_1}, y_{k_2}]$ *such that* $z \in [x_{k_1}]$ *or* $\gamma(z)$ *is undefined. If no such* z *exists, the winning condition is checked directly. Then a destination* $w \in V(G)$ *for* z *is declared.*
2. *The robber chooses an edge* e' *in the connected component of* e *in* $G - \mathrm{im}(\gamma[z/\bot])$*.*
3. *The cop* z *is placed on the vertex* w*.*
4. *The new position of the game is given by* $(\gamma[z/w], e')$*.*

The winning condition is as follows:

- *The cops win the game if at the end of the current round both vertices incident to the robber edge* e' *hold cops. The robber wins if the cops never win.*

We also introduce the game $\mathrm{CR}^r_{(k_1,k_2)}(G)$ *with the additional winning condition that the robber wins if the cops do not win in* r *rounds.*

Intuitively, in the game $\mathrm{BP}_{(k_1,k_2)}(X(G), \widetilde{X}(G))$ Spoiler has to *catch* the twist in $\widetilde{X}(G)$ with pebbles to show the difference of the graphs. This corresponds to moving the cops (according to the reusability of the used pebbles) in $\mathrm{CR}_{(k_1,k_2)}(G)$. Duplicator, however, moves the twist in $\widetilde{X}(G)$ using automorphisms of the graph to hide the difference, which corresponds to moving the robber in $\mathrm{CR}_{(k_1,k_2)}(G)$. Following similar arguments from [7, 13], this yields the following lemma, stating that the bijective pebble game on CFI graphs can be simulated appropriately.

▶ **Lemma 8.** *Let* $k_1 + k_2 \geq 2$ *and* $r \in \mathbb{N}$*. Then the robber has a winning strategy in* $\mathrm{CR}^r_{(k_1,k_2)}(G)$ *if and only if (D) has a winning strategy in* $\mathrm{BP}^r_{(k_1,k_2)}(X(G), \widetilde{X}(G))$*.*

We now prove a strict hierarchy for the logics $\mathsf{C}^{(k_1,k_2)}$ by providing, for every pair of logics we want to separate, two CFI graphs $X(G), \widetilde{X}(G)$ that are distinguished by one of the logics, but not the other. To show this, it now suffices to provide strategies for the game $\mathrm{CR}_{(k_1,k_2)}(G)$ by Lemma 8 and Theorem 6. The idea for the choice of the base graphs G is inspired by [13] where grid graphs were chosen as base graphs.

▶ **Definition 9.** *The graph*

$$G_{h\times\ell} \coloneqq (\{v_{i,j} : i \in [h], j \in [\ell]\}, \{\{v_{i,j}, v_{r,s}\} : |i-r| + |j-s| = 1\})$$

is called the grid graph with h *rows and* ℓ *columns. We also call* h *the* height *and* ℓ *the* length *of the grid. We say that the cops* build a barrier *in the game* $\mathrm{CR}_{(k_1,k_2)}$ *played on a graph* G *containing a grid if they are placed on a separator of the graph* G *disconnecting the first column from the last column of the grid.*

First, we show the advantages of reusability: When the cops-and-robber game is played on the grid graph $G_{h\times\ell}$ with at least $h+1$ cops, the cops can be placed on a column to form a barrier and move it through the grid maintaining this formation by using the additional cop. To show the separation, we choose the base graph as a grid of sufficient length such that the cops are required to move or build a barrier repeatedly. This makes reusability necessary as all non-reusable cops are placed at some point and cannot be used to move a barrier any further.

▶ **Lemma 10.** *For all $k_1, k_2, k_1', k_2' \geq 0$, if $k_1 > \max(k_1', 1)$, then $\mathsf{C}^{(k_1,k_2)} \not\preceq \mathsf{C}^{(k_1',k_2')}$.*

Proof. First, consider the game $\mathrm{CR}_{(k_1,k_2)}(G_{(k_1-1)\times(k_1 2^{2k_2'}+1)})$. The cops can build a barrier in the middle of the grid and move it towards the robber only using reusable cops. This is a winning strategy for the cops since the size of the component containing the robber is decreased by a constant in each round and eventually vanishes. On the other hand, we show that the robber has a winning strategy in the game $\mathrm{CR}_{(k_1',k_2')}(G_{(k_1-1)\times(k_1 2^{2k_2'}+1)})$ by induction on k_2'. The base case for $k_2' = 0$ is the game $\mathrm{CR}_{(k_1',0)}(G_{(k_1-1)\times(k_1+1)})$, for which the robber has a winning strategy as a barrier can be built, but not moved. For the inductive step assume $k_2' > 0$ and consider the game $\mathrm{CR}_{(k_1',k_2'+1)}(G_{(k_1-1)\times(k_1 2^{2k_2'+2}+1)})$. Using only reusable cops, the cops can reduce the size of the robber component with a barrier. However, the size remains at least $(k_1 - 1) \cdot (k_1 2^{2k_2'+1} + 1)$, since the robber can choose the larger induced component. When a reusable cop is reused before a non-reusable cop was used to build another barrier, the reusable barrier breaks down and the robber as an escape strategy. Using non-reusable cops, the cops can build another wall to reduce the size of the robber component to $(k_1 - 1) \cdot (k_1 2^{2k_2'} + 1)$. Again, the robber chooses the larger induced component. But now the remaining game is $\mathrm{CR}_{(k_1',k_2'-k_1+2)}(G_{(k_1-1)\times(k_1 2^{2k_2'}+1)})$ and we have $k_2' \geq k_2' - k_1 + 2$. Thus, by the inductive hypothesis the robber has a winning strategy for the remaining game. ◀

Second, we show the advantages of mere capacity: When the base graph is chosen as a complete graph, the robber can choose any edge independently of the choice of vertices by the cops and the only possibility to win for the cops is to have sufficient capacity. In this case, capacity is more valuable than reusability.

▶ **Lemma 11.** *For all $k_1, k_2, k_1', k_2' \geq 0$, if $k_1 + k_2 > k_1' + k_2'$, then $\mathsf{C}^{(k_1,k_2)} \not\preceq \mathsf{C}^{(k_1',k_2')}$.*

Proof. In the game $\mathrm{CR}_{(k_1,k_2)}(K_{k_1+k_2})$, the cops have a winning strategy just by covering all base vertices. In contrast, in the game $\mathrm{CR}_{(k_1',k_2')}(K_{k_1+k_2})$ the robber has a winning strategy. Whenever a cop is picked up there is one edge that is not incident to a cop and thus yields a safe escape for the robber. ◀

Third, we treat the special case of a single requantifiable variable: Intuitively, at least two reusable cops are needed to move a barrier for an arbitrarily large distance in a base graph. When only one single reusable cop is available, the distance that can be covered by a barrier of cops is bounded by $2k_1 + 1$ because for every other move a non-reusable cop must be used. The *perfect binary tree of depth d* is the binary tree B^d such that all interior vertices have two children and all leaves have the same depth.

▶ **Lemma 12.** *For all $k_2, k_2' \geq 1$ it holds that $\mathsf{C}^{(1,k_2)} \not\preceq \mathsf{C}^{(0,k_2')}$ if and only if $k_2' \leq 2k_2$.*

Proof. In the game $\mathrm{CR}_{(1,k_2)}(B^{2k_2})$, the cops have a winning strategy by alternately using non-reusable cops and the reusable cop. In the game $\mathrm{CR}_{(0,2k_2)}(B^{2k_2})$ the robber has a winning strategy as the non-reusable cops are exhausted before the robber is caught. This yields $\mathsf{C}^{(1,k_2)} \not\preceq \mathsf{C}^{(0,2k_2)}$. For $k_2' \leq 2k_2$ we get $\mathsf{C}^{(1,k_2)} \not\preceq \mathsf{C}^{(0,k_2')}$ since the robber wins with the same strategy in $\mathrm{CR}_{(0,k_2')}(B^{2k_2})$. For $k_2' > 2k_2$, let (S) have a winning strategy for $\mathrm{BP}_{(1,k_2)}(G, H)$. Then (S) has a winning strategy for $\mathrm{BP}^{2k_2+1}_{(1,k_2)}(G, H)$ since consecutive moves involving the pebble pair x_1 can be replaced by a single move instead. The winning strategy for (S) in $\mathrm{BP}^{2k_2+1}_{(1,k_2)}(G, H)$ directly yields a winning strategy for (S) in $\mathrm{BP}_{(0,k_2')}(G, H)$: For every pebble pair played by (S) in $\mathrm{BP}^{2k_2+1}_{(1,k_2)}(G, H)$, the player (S) can use a new pair in $\mathrm{BP}_{(0,k_2')}(G, H)$. ◀

With the previous lemmas we can determine the relation of the logics $\mathsf{C}^{(k_1,k_2)}$ and $\mathsf{C}^{(k_1',k_2')}$ for any given combination of parameters.

▶ **Theorem 13.** *For all $k_1, k_2 \in \mathbb{N}$ and $k_1', k_2' \in \mathbb{N}$ with $k_1 + k_2, k_1' + k_2' \geq 2$ it holds that $\mathsf{C}^{(k_1,k_2)} \prec \mathsf{C}^{(k_1',k_2')}$ if and only if one of the following assertions holds:*

1. *$k_1 < k_1'$ and $k_1 + k_2 \leq k_1' + k_2'$,*
2. *$k_1 \leq k_1'$ and $k_1 + k_2 < k_1' + k_2'$, or*
3. *$k_1 = 1$, $k_1' = 0$, and $k_2' > 2k_2$.*

Furthermore, it holds that $\mathsf{C}^{(k_1,k_2)} \equiv \mathsf{C}^{(k_1',k_2')}$ if and only if $(k_1, k_2) = (k_1', k_2')$.

This settles the question of how the use of non-requantifiable variables affects the expressive power of the logic. For the investigation of the logic $\mathsf{C}^{(k_1,k_2)}$ for fixed parameters, it is also of interest how the non-requantifiable variables behave in concrete formulas of the logic. This relates closely to asking whether there are normal forms for the logic $\mathsf{C}^{(k_1,k_2)}$ with respect to reusability. We give a precise answer to this question that rules out many such normal forms. The idea is to construct a new family of base graphs that allow the use of non-reusable cops only after all reusable cops have been used a certain number of times.

▶ **Definition 14.** *We construct the graph $\dot{G}_{h\times\ell}$ from the grid graph $G_{h\times\ell}$ by adding one additional vertex b, which we call bridge vertex, and the edges $\{\{v_{i,\lfloor\frac{\ell}{2}\rfloor}, b\} : i \in [h]\} \cup \{\{b, v_{i,\lfloor\frac{\ell}{2}+1\rfloor}\} : i \in [h]\}$ to $G_{h\times\ell}$. Using this modified grid, we define the following base graph:*

For $\ell \geq 2, h \geq 1$ and $d \geq 1$ we obtain the graph $B^d_{h\times\ell}$ by replacing every vertex of a perfect binary tree B^d of depth d by a grid $\dot{G}_{h\times\ell}$ and connect adjacent grids row-wise (see Figure 1).

▶ **Theorem 15.** *For all $k_1, k_2 \geq 1$ and $r \geq 1$ there exist graphs G and H such that (S) has a winning strategy for $\mathrm{BP}_{(k_1,k_2)}(G, H)$ and in every winning strategy (S) must reuse every reusable pebble pair at least r times (once if $k_1 = 1$) before using a new non-reusable pebble pair.*

Proof. For $k_1 > 1$, we consider the game $\mathrm{CR}_{(k_1,k_2)}(B^{k_2+1}_{(k_1-1)\times 2r})$, see Figure 1. The cops have the following winning strategy: First, they build a barrier in the root grid (behind the bridge vertex) by occupying one full column using k_1 reusable cops. The barrier can then be moved towards the robber using the additional reusable cop. When the barrier reaches the last column, the two subtrees induced by the children of the root grid are disconnected components with respect to the cops. Thus, the robber has to choose an edge in one of these components to escape to. The cops move the barrier into the corresponding subtree, which essentially results in the game $\mathrm{CR}_{(k_1,k_2)}(B^{k_2}_{(k_1-1)\times 2r})$. In every grid of the tree, the cops encounter a bridge vertex that can be covered by one of the k_2 non-reusable cops. The game continues inductively for $\Omega(rk_2)$ rounds, until the cops use the last remaining non-reusable cop to cover the bridge vertex in a leaf grid. The barrier can be moved to the end of the grid and the robber will be caught. Now assume a new non-reusable cop y has been used before all reusable cops have been (re)used r times at some point of the game. Then the barrier was not moved out of the current grid at this point, since all reusable cops have to be moved at least r times to achieve this. Hence, the robber has not chosen a new subtree so far and can pick an edge in a subtree that does not contain y. The cops need to move the barrier into that subtree and use non-reusable cops for the bridge vertices. Since y was used in another subtree, at some point there will be no non-reusable cops left to cover a bridge vertex and the barrier cannot be moved further without breaking down. Thus, the robber can escape indefinitely. For $k_1 = 1$ we consider the game $\mathrm{CR}_{(1,k_2)}(B^{2k_2})$. The cops have the following winning strategy: First, the reusable cop x_1 is placed in the root node. This disconnects

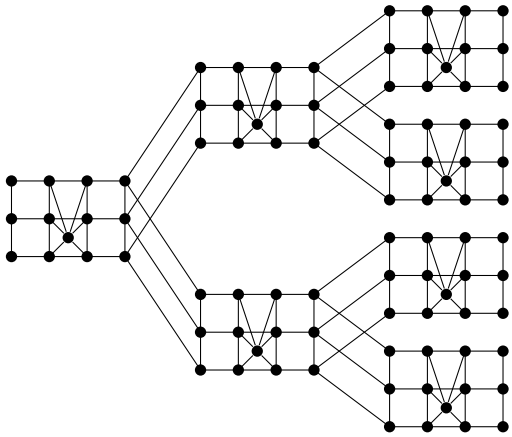


Figure 1 A drawing of the base graph $B^3_{3\times 4}$ for Theorem 15.

the subtrees induced by the two children of the root node for the robber, and the robber has to choose an edge in one of the subtrees of depth $2k_2 - 1$. Accordingly, a non-reusable cop y_1 is placed on the node inducing that subtree, which again disconnects two subtrees of depth $2k_2 - 2$. The cop x_1 can be picked up again from the root node to be placed on the corresponding subtree. Inductively, the cops alternately use non-reusable cops y_j and the reusable cop x_1 to cover the next child node. After $2k_2$ moves, the induced subtree is of depth 0 and the robber is caught in the edge to a leaf node. If two non-reusable cops are used consecutively, similar to the case $k_1 > 1$, there are no non-reusable cops left at depth $2k_2 - 1$ and the remaining reusable cop does not suffice to catch the robber. ◀

Again using Theorem 6 this result translates into the language of logic as follows:

▶ **Corollary 16.** *For all $k_1, k_2 \geq 1$ and $r \geq 1$ there exist graphs G, H such that G and H are not $\mathsf{C}^{(k_1,k_2)}$-equivalent and for every formula $\varphi \in \mathsf{C}^{(k_1,k_2)}$ that distinguishes G and H the following holds: There exists a sequence of subformulas $\exists^{\geq n_1} y_{j_1}\psi_1, \ldots, \exists^{\geq n_{k_2}} y_{j_{k_2}}\psi_{k_2}$ of φ such that $\mathrm{qr}(\psi_{k_2}) = k_1$ and for $\ell \in [k_2 - 1]$ the formula $\psi_{\ell+1}$ is a subformula of ψ_ℓ with $\mathrm{qr}(\psi_\ell) \geq \mathrm{qr}(\psi_{\ell+1}) + k_1 r$ if $k_1 > 1$ and $\mathrm{qr}(\psi_\ell) \geq \mathrm{qr}(\psi_{\ell+1}) + 1$ if $k_1 = 1$. Moreover, between the quantifications $\exists^{\geq n_\ell} y_{j_\ell}\psi_\ell$ and $\exists^{\geq n_{\ell+1}} y_{j_{\ell+1}}\psi_{\ell+1}$ all requantifiable variables have to be requantified r times (once if $k_1 = 1$) in ψ_ℓ.*

The necessity of this pattern of (re)quantification rules out various normal forms with respect to requantification for $\mathsf{C}^{(k_1,k_2)}$ one might have hoped to have. In particular, it is not sufficient to quantify all non-requantifiable variables directly one after the other.

Regarding classical logics without restricted requantification, Theorem 13 and Corollary 16 yield that for $k_1, k_2 \geq 1$ the power of $\mathsf{C}^{(k_1,k_2)}$ to distinguish graphs is not identical to that of C^k, C_r, or C^k_r for all $k, r \in \mathbb{N}$.

5 Space Complexity

In this section we investigate the space complexity of deciding whether two given graphs are $\mathsf{C}^{(k_1,k_2)}$-equivalent. In principle, this can be achieved by testing $\mathrm{owl}^{(\infty)}_{(k_1,k_2)}(G) = \mathrm{owl}^{(\infty)}_{(k_1,k_2)}(H)$ by Theorem 6. However, a naive implementation of (k_1, k_2)-OWL requires space $\Omega(n^{k_1+k_2})$ and hence provides no improvement compared to the situation with unrestricted reusability. We seek to improve the space complexity to $O\big(n^{k_1} \log n\big)$ when both requantifiable and non-requantifiable variables are involved. Here the O notation hides factors depending on k_1 and k_2 but not on n.

To achieve this, we observe that the color $\mathrm{owl}^{(r)}_{(k_1,k_2)}(G,\overline{\alpha})$ only depends on the colors $\mathrm{owl}^{(s)}_{(k_1,k_2)}(G,\overline{\beta})$ with $s<r$ where $\beta|_{[y_{k_2}]}$ is an extension of $\alpha|_{[y_{k_2}]}$. This allows us to compute these colorings, while only ever remembering colors of assignments with a few distinct $[y_{k_2}]$-parts. Moreover, we show that the (k_1,k_2)-dimensional oblivious Weisfeiler-Leman algorithm can equivalently be implemented by alternatingly refining with respect to the reusable dimensions until the coloring stabilizes, and refining with respect to the first unassigned non-requantifiable variable. We use this to show that the iteration number of (k_1,k_2)-OWL is at most $(k_2+1)n^{k_1}-1$.

▶ **Definition 17.** *For colorings χ and χ' on a set S, we say that χ* refines *χ', written $\chi \preceq \chi'$, if every χ'-color class is a union of χ-color classes. If $\chi \preceq \chi'$ and $\chi \succeq \chi'$, we write $\chi \equiv \chi'$.*

In the context of colorings, it is natural to understand the oblivious Weisfeiler-Leman algorithm as a *refinement operator*, i.e., a function that maps every coloring to a refined coloring. To make this formal, we define for every coloring χ on assignments $\alpha\colon [x_{k_1},y_{k_2}] \rightharpoonup V(G)$ the OWL-refinement

$$\mathrm{owl\text{-}ref}_{(k_1,k_2)}(\chi)(\alpha) \coloneqq \Big(\chi(\alpha), \{\!\{\chi(\alpha[x_i/w]) : w\in V(G)\}\!\}_{i\in[k_1]}, \{\!\{\chi(\alpha[y_j/w]) : w\in V(G)\}\!\}_{j\in J(\alpha)}\Big).$$

This refinement is precisely the refinement that is applied by OWL in each iteration. In particular, applying it r times to the initial coloring by atomic types yields precisely the r-round OWL-coloring. That is,

$$\mathrm{owl\text{-}ref}^{(r)}_{(k_1,k_2)}(\mathrm{atp}_{k_1+k_2}(G)) = \mathrm{owl}^{(r)}_{(k_1,k_2)}(G).$$

Note that OWL-refinement is *monotone* in the sense that for all colorings χ and χ' with the property $\chi\preceq\chi'$ it also holds that $\mathrm{owl\text{-}ref}_{(k_1,k_2)}(\chi)\preceq\mathrm{owl\text{-}ref}_{(k_1,k_2)}(\chi')$.

In order to space-efficiently deal with these refinements, we want to separate the refinements with respect to reusable dimensions from those with respect to non-reusable dimensions. To do this, note that the definition of $\mathrm{owl\text{-}ref}_{(k_1,k_2)}$ still makes sense for colorings assignments $\alpha\colon [x_{k_1'},y_{k_2'}] \rightharpoonup V(G)$ for $k_1'\geq k_1$ or $k_2'\geq k_2$, where the refinement just refines with respect to some but not all of the dimensions.

Moreover, in order to handle the distinguishing power of (k_1,k_2)-OWL on two different graphs, we note that we can also simultaneously apply these refinement operators to colorings on assignments over two different graphs.

With this terminology at hand, we can clarify the intuition we may gain from Section 4 regarding the employment of non-reusability. That is, in order to distinguish graphs, it suffices to alternatingly refine with respect to all requantifiable variables and a non-requantifiable variable.

▶ **Lemma 18.** *For all $k_1+k_2\geq 1$, we have*

$$\mathrm{owl}^{(\infty)}_{(k_1,k_2)}(G) \equiv (\mathrm{owl\text{-}ref}^{(\infty)}_{(k_1,0)} \circ \mathrm{owl\text{-}ref}_{(0,k_2)})^{(k_2)}(\mathrm{owl}^{(\infty)}_{(k_1,0)}(G))$$

Proof. Because $\mathrm{owl}_{(k_1,k_2)}$ is a finer refinement than $\mathrm{owl\text{-}ref}_{(k_1,0)}$ and than $\mathrm{owl\text{-}ref}_{(0,k_2)}$, the direction $\preceq$ is immediate. For the other direction, set

$$\chi_r \coloneqq \left(\mathrm{owl\text{-}ref}^{(\infty)}_{(k_1,0)} \circ \mathrm{owl\text{-}ref}_{(0,k_2)}\right)^{(r)} \left(\mathrm{owl}^{(\infty)}_{(k_1,0)}(G)\right).$$

We use the bijective pebble game and show, by induction on r, that for every (k_1, k_2)-configuration (α, β) over G with $|\mathrm{dom}(\alpha) \cap [y_{k_2}]| = k_2 - r$ such that α and β have equal χ_r-colors, (D) has a winning strategy in the game $\mathrm{BP}_{(k_1,k_2)}(G, G)$ with initial position (α, β). This then implies the equality of owl-ref$^{(\infty)}_{(k_1,k_2)}(\chi)$-colors.

For $r = 0$, the colors are precisely the colors computed by (k_1, k_2)-OWL. Thus, (D) has a winning strategy by Theorem 6.

Now, assume the claim is true for some r and let (α, β) be a (k_1, k_2)-configuration with $|\mathrm{dom}(\alpha) \cap [y_{k_2}]| = k_2 - (r+1)$ such that α and β have equal χ_{r+1}-colors. By the construction of $(k_1, 0)$-OWL refinement, (D) can preserve the equality of χ_{r+1}-colors as long as (S) picks up reusable pebble pairs. When (S) picks up a non-reusable pebble pair, (D) can play such that the resulting positions have the same χ_r-colors. But then, (D) has a winning strategy by the induction hypothesis. ◀

Because classical owl$_{(k_1,0)}$-refinement stabilizes after at most $n^{k_1} - 1$ rounds, this scheme yields an upper bound on the iteration number of the oblivious Weisfeiler-Leman algorithm.

▶ **Corollary 19.** *The sequence of colorings* owl$^{(r)}_{(k_1,k_2)}$ *computed on a graph G stabilizes after at most $(k_2 + 1)n^{k_1} - 1$ rounds.*

We will now turn to the computation of the OWL-colorings. Because the names of the OWL-colors consist of nested multisets, they can become exponentially long. The usual way to deal with this is to either replace after each iteration round all color names by numbers of logarithmic length, or to not compute the colors at all but only consider the order on the variable assignments induced by the lexicographic ordering of their OWL-colors. We will switch between these two viewpoints depending on suitability to the task at hand. Accordingly, we use two different encodings of the colorings we deal with. Consider a coloring $\chi\colon M \to C$ and an order $\leq$ on the set of colors C. We say that an algorithm is given *oracle access to the ordering of χ-colors* if the algorithm has access to a function that, given two elements $m, m' \in M$, returns whether $\chi(m) \leq \chi(m')$. For the second way that our algorithms interact with colorings, we call a coloring $\chi'\colon M \to [|M|]$ a *normalization of* χ if for all $m, m' \in M$ we have $\chi(m) \leq \chi(m')$ if and only if $\chi'(m) \leq \chi'(m')$. Now, we say that an algorithm is given a *function table for* χ if for some normalization χ' of χ the algorithm is given an array A with $A[m] = \chi'(m)$ for all $m \in M$ suitably encoded as numbers in $[|M|]$. Similarly, we say that an algorithm *computes a function table for* χ if it outputs such an array. Note that a function table can be stored in space $|M| \cdot \lceil \log_2 |M| \rceil \in O(|M| \log |M|)$.

The main technical tool needed for the implementation of owl-ref$_{(k_1,k_2)}$-refinements, is the ability to compare multisets of previously computed colors when given oracle access to a function comparing these previous colors.

▶ **Lemma 20.** *Given a natural number n in unary, oracle access to a total order $\preceq$ on $[n]$, and two multisets M and M' on $[n]$ of order at most n, the lexicographic order of M and M' can be decided in logarithmic space using quadratic time. Also, the lexicographical order of tuples of colors can be computed in logarithmic space.*

Proof. Consider two multisets $M = \{\!\{s_1, \ldots, s_n\}\!\}$ and $M' = \{\!\{s'_1, \ldots, s'_n\}\!\}$. Note that we have enough space to store a constant number of elements of $[n]$.

For a number $i \in [n]$, we denote the number of occurrences of i in M or M' by $M(i)$ and $M'(i)$ respectively. Note that these numbers can be computed in logarithmic space and linear time by simply comparing i to all elements in either set.

We start by finding the minimal element m_1 of M and m'_1 of M'. If $m_1 \neq m'_1$, we return their order. Otherwise, if $M(m_1) \neq M'(m_1)$, we return this order.

Thus, assume $m_1 = m_1'$ and that they occur in both multisets the same number of times. Next, we find the second-smallest elements m_2 and m_2' of both sets, and can now forget about m_1 and m_1'. We again compare m_2 and m_2' and their number of occurrences and possibly return the order accordingly. Iteratively, we only need to remember the i-th smallest elements to find the $(i+1)$-th smallest elements, and we iteratively compare these elements and their number of occurrences. ◀

This allows us to compute the order of owl-ref$_{(k_1,k_2)}(\chi)$-colors in logarithmic space when we are given oracle access to the order of χ-colors. Using the bound on the iteration number of (k_1, k_2)-OWL from Corollary 19, and the fact that we can perform one iteration using only logarithmic additional space, we immediately obtain an algorithm that can compare $\mathrm{owl}^{(\infty)}_{(k_1,k_2)}$-colors using space at most $O(n^{k_1} \log n)$, where we again dropped multiplicative factors depending on k_1 and k_2. However, this naive implementation will not run in polynomial time. Indeed, because there are already $n^{k_1+k_2}$ many variable assignments, we do not have enough space to store even the (order of) colors computed in the previous round. Instead, this naive algorithm recomputes polynomially many previous colors in every step, which leads to a polynomially branching algorithm with exponential running time.

To remedy this, we make full use of the scheme from Lemma 18. While performing $\mathrm{owl}_{(k_1,0)}$-refinements, we are able to store a function table with the previously computed colors for all assignments with the same $[y_{k_2}]$-part. This allows us to perform a full $\mathrm{owl}^{(\infty)}_{(k_1,0)}$-refinement in polynomial time and the required space. Only when performing one of the k_2 many $\mathrm{owl}_{(0,k_2)}$-refinement steps do we need to consider variable assignments with different $[y_{k_2}]$-parts. In this latter case, we cannot circumvent needing to compute colors for these assignments polynomially many times. While this does again lead to a polynomially branching algorithm, the depth of this branching is bounded by k_2, which leads to a polynomial running time increase of n^{k_2}.

For a fixed assignment $\eta\colon [y_{k_2}] \rightharpoonup V(G)$, we denote by $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_\eta$ the set of assignments $\alpha\colon [x_{k_1}, y_{k_2}] \rightharpoonup V(G)$ whose $[y_{k_2}]$-part is η. Because we only ever need to compare the colors of two assignments at a time, it will always be sufficient to compute the OWL-coloring on sets of the form

$$[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}$$

for a $(0, k_2)$-configuration (η_G, η_H) over G and H. When restricting ourselves to assignments in such a set, the coloring computed by $(k_1, 0)$-OWL can be computed as usual:

▶ **Lemma 21.** *Let $k_1 + k_2 \geq 1$, G, H be graphs and (η_G, η_H) be a $(0, k_2)$-configuration over G, H. Given a function table for a coloring χ on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}$, a function table for* owl-ref$_{(k_1,0)}(\chi)$ *can be computed in time $n^{O(k_1)}$ and space $O(k_1 n^{k_1} \log n)$.*

Proof. Note that $\left|[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}\right| = 2(n+1)^{k_1}$, which means that we can store a function tables for χ and owl-ref$_{(k_1,0)}(\chi)$ in space

$$O\left(2(n+1)^{k_1} \cdot \log\left(2(n+1)^{k_1}\right)\right) = O\left(k_1 n^{k_1} \log n\right).$$

In addition to these two function tables, we will only need logarithmic space.

In order to compute the function table for owl-ref$_{(k_1,0)}(\chi)$, we need to refine the coloring χ with respect to the multisets

$$\{\!\{\chi(\alpha[x_i/w]) : w \in V(G)\}\!\} \quad \text{or} \quad \{\!\{\chi(\alpha[x_i/w]) : w \in V(H)\}\!\}$$

for all $i \in [k_1]$. By using the function table for χ as an oracle, we can compare these multisets in logarithmic additional space using Lemma 20.

This allows us to compare owl-ref$_{(k_1,0)}(\chi)$-colors in the required space. Now, we simply start to compare each variable assignment α with all other assignments and count the number of assignments whose color is less than or equal to α. Then, we use this count as the new color of α and insert it into our function table. ◀

By applying Lemma 21 repeatedly until the coloring stabilizes, and only ever storing the function table from the previous and current iteration round we get the following:

▶ **Corollary 22.** *Let $k_1 + k_2 \geq 1$, G and H be graphs, and (η_G, η_H) be a $(0, k_2)$-configuration over G and H. Given a function table for a coloring χ on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}$, a function table for* owl-ref$^{(\infty)}_{(k_1,0)}(\chi)$ *can be computed in time $n^{O(k_1)}$ space $O(k_1 n^{k_1} \log n)$.*

Now, we turn to refinements with respect to non-requantifiable variables.

▶ **Lemma 23.** *Let $k_1 + k_2 \geq 1$, G and H be graphs, and χ a coloring on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)] \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]$.*

Given oracle access to the order of χ-colors, we can compute for every $(0, k_2)$-configuration (η_G, η_H) the function table of owl-ref$_{(0,k_2)}(\chi)$ *on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}$ using space $O(k_1 n^{k_1} \log n + k_2 \log n)$ and time $n^{O(k_1)}$.*

Proof. Note that we have enough space to hold the function table. In addition, we will only need logarithmic space.

Using Lemma 20, we can compute the lexicographic ordering of owl-ref$_{(0,k_2)}(\chi)$-colors with logarithmic additional space. We can then compute the function table by assigning to each assignment α as the new color the number of assignments β in $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_G} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(H)]_{\eta_H}$ such that

$$\text{owl-ref}_{(0,k_2)}(\chi)(\beta) \leq_{\text{lex}} \text{owl-ref}_{(0,k_2)}(\chi)(\alpha),$$

which is a number in $[2(n+1)^{k_1}]$. ◀

Together, these two statements allow us to compare the colors computed by the (k_1, k_2)-dimensional oblivious Weisfeiler-Leman algorithm in a time- and space-efficient manner.

▶ **Theorem 24.** *Let $k_1 + k_2 \geq 1$ be fixed. For all (k_1, k_2)-configurations (α, β) over graphs G and H, we can decide whether $G, \alpha_1 \equiv_{\mathsf{C}^{(k_1,k_2)}} H, \alpha_2$ using space $O\big(k_1(k_2+1)n^{k_1} \log n + (k_2)^2 \log n\big)$ and polynomial time.*

Proof. By Lemma 18, we have

$$\text{owl}^{(\infty)}_{(k_1,k_2)}(G) \equiv \left(\text{owl-ref}^{(\infty)}_{(k_1,0)} \circ \text{owl-ref}_{(0,k_2)}\right)^{(k_2)} \left(\text{owl}^{(\infty)}_{(k_1,0)}(G)\right).$$

and similarly for H. We show by induction on r that we can compute for every pair of graphs $G_1, G_2 \in \{G, H\}$ and every $(0, k_2)$-configuration (η_1, η_2) over G_1 and G_2, a function table of

$$\chi_r \coloneqq \left(\text{owl-ref}^{(\infty)}_{(k_1,0)} \circ \text{owl-ref}_{(0,k_2)}\right)^{(r)} \left(\text{owl}^{(\infty)}_{(k_1,0)}\right)$$

on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G_1)]|_{\eta_1} \dot\cup [[x_{k_1}, y_{k_2}] \rightharpoonup V(G_2)]|_{\eta_2}$ using time $n^{O((k_1+1)(r+1))}$ and space $O\big(k_1(r+1)n^{k_1} \log n + k_2(r+1) \log n\big)$, where $\text{owl}^{(\infty)}_{(k_1,0)}(G_1, G_2)$ is the common coloring computed by (k_1, k_2)-OWL on both G_1 and G_2.

If $r = 0$, we only need to compute the classical OWL-coloring. To do this, we first note that we can compute a function table listing the atomic types of assignments, each encoded as numbers in $[2(n+1)^{k_1}]$. Then, the claim follows from Corollary 22.

For the induction step, assume we can compute function tables for (restrictions of) the coloring χ_r for every fixed $(0, k_2)$-configuration (η_1, η_2) over G_1 and G_2 in the required time and space. This in particular implies that we can compute the order of χ_r-colors of arbitrary assignments in time $n^{O((k_1+1)(r+1))}$ and space $O\big((r+1)(k_1 n^{k_1} \log n + k_2 \log n)\big)$.

For every fixed $(0, k_2)$-configuration (η_1, η_2), we can thus compute a function table for the refined coloring $\text{owl-ref}_{(0,k_2)}(\chi_r)$ on $[[x_{k_1}, y_{k_2}] \rightharpoonup V(G)]_{\eta_1,\eta_2}$ using space $O\big((r+2)(k_1 n^{k_1} \log n + k_2 \log n)\big)$ by Lemma 23. Because the algorithm from Lemma 23 runs in time $n^{O(k_1)}$, it can make at most $n^{O(k_1)}$ comparisons of previously computed colors, which means that this step takes time at most $n^{O(k_1)} \cdot n^{O((k_1+1)r)} = n^{O((k_1+1)(r+2))}$.

Using Corollary 22, we can refine this to a function table of

$$\chi_{r+1} = \text{owl-ref}^{(\infty)}_{(k_1,0)} \circ \text{owl-ref}_{(0,k_2)}(\chi_r)$$

For $r = k_2$, this yields the claim. ◀

6 Graphs Identified by Logics with Restricted Requantification

We finally give two classes of graphs where very few reusable variables already suffice for identification. We say that a logic *identifies* a graph G if it distinguishes G from every non-isomorphic graph.

First, we show that the logic $\mathsf{C}^{(0,d+1)}$ identifies all graphs of tree-depth at most d. Second, we refine previous work in which it was shown that $\mathsf{C}^{(4,0)}$ identifies all planar graphs [28]. By closely inspecting the arguments, we show that already $\mathsf{C}^{(2,2)}$ suffices to identify all 3-connected planar graphs.

Graphs of bounded tree-depth

Tree-depth is a graph parameter that intuitively measures how close a graph is to a star [32]. Let G be a graph and let $G_1, \dots, G_p$ be the connected components of G. Then the *tree-depth* of G is inductively defined as

$$\text{td}(G) \coloneqq \begin{cases} 1 & \text{if } |G| = 1 \\ 1 + \min_{v \in V(G)} \text{td}(G - v) & \text{if } p = 1 \text{ and } |G| > 1 \\ \max_{i \in [p]} \text{td}(G_i) & \text{otherwise.} \end{cases}$$

Note that a graph has tree-depth 1 if and only if it is an independent set, and tree-depth 2 if and only if it is a disjoint union of isolated vertices and stars. Intuitively, graphs of bounded tree-depth are those which, by repeatedly deleting one vertex in each connected component, can be eliminated in a bounded number of rounds. We start by showing how to simulate this deletion of vertices by pebbling them instead.

▶ **Definition 25.** *Let G be a graph with vertex coloring χ_G and let $v \in V(G)$. We define the colored graph $G \wr v$ as the graph $G - v$ with the new coloring $\chi_{G \wr v}$ defined by setting*

$$\chi_{G \wr v}(w) \coloneqq \begin{cases} (\chi_G(w), 1) & \text{if } \{v, w\} \in E(G) \\ (\chi_G(w), 0) & \text{if } \{v, w\} \notin E(G) \end{cases}$$

for all $w \in V(G) \setminus \{v\}$.

▶ **Lemma 26** ([16, Lemma 4.5]). *Let $k_2 \geq 2$, G, H be graphs with $|G| = |H| \geq 2$ and $v \in V(G), w \in V(H)$ with $\chi_G(v) = \chi_H(w)$. Then the following are equivalent:*

1. *(D) has a winning strategy for $\mathrm{BP}_{(0,k_2)}(G, H)$ with initial position given by $\alpha(y_1) = v, \beta(y_1) = w$ and $\mathrm{dom}(\alpha) = \mathrm{dom}(\beta) = \{y_1\}$.*
2. *(D) has a winning strategy for $\mathrm{BP}_{(0,k_2-1)}(G \wr v, H \wr w)$.*

We can now prove our result on identification of graphs of bounded tree-depth:

▶ **Theorem 27.** *For all $d \geq 1$, the logic $\mathsf{C}^{(0,d+1)}$ identifies all colored graphs of tree-depth at most d.*

Proof. For two graphs G and C, denote by $\mathrm{noc}(G, C)$ the number of connected components of G that are isomorphic to C. Using the equivalence of the logic and bijective pebble game, it suffices to prove the following claim, which lends itself better to an inductive proof:

▷ Claim 28. Let G and H be colored graphs, and C a connected, colored graph of tree-depth at most k_2. If $\mathrm{noc}(G, C) \neq \mathrm{noc}(H, C)$, then (S) has a winning strategy for the game $\mathrm{BP}_{(0,k_2+1)}(G, H)$.

Proof. We argue by induction on k_2. If $k_2 = 1$, then C is a single vertex. Thus, G and H differ in the number of isolated vertices of some specific color, which allows (S) to win in 2 rounds.

For the induction step, assume that the claim is true for k_2. Now, consider a graph C with $\mathrm{td}(C) \leq k_2 + 1$ and assume w.l.o.g. that $\mathrm{noc}(G, C) > \mathrm{noc}(H, C)$. By the definition of tree-depth, C contains a vertex c such that $\mathrm{td}(C - c) \leq k_2$. Let s be the number of such vertices.

We call a vertex v in either G or H *C-shrinking* if it is contained in a connected component C_v isomorphic to C, and $\mathrm{td}(C_v - v) \leq k_2$. The number of C-shrinking vertices in G and H is $s \cdot \mathrm{noc}(G, C)$ and $s \cdot \mathrm{noc}(H, C)$ respectively. In particular, G contains more C-shrinking vertices than H.

Now, we describe the winning strategy for (S) in the game $\mathrm{BP}_{(0,k_2+2)}(G, H)$. First, (S) picks up the (unused) pebble pair y_1, and (D) picks a bijection $f \colon V(G) \to V(H)$. Then there exist some C-shrinking vertex $v \in V(G)$ such that its image $f(v)$ is not C-shrinking in H. Then (S) places the pebble pair on these two vertices. By Lemma 26, it now suffices to argue that (D) has a winning strategy for the game $\mathrm{BP}_{(0,k_2+1)}(G \wr v, H \wr f(v))$. For this, let C_v be the connected component of v in G, and consider the connected components $C_v^{(1)}, \ldots, C_v^{(\ell)}$ of $C_v \wr v \subseteq G \wr v$. If for some $i \in [\ell]$, we have $\mathrm{noc}(G \wr v, C_v^{(i)}) \neq \mathrm{noc}(H \wr f(v), C_v^{(i)})$, then we are done by the induction hypothesis. Thus, we are left with the case that $\mathrm{noc}(G \wr v, C_v^{(i)}) = \mathrm{noc}(H \wr f(v), C_v^{(i)})$. Note that the vertex-colorings of $G \wr v$ and $H \wr f(v)$ ensure that all connected components isomorphic to $C_v^{(i)}$ for some i are incident to v or $f(v)$ respectively. Thus, all copies of $C_v^{(i)}$ in G lie in C_v.

In this case, there is an isomorphism φ between the subgraphs induced by $G \wr v$ and $H \wr f(v)$ on the union of connected components isomorphic to $C_v^{(i)}$ for some $i \in [\ell]$. The vertex-colorings of $G \wr v$ and $H \wr f(v)$ further ensure that φ can be extended by $\varphi(v) \coloneqq f(v)$, so that its domain is all of C_v. Thus, φ now embeds C_v into the connected component of $f(v)$, which might, however, have additional vertices attached to $f(v)$. If the image of C_v under φ was the whole connected component of $f(v)$, then $f(v)$ would be C-shrinking, which contradicts our assumption. Thus, there are additional vertices attached to $f(v)$. Thus, v and $f(v)$ have distinct degrees, which allows (S) to win in one further round. ◁

The theorem now follows by applying the claim to all connected components of G. ◀

Note that using Theorem 24 on the space complexity of $\mathsf{C}^{(k_1,k_2)}$-equivalence, the theorem in particular reproves the statement that isomorphism of graphs of bounded tree-depth can be decided in logarithmic space [5].

Note moreover that because $\mathsf{C}^{(1,1)}$ can count the number of connected components isomorphic to a fixed colored star, the above inductive proof also shows that for $d \geq 2$, the logic $\mathsf{C}^{(1,d-1)}$ identifies all colored graphs of tree-depth at most d.

3-connected planar graphs

The next class of graphs we consider are 3-connected planar graphs. These naturally appear in the proof that C^4 identifies all planar graphs, which starts by reducing the claim for arbitrary planar graphs to 3-connected planar graphs via the decomposition into triconnected components [28]. We recall their proof that C^4 identifies every 3-connected planar graph and show that it actually already yields that $\mathsf{C}^{(2,2)}$ suffices.

The underlying technical lemma is the following:

▶ **Lemma 29** ([28, Lemma 23]). *Let G be a 3-connected planar graph and let $v_1, v_2, v_3 \in V(G)$. If v_1, v_2, v_3 lie on a common face of G, then $\mathrm{wl}_1^{(\infty)}(G_{(v_1,v_2,v_3)})$ is a discrete coloring.*

In the classical setting, from this lemma one can obtain that 4-WL, i.e., C^5 identifies all 3-connected planar graphs. We make use of our framework and show that instead it suffices to use non-reusable resources to cover the individualized vertices.

▶ **Corollary 30** (compare [28, Corollary 24]). *The logic $\mathsf{C}^{(2,3)}$ identifies all 3-connected planar graphs.*

Proof. Let G be a 3-connected planar graph. By Lemma 29, there are $v_1, v_2, v_3 \in V(G)$ such that $\mathrm{wl}_1^{(\infty)}(G_{(v_1,v_2,v_3)})$ is a discrete coloring. Then by Lemma 1, also $\mathrm{owl}_{(2,0)}^{(\infty)}(G_{(v_1,v_2,v_3)})$ is a discrete coloring, which implies that $G_{(v_1,v_2,v_3)}$ is identified by $\mathsf{C}^{(2,0)}$. Thus, there exists a formula $\varphi(y_1, y_2, y_3) \in \mathsf{C}^{(2,3)}$ which defines the graph G with three individualized vertices represented by y_1, y_2 and y_3 up to isomorphism. But then, the formula $\exists y_1 \exists y_2 \exists y_3 \varphi(y_1, y_2, y_3)$ identifies G. ◀

In order to improve the identification result from C^5 to C^4 and from $\mathsf{C}^{(2,3)}$ to $\mathsf{C}^{(2,2)}$, one observes that for almost all 3-connected graphs, the individualization of just 2 vertices already suffices for the above claim, and the exceptions, where indeed, 3 vertices are necessary are somewhat rare.

▶ **Definition 31.** *A 3-connected planar graph is called an* exception *if there are no two vertices $v_1, v_2 \in V(G)$ such that $\mathrm{wl}_1^{(\infty)}(G_{(v_1,v_2)})$ is discrete.*

The crucial tool in lowering the number of variables needed is an explicit classification of all exceptions [28]. This allows to prove the following:

▶ **Lemma 32.** *All exceptions are identified by $\mathsf{C}^{(2,2)}$.*

Proof. The exceptions are the following:

- all bipyramids, i.e., cycles with two additional non-adjacent but otherwise universal vertices,
- all platonic solids besides the dodecahedron, and the rhombic dodecahedron,
- the triakis tetrahedron, the tetrakis hexahedron and the triakis octahedron, i.e., the graphs obtained from the tetrahedron, the hexahedron and the octahedron by adding one vertex per face, whose neighborhood consists of the vertices on that face.

Because even color refinement identifies every graph with at most 5 vertices, and the bipyramid of order 6 is the octahedron, we start with bipyramids of order at least 7. We individualize two adjacent vertices of the cycle. Then, color refinement computes a discrete coloring on the underlying cycle, while the two pyramid tips get the same color, which is, however, distinct from all other colors. This graph is identified by color refinement.

Next, consider the platonic solids and the rhombic dodecahedron. All of these are distance-regular graphs of diameter at most 4, with at most 14 vertices. Note that for every $d \in \mathbb{N}$, there exists a formula $\varphi_d(y_1, y_2) \in \mathsf{C}^{(2,2)}$ stating that y_1 and y_2 have distance d. This implies that in $\mathsf{C}^{(2,2)}$ we can express that a graph G is distance-regular with a given parameter set. Because every distance-regular graph of order at most 14 is determined by its parameters [38], $\mathsf{C}^{(2,2)}$ identifies all platonic solids and the rhombic dodecahedron.

For the last case, note that the vertices of the platonic solid and the added vertices for every face have distinct degrees. As moreover, adding these vertices does not change the distance between any two original vertices, $\mathsf{C}^{(2,2)}$ can still express that the underlying platonic solid is of the correct type. Additionally, we can express that the neighborhood of each added face vertex is a cycle of the correct length, and that no two face vertices share more than 2 common neighbors. This identifies the last class of exceptions. ◀

This finally allows us to prove identification by $\mathsf{C}^{(2,2)}$:

▶ **Theorem 33.** *Every 3-connected planar graph is identified by* $\mathsf{C}^{(2,2)}$.

Proof. Let G be a 3-connected planar graph. If G is an exception, this follows from Lemma 32. If G is not an exception, the claim can be proven as in Corollary 30, where we instead only need to individualize 2 instead of 3 vertices. ◀

It is unclear how precisely this result generalizes to all planar graphs. Moreover, it is known that all graphs of Euler genus g are identified by C^{4g+4} [18], and we would expect that also here, for sufficiently connected graphs only a very small number of the variables must be requantified.

7 Outlook

In this work, we establish a refined framework for the logical description of graphs by means of the requantification of variables. We indicate some open questions for future work in vastly different directions.

Towards structural graph theory, the newly defined cops-and-robber game $\mathrm{CR}^{(k_1,k_2)}$ defines a two-parametric family of graph classes, which contain the tree-width and tree-depth graphs as subclasses. It will be interesting to obtain graph-theoretic characterizations for these classes and to study them from an algorithmic and logical point of view.

From a practical viewpoint, the space complexity is generally the roadblock to a use of higher-dimensional Weisfeiler-Leman in isomorphism testing and graph neural networks. The introduction of non-requantifiable variables is a technique to limit space complexity, so it needs to be investigated whether problems that arise in practice can be solved by this technique.

In another direction, it will be interesting to investigate other classes of graphs that are known to have bounded Weisfeiler-Leman dimension. Using the new variant with restricted reusability, it may be possible to obtain a more fine-grained complexity measure and more space-efficient algorithms for such graph classes. In particular, it seems highly plausible that (sufficiently connected) bounded genus graphs require only a limited number of requantifiable

variables. This might allow to design easier fixed-parameter tractable results for graph isomorphism, for example on bounded genus graphs (see [18]). However, for this there are further restrictions, as non-reusable variables must be choosable from FPT-size bounded sets.

References

1 Pablo Barceló, Floris Geerts, Juan L. Reutter, and Maksimilian Ryschkov. Graph neural networks with local graph parameters. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 25280–25293, 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/d4d8d1ac7e00e9105775a6b660dd3cbb-Abstract.html.

2 Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M. Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: https://openreview.net/forum?id=dFbKQaRk15w.

3 Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

4 Jinzhuan Cai, Jin Guo, Alexander L. Gavrilyuk, and Ilia Ponomarenko. A large family of strongly regular graphs with small Weisfeiler-Leman dimension. *arXiv:2312.00460 [math.CO]*, 2023. arXiv. doi:10.48550/arXiv.2312.00460.

5 Bireswar Das, Murali Krishna Enduri, and I. Vinod Reddy. Logspace and FPT algorithms for graph isomorphism for subclasses of bounded tree-width graphs. In M. Sohel Rahman and Etsuji Tomita, editors, *WALCOM: Algorithms and Computation - 9th International Workshop, WALCOM 2015, Dhaka, Bangladesh, February 26-28, 2015. Proceedings*, volume 8973 of *Lecture Notes in Computer Science*, pages 329–334. Springer, 2015. doi:10.1007/978-3-319-15612-5_30.

6 Samir Datta, Nutan Limaye, Prajakta Nimbhorkar, Thomas Thierauf, and Fabian Wagner. Planar graph isomorphism is in log-space. *ACM Trans. Comput. Theory*, 14(2):8:1–8:33, 2022. doi:10.1145/3543686.

7 Anuj Dawar and David Richerby. The power of counting logics on restricted classes of finite structures. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic*, pages 84–98, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-74915-8_10.

8 Holger Dell, Martin Grohe, and Gaurav Rattan. Lovász meets Weisfeiler and Leman. 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 40:1–40:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICS.ICALP.2018.40.

9 Zdenek Dvorák. On recognizing graphs by numbers of homomorphisms. *J. Graph Theory*, 64(4):330–342, 2010. doi:10.1002/JGT.20461.

10 Michael Elberfeld and Ken-ichi Kawarabayashi. Embedding and canonizing graphs of bounded genus in logspace. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 383–392. ACM, 2014. doi:10.1145/2591796.2591865.

11 Michael Elberfeld and Pascal Schweitzer. Canonizing graphs of bounded tree width in logspace. *ACM Trans. Comput. Theory*, 9(3):12:1–12:29, 2017. doi:10.1145/3132720.

12 Eva Fluck, Tim Seppelt, and Gian Luca Spitzer. Going deep and going wide: Counting logic and homomorphism indistinguishability over graphs of bounded treedepth and treewidth. In Aniello Murano and Alexandra Silva, editors, *32nd EACSL Annual Conference on Computer Science Logic, CSL 2024, February 19-23, 2024, Naples, Italy*, volume 288 of *LIPIcs*, pages 27:1–27:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.CSL.2024.27.

13 Martin Fürer. Weisfeiler-Lehman refinement requires at least a linear number of iterations. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 322–333. Springer, 2001. doi:10.1007/3-540-48224-5_27.

14 Martin Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. *J. ACM*, 59(5):27:1–27:64, 2012. doi:10.1145/2371656.2371662.

15 Martin Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory*, volume 47 of *Lecture Notes in Logic*. Cambridge University Press, 2017. doi:10.1017/9781139028868.

16 Martin Grohe. Counting bounded tree depth homomorphisms. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 507–520. ACM, 2020. doi:10.1145/3373718.3394739.

17 Martin Grohe. The logic of graph neural networks. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–17. IEEE, 2021. doi:10.1109/LICS52264.2021.9470677.

18 Martin Grohe and Sandra Kiefer. A linear upper bound on the Weisfeiler-Leman dimension of graphs of bounded genus. 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 117:1–117:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.ICALP.2019.117.

19 Martin Grohe, Moritz Lichter, Daniel Neuen, and Pascal Schweitzer. Compressing CFI graphs and lower bounds for the Weisfeiler-Leman refinements. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 798–809. IEEE, 2023. doi:10.1109/FOCS57990.2023.00052.

20 Martin Grohe and Daniel Neuen. Canonisation and definability for graphs of bounded rank width. *ACM Trans. Comput. Log.*, 24(1):6:1–6:31, 2023. doi:10.1145/3568025.

21 Jin Guo, Alexander L. Gavrilyuk, and Ilia Ponomarenko. On the Weisfeiler-Leman dimension of permutation graphs. *arXiv:2305.15861 [math.CO]*, May 2023. arXiv. URL: https://arxiv.org/abs/2305.15861, doi:10.48550/arXiv.2305.15861.

22 Lauri Hella. Logical hierarchies in PTIME. *Inf. Comput.*, 129(1):1–19, 1996. doi:10.1006/INCO.1996.0070.

23 Jelle Hellings, Marc Gyssens, Jan Van den Bussche, and Dirk Van Gucht. Expressive completeness of two-variable first-order logic with counting for first-order logic queries on rooted unranked trees. In *Proceedings of the 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, 2023. doi:10.1109/LICS56636.2023.10175828.

24 Neil Immerman. Relational queries computable in polynomial time. *Inf. Control.*, 68(1-3):86–104, 1986. doi:10.1016/S0019-9958(86)80029-8.

25 Neil Immerman. *Descriptive complexity*. Graduate texts in computer science. Springer, 1999. doi:10.1007/978-1-4612-0539-5.

26 Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. *Complexity Theory Retrospective*, pages 59–81, 1990. doi:10.1007/978-1-4612-4478-3_5.

27 Sandra Kiefer. The Weisfeiler-Leman algorithm: an exploration of its power. *ACM SIGLOG News*, 7(3):5–27, 2020. doi:10.1145/3436980.3436982.

28 Sandra Kiefer, Ilia Ponomarenko, and Pascal Schweitzer. The Weisfeiler-Leman dimension of planar graphs is at most 3. *J. ACM*, 66(6):44:1–44:31, 2019. doi:10.1145/3333003.

29 Haiyan Li, Ilia Ponomarenko, and Peter Zeman. On the Weisfeiler-Leman dimension of some polyhedral graphs. *arXiv:2305.17302 [math.CO]*, May 2023. arXiv. doi:10.48550/arXiv.2305.17302.

30 Steven Lindell. A logspace algorithm for tree canonization (extended abstract). In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, editors, *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 400–404. ACM, 1992. doi:10.1145/129712.129750.

31 Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. 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/f81dee42585b3814de199b2e88757f5c-Abstract.html.

32 Jaroslav Nesetril and Patrice Ossona de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27(6):1022–1041, 2006. doi:10.1016/J.EJC.2005.01.010.

33 Martin Otto. *Bounded Variable Logics and Counting: A Study in Finite Models*, volume 9 of *Lecture Notes in Logic*. Cambridge University Press, 2017. doi:10.1017/9781316716878.

34 Oleg Pikhurko and Oleg Verbitsky. Logical complexity of graphs: A survey. In Martin Grohe and Johann A. Makowsky, editors, *Model Theoretic Methods in Finite Combinatorics - AMS-ASL Joint Special Session, Washington, DC, USA, January 5-8, 2009*, volume 558 of *Contemporary Mathematics*, pages 129–180. American Mathematical Society, 2009.

35 Jamie Tucker-Foltz. Inapproximability of unique games in fixed-point logic with counting. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470706.

36 Steffen van Bergerem. Learning concepts definable in first-order logic with counting. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785811.

37 Steffen van Bergerem. *Descriptive complexity of learning*. PhD thesis, RWTH Aachen University, Germany, 2023. URL: https://publications.rwth-aachen.de/record/953243, doi:10.18154/RWTH-2023-02554.

38 E. R. van Dam, J. H. Koolen, and H. Tanaka. Distance-regular graphs. *Electronic Journal of Combinatorics*, 1(DynamicSurveys), 2018. doi:10.37236/4925.

39 Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 137–146. ACM, 1982. doi:10.1145/800070.802186.

40 Qing Wang, Dillon Ze Chen, Asiri Wijesinghe, Shouheng Li, and Muhammad Farhan. $\mathcal{N}$-WL: A new hierarchy of expressivity for graph neural networks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL: https://openreview.net/pdf?id=5cAI0qXxyv.

On the VC Dimension of First-Order Logic with Counting and Weight Aggregation

Steffen van Bergerem
Humboldt-Universität zu Berlin, Germany

Nicole Schweikardt
Humboldt-Universität zu Berlin, Germany

Abstract

We prove optimal upper bounds on the Vapnik–Chervonenkis density of formulas in the extensions of first-order logic with counting (FOC_1) and with weight aggregation (FOWA_1) on nowhere dense classes of (vertex- and edge-)weighted finite graphs. This lifts a result of Pilipczuk, Siebertz, and Toruńczyk [14] from first-order logic on ordinary finite graphs to substantially more expressive logics on weighted finite graphs. Moreover, this proves that every FOC_1 formula and every FOWA_1 formula has bounded Vapnik–Chervonenkis dimension on nowhere dense classes of weighted finite graphs; thereby, it lifts a result of Adler and Adler [1] from first-order logic to FOC_1 and FOWA_1.

Generalising another result of Pilipczuk, Siebertz, and Toruńczyk [14], we also provide an explicit upper bound on the ladder index of FOC_1 and FOWA_1 formulas on nowhere dense classes. This shows that nowhere dense classes of weighted finite graphs are FOC_1-stable and FOWA_1-stable.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases VC dimension, VC density, stability, nowhere dense graphs, first-order logic with weight aggregation, first-order logic with counting

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.15

Funding This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 541000908 (gefördert durch die Deutsche Forschungsgemeinschaft (DFG) – Projektnummer 541000908).

Acknowledgements We thank the anonymous reviewers for their valuable comments that helped to improve the presentation of this paper.

1 Introduction

The *Vapnik–Chervonenkis dimension* (for short: *VC dimension*) is a measure for the complexity of set systems; it was introduced in the 1970s [19, 17, 16] and has been widely studied since then. It is formally defined as follows. Let X be a set and let $\mathcal{F} \subseteq 2^X$ be a family of subsets of X. A set $Y \subseteq X$ is *shattered by* $\mathcal{F}$ if every subset of Y can be obtained as the intersection of Y with some $F \in \mathcal{F}$, i. e., $\{Y \cap F : F \in \mathcal{F}\} = 2^Y$. The *VC dimension* of $\mathcal{F}$ is the maximum size of a set $Y \subseteq X$ that is shattered by $\mathcal{F}$ (or ∞, if this maximum does not exist).

Given a logical formula $\varphi(\bar{x}, \bar{y})$ with its free variables partitioned into a k-tuple $\bar{x}$ and an ℓ-tuple $\bar{y}$, the *VC dimension* of $\varphi(\bar{x}, \bar{y})$ on a graph $G = (V(G), E(G))$ is defined as the VC dimension of the family $S^\varphi(G/V(G)) := S^\varphi_G(V(G)/V(G))$, where for $V, W \subseteq V(G)$ we let

$$S^\varphi_G(V/W) := \{\mathrm{tp}^\varphi_G(\bar{v}/W) : \bar{v} \in V^k\}, \quad \text{where} \quad \mathrm{tp}^\varphi_G(\bar{v}/W) := \{\bar{w} \in W^\ell : G \models \varphi[\bar{v}, \bar{w}]\}.$$

We say that $\varphi(\bar{x}, \bar{y})$ *has bounded VC dimension* on a class $\mathcal{C}$ of graphs if there is a number c such that for every $G \in \mathcal{C}$ the VC dimension of $\varphi(\bar{x}, \bar{y})$ on G is at most c. In the following, all graphs considered in this paper are finite.

Motivated by applications on the learnability of concept classes in the model of *Probably Approximately Correct (PAC)* learning, Grohe and Turán [9] showed that every first-order formula $\varphi(\bar{x}, \bar{y})$ has bounded VC dimension on classes of graphs of bounded local clique-width (this, in particular, includes planar graphs). Adler and Adler [1] generalised this to all *nowhere dense* classes of graphs. The notion of nowhere dense classes was introduced by Nešetřil and Ossona de Mendez [12, 11] as a formalisation of classes of "sparse" graphs. It subsumes and extends many well-known classes of sparse graphs, including planar graphs, trees, classes of graphs of bounded tree-width or bounded degree, and all classes that exclude a fixed topological minor. It is a robust notion that has numerous equivalent characterisations; for details we refer to the book [13].

The goal of the present paper is to lift Adler and Adler's result [1] from first-order logic FO to the substantially more expressive logics FOC_1 and FOWA_1 (introduced in [8, 5]) that enrich FO by mechanisms for counting and for weight aggregation. An obstacle in achieving this is that the proof in [1] relies on model-theoretic results of [15] based on the compactness of FO – and these are not available for FOC_1 or FOWA_1. Fortunately, Pilipczuk, Siebertz and Toruńczyk [14] presented a different, constructive proof of Adler and Adler's result. Their proof is based on Gaifman locality and Feferman–Vaught decompositions of FO. Similar locality results and decompositions were achieved for FOC_1 and FOWA_1 in [8, 5].

The logic FOC (first-order logic with counting terms) was introduced in [10] and further studied in [8, 3]. This logic extends FO by the ability to formulate *counting terms* that evaluate to integers, and by *numerical predicates* that allow to compare counting terms. If φ is a formula with free variables $\bar{x} = (x_1, \ldots, x_k)$ and $\bar{y} = (y_1, \ldots, y_\ell)$, then $\#\bar{y}.\varphi$ is a counting term with free variables $\bar{x}$ that specifies the number of tuples $\bar{y}$ that satisfy the formula φ. Apart from this, every fixed integer is a counting term; and if t_1 and t_2 are counting terms, then so are $(t_1 + t_2)$ and $(t_1 \cdot t_2)$. The results of terms can be combined into a formula by means of numerical predicates: an m-ary numerical predicate P is an m-ary relation on the integers (e. g. $\mathsf{P}_{\leqslant}$ is the binary relation consisting of all pairs (i, j) of integers where $i \leqslant j$). The logic FOC allows formulas of the form $\mathsf{P}(t_1, \ldots, t_m)$ that evaluate to "true" if and only if the m-tuple of integers obtained by evaluating the counting terms $t_1, \ldots, t_m$ belongs to the relation P.

The logic FOWA (first-order logic with weight aggregation) was introduced in [5]. Formulas and terms of this logic are evaluated on *weighted graphs*, which extend ordinary undirected graphs by assigning weights (i. e., elements from particular rings or abelian groups) to vertices or edges present in the graph. Pairs that do not occur as edges of the graph receive the weight 0, i. e., the neutral element of the ring or abelian group. FOWA extends FO by the ability to formulate *(weight aggregation) terms* that evaluate to elements in the given ring (or abelian group), and by predicates that allow to compare these terms. Every fixed element of the ring or abelian group is a term, as well as every expression of the form $\mathtt{w}(x)$ or $\mathtt{w}(x, y)$; the latter yields the weight of vertex x and edge (x, y), respectively. If φ is a formula with free variables $\bar{x} = (x_1, \ldots, x_k)$ and $\bar{y} = (y_1, \ldots, y_\ell)$, then $\sum \mathtt{w}(\bar{y}).\varphi$ is a (weight aggregation) term with free variables $\bar{x}$ that specifies the sum (w.r.t. the ring or abelian group) of the weights of all tuples $\bar{y}$ for which the formula φ is satisfied. More generally, instead of a single expression $\mathtt{w}(\bar{y})$, the term may also refer to a product (w.r.t. the given ring) of such expressions and fixed elements of the ring. Analogously as for FOC, terms can be combined using the operations present in the ring or abelian group; and the results of terms can be combined into a formula by means of predicates on the ring or abelian group: a formula of the form $\mathsf{P}(t_1, \ldots, t_m)$ expresses that the m-tuple of elements in the ring or abelian group obtained by evaluating the terms $t_1, \ldots, t_m$ belongs to the relation P.

FOC can be viewed as a special case of FOWA where the ring is the ring of integers, and every vertex of the graph is equipped with the weight 1. Thus, all results that are available for (fragments of) FOWA immediately translate into analogous results on (the corresponding fragment of) FOC (but not necessarily vice versa).

For each number n, the fragments FOC_n and FOWA_n of FOC and FOWA restrict subformulas of the form $\mathsf{P}(t_1, \ldots, t_m)$ to have at most n free variables.

In this paper, we follow the approach of Pilipczuk, Siebertz and Toruńczyk [14] and extend it to FOC and FOWA by utilising results of van Bergerem and Schweikardt [5] and Grohe and Schweikardt [8]. Our main results are as follows.

(1) There is a formula $\varphi(x, y)$ of FOC_2 that has unbounded VC dimension on the class $\mathcal{T}_3$ of unranked trees of height $\leqslant 3$ (note that $\mathcal{T}_3$ is nowhere dense). (Theorem 3.1)

(2) Every formula $\varphi(\bar{x}, \bar{y})$ of FOC_1 or FOWA_1 has bounded VC dimension on every nowhere dense class $\mathcal{C}$ of weighted graphs. (Corollary 5.3)

Result (1) is obtained by representing arbitrary graphs G via unranked trees T_G of height 3 in the same way as in [8]. Then, arbitrary FO formulas on G can be translated into corresponding FOC_2 formulas on T_G. By applying this translation to the formula $E(x, y)$, which has unbounded VC dimension on the class of all graphs, one obtains Result (1).

For obtaining Result (2), we combine the approach of [14] with the locality results of [8, 5]. This allows us to lift the following key result of [14] from FO to FOC_1 and FOWA_1.

(3) For every nowhere dense class $\mathcal{C}$ of weighted graphs, for every formula $\varphi(\bar{x}, \bar{y})$ of FOWA_1 or FOC_1, and for and every $\varepsilon > 0$, there exists a number c such that for every $G \in \mathcal{C}$ and every non-empty $W \subseteq V(G)$, we have $|S^\varphi(G/W)| \leqslant c \cdot |W|^{|\bar{x}|+\varepsilon}$, where $S^\varphi(G/W) := S^\varphi_G(V(G), W)$. (Theorem 5.1)

As an immediate consequence of this, by definition, we obtain the following result.

(4) Every formula $\varphi(\bar{x}, \bar{y})$ of FOWA_1 or FOC_1 has VC density at most $|\bar{x}|$ on every nowhere dense class $\mathcal{C}$ of weighted graphs. (Corollary 5.2)

Here, the *VC density* of $\varphi(\bar{x}, \bar{y})$ on $\mathcal{C}$ is defined as the infimum of all reals $\alpha > 0$ such that $|S^\varphi(G/W)| \in \mathcal{O}(|W|^\alpha)$, for all $G \in \mathcal{C}$ and all $W \subseteq V(G)$ (where constants hidden in the $\mathcal{O}$-notation may depend on α). We want to remark that Result (4) implies Result (2), because the VC dimension is finite if and only if the VC density is finite (see, e. g., [2]).

For proving Result (3), we rely on a technical main lemma (see Lemma 4.1). The same statement was proven in [14] for FO instead of FOWA_1. Lifting this from FO to FOWA_1 (and FOC_1) was one of the main technical obstacles we had to overcome in this paper.

From [14], we know that the bounds provided by Results (3) and (4) are optimal (since FO is included in FOC_1 and FOWA_1) and, furthermore, that Results (2)–(4) cannot be extended to classes that are not nowhere dense but closed under taking subgraphs.

As another application of our main technical lemma (Lemma 4.1), we provide upper bounds (Theorem 6.1) on the *ladder index*, which is defined as follows. For a FOWA_1 formula $\varphi(\bar{x}, \bar{y})$, a *$\varphi$-ladder* of length L in a weighted graph G is a sequence $\bar{v}_1, \ldots, \bar{v}_L, \bar{w}_1, \ldots, \bar{w}_L$ such that $\bar{v}_i \in \big(V(G)\big)^{|\bar{x}|}$ and $\bar{w}_i \in \big(V(G)\big)^{|\bar{y}|}$ for all $i \in [L]$, and, for all $i, j \in [L]$, it holds that $G \models \varphi[\bar{v}_i, \bar{w}_j]$ if and only if $i \leqslant j$. The smallest L for which there is no φ-ladder of length L in G is called the *ladder index of φ in G*.

A class $\mathcal{C}$ of graphs is called *stable* if the ladder index of every first-order formula φ in every graph from $\mathcal{C}$ is bounded by a constant depending only on φ and $\mathcal{C}$ [18]. Adler and Adler [1] showed that every nowhere dense class of graphs is stable. Using our bound on the ladder index (Theorem 6.1), we obtain the following result, which also implies Result (2).

(5) Every nowhere dense class $\mathcal{C}$ of weighted graphs is FOC_1-stable and FOWA_1-stable, that is, the ladder index of every FOWA_1 formula (and therefore also of every FOC_1 formula) φ in every weighted graph from $\mathcal{C}$ is bounded by a constant depending only on φ and $\mathcal{C}$. (Corollary 6.2)

The remainder of the paper is structured as follows. Section 2 provides the necessary background on graphs, nowhere dense classes, the logics FOC and FOWA, and the locality results that are known for these logics and used in our proofs. Section 3 presents the proof of Result (1). Section 4 is devoted to the main technical lemma (Lemma 4.1). In Section 5, we utilise this lemma to prove our Results (2)–(4). Section 6 proves Result (5) based on Lemma 4.1. We conclude in Section 7.

2 Preliminaries

We let $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{N}_{\geqslant 1}$, $\mathbb{Q}_{>0}$ denote the sets of integers, non-negative integers, positive integers, and positive rationals, respectively. For $m, n \in \mathbb{Z}$, we let $[m, n] \coloneqq \{\ell \in \mathbb{Z} : m \leqslant \ell \leqslant n\}$ and $[n] \coloneqq [1, n]$. For a k-tuple $\bar{v} = (v_1, \ldots, v_k)$, we write $|\bar{v}|$ to denote its *length* k. We denote the power set of a set S by 2^S.

A *group* $(G, \circ)$ is a set G equipped with a binary operator $\circ\colon G \times G \to G$ that is associative (i. e. $(a \circ b) \circ c = a \circ (b \circ c)$ for all $a, b, c \in G$) and has a neutral element $e_G \in G$ (i. e. $a \circ e_G = e_G \circ a = a$ for all $a \in G$) such that each $a \in G$ has an inverse $a' \in G$ (i. e. $a \circ a' = a' \circ a = e_G$); we write a^{-1} for this a'. A group is *abelian* if $\circ$ is commutative (i. e. $a \circ b = b \circ a$ for all $a, b \in G$). A *ring* $(R, +, \cdot)$ is a set R equipped with two binary operators $+$ (*addition*) and $\cdot$ (*multiplication*) such that $(R, +)$ is an abelian group with neutral element $0_R \in R$, $\cdot$ is associative and has a neutral element $1_R \in R$, and multiplication is distributive with respect to addition, i. e. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ and $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$ for all $a, b, c \in R$. A ring is *commutative* if $\cdot$ is commutative.

When referring to an abelian group (or ring), we will usually write $(S, +_S)$ (or $(S, +_S, \cdot_S)$), we denote the neutral element of the group by 0_S, and $-a$ denotes the inverse of an element a in $(S, +_S)$ (and we denote the neutral element of the ring for $(S, \cdot_S)$ by 1_S).

σ-Graphs

A (simple, undirected and finite) graph $G = (V(G), E(G))$ consists of a finite set $V(G)$ (the vertices of G) and a set $E(G)$ of subsets of $V(G)$ of size 2 (the edges of G).

A *graph signature* σ is a finite set consisting of a symbol E and a finite number of further symbols. The symbol E as *arity* $\mathrm{ar}(E) = 2$, while all other symbols $R \in \sigma \setminus \{E\}$ have arity $\mathrm{ar}(R) \in \{0, 1\}$. Let σ be a graph signature. A *σ-graph* G consists of a graph $(V(G), E(G))$, and a relation $R(G) \subseteq \big(V(G)\big)^{\mathrm{ar}(R)}$ for every $R \in \sigma \setminus \{E\}$. Note that relations of arity 1 are subsets of $V(G)$, and since $S^0 = \{()\}$ for every set S, there exist only two relations of arity 0, namely $\emptyset$ and $\{()\}$. We identify the latter with `true` and the former with `false`.

The *order* of a σ-graph G is $|G| \coloneqq |V(G)|$.

Weighted σ-Graphs

Let σ be a graph signature. Let $\mathbb{S}$ be a collection of rings and/or abelian groups. Let $\mathbf{W}$ be a finite set of *weight symbols* such that each $\mathtt{w} \in \mathbf{W}$ has an associated *arity* $\mathrm{ar}(\mathtt{w}) \in \{1, 2\}$ and a *type* $\mathrm{type}(\mathtt{w}) \in \mathbb{S}$. A *$(\sigma, \mathbf{W})$-graph* (or, *$\mathbf{W}$-weighted σ-graph*) is a σ-graph G that is enriched, for every $\mathtt{w} \in \mathbf{W}$, by an interpretation $\mathtt{w}^G\colon \big(V(G)\big)^{\mathrm{ar}(\mathtt{w})} \to \mathrm{type}(\mathtt{w})$, which satisfies the following *edge condition* for all $\mathtt{w} \in \mathbf{W}$ with $\mathrm{ar}(\mathtt{w}) = 2$: if $\mathtt{w}^G(v_1, v_2) \neq 0_S$ for $S \coloneqq \mathrm{type}(\mathtt{w})$, and $v_1, v_2 \in V(G)$, then $\{v_1, v_2\} \in E(G)$.

Standard notions used for graphs are defined for $(\mathbf{W}, \sigma)$-graphs G by referring to their *Gaifman graph* $(V(G), E(G))$. In particular, a *path* between two vertices u and v in G is a path between u and v in the graph $(V(G), E(G))$, and the *distance* $\mathrm{dist}^G(u, v)$ between vertices u and v is their distance in the graph $(V(G), E(G))$. The *degree* $\deg(G)$ is the maximum degree of $(V(G), E(G))$.

For a set $X \subseteq V(G)$, the *induced subgraph of G on X* is the $(\sigma, \mathbf{W})$-graph $G[X]$ with vertex set $V(G[X]) = X$, edge set $E(G[X]) = \{e \in E(G) : e \subseteq X\}$, relations $R(G[X]) = R(G) \cap X^{\mathrm{ar}(R)}$ for every $R \in \sigma \setminus \{E\}$, and weights $\mathtt{w}^{G[X]}(\bar{v}) = \mathtt{w}^G(\bar{v})$ for every $\mathtt{w} \in \mathbf{W}$ and every $\bar{v} \in X^{\mathrm{ar}(\mathtt{w})}$. For a $(\sigma, \mathbf{W})$-graph G and a set $S \subseteq V(G)$, we let $G \setminus S := G[V(G) \setminus S]$.

For a number $r \geqslant 0$, the *r-ball* around a vertex $v \in V(G)$ is $N_r^G(v) := \{u \in V(G) : \mathrm{dist}^G(v,u) \leqslant r\}$, and the r-ball around a set $S \subseteq V(G)$ is $N_r^G(S) := \bigcup_{v \in S} N_r^G(v)$. The *$r$-neighbourhood around S* is the $(\sigma, \mathbf{W})$-graph $\mathcal{N}_r^G(S) := G[N_r^G(S)]$. For a tuple $\bar{a} = (a_1, \ldots, a_k) \in V(G)^k$ we let $\mathcal{N}_r^G(\bar{a}) := \mathcal{N}_r^G(S)$ and $N_r^G(\bar{a}) := N_r^G(S)$ for $S := \{a_1, \ldots, a_k\}$.

Let σ' be a graph signature with $\sigma' \supseteq \sigma$, and let $\mathbf{W}'$ be a finite set of weight symbols with $\mathbf{W}' \supseteq \mathbf{W}$. A $(\sigma', \mathbf{W}')$-graph G' is a *$(\sigma', \mathbf{W}')$-expansion* of a $(\sigma, \mathbf{W})$-graph G if $V(G') = V(G)$, $R(G') = R(G)$ for all $R \in \sigma$, and $\mathtt{w}^{G'} = \mathtt{w}^G$ for every $\mathtt{w} \in \mathbf{W}$. If G' is a $(\sigma', \mathbf{W}')$-expansion of the $(\sigma, \mathbf{W})$-graph G, then G is the *$(\sigma, \mathbf{W})$-reduct* of G'.

Let G and H be two $(\sigma, \mathbf{W})$-graphs with $V(G) \cap V(H) = \emptyset$. The *disjoint union* of G and H is the $(\sigma, \mathbf{W})$-graph $G \uplus H$ with vertex set $V(G \uplus H) = V(G) \cup V(H)$, and $R(G \uplus H) = R(G) \cup R(H)$ for all $R \in \sigma$, and weight functions as follows: For all unary $\mathtt{w} \in \mathbf{W}$ we have $\mathtt{w}^{G \uplus H}(v) = \mathtt{w}^G(v)$ for all $v \in V(G)$ and $\mathtt{w}^{G \uplus H}(v) = \mathtt{w}^H(v)$ for all $v \in V(H)$. For all binary $\mathtt{w} \in \mathbf{W}$ we have $\mathtt{w}^{G \uplus H}(u,v) = \mathtt{w}^G(u,v)$ for all $(u,v) \in V(G)^2$, $\mathtt{w}^{G \uplus H}(u,v) = \mathtt{w}^H(u,v)$ for all $(u,v) \in V(H)^2$, and $\mathtt{w}^{G \uplus H}(u,v) = 0_S$ for all $(u,v) \in (V(G) \times V(H)) \cup (V(H) \times V(G))$, where $S = \mathrm{type}(\mathtt{w})$.

Nowhere Dense Classes

For $n \in \mathbb{N}$, we write K_n for the complete graph on n vertices. A *depth-n minor* of a graph $G = (V(G), E(G))$ is a subgraph of a graph obtained from G by contracting mutually vertex-disjoint connected subgraphs of radius at most n to single vertices.

As mentioned in Section 1, the notion of nowhere dense classes of graphs is a robust notion that has numerous equivalent characterisations; for an overview we refer to the introduction of [14]; details can be found in the book [13]. For the purpose of this paper, the following characterisation serves as our definition of the notion.

▶ **Definition 2.1.** A class $\mathcal{C}$ of graphs is *nowhere dense* if there is a function $t\colon \mathbb{N} \to \mathbb{N}$ such that for every $r \in \mathbb{N}$, no graph $G \in \mathcal{C}$ contains the complete graph $K_{t(r)}$ as a depth-r minor. A class $\mathcal{C}$ of $(\sigma, \mathbf{W})$-graphs is nowhere dense if and only if the class $\{(V(G), E(G)) : G \in \mathcal{C}\}$ is nowhere dense.

The following theorem was proved in [14] (there, it was formulated for classes of graphs; here we adapted the formulation to classes of $(\sigma, \mathbf{W})$-graphs). We will use this result for proving our results on VC density in Section 5. The result uses the following notion. Let G be a $(\sigma, \mathbf{W})$-graph, let $r \in \mathbb{N}$, and let $V, W, S \subseteq V(G)$. We say that *V and W are r-separated by S (in G)* if every path of length at most r in G from a vertex in V to a vertex in W contains a vertex from S. This notion naturally extends to tuples $\bar{v} = (v_1, \ldots, v_k)$ and $\bar{w} = (w_1, \ldots, w_\ell)$ for any $k, \ell \in \mathbb{N}_{\geqslant 1}$ by considering the sets $\{v_1, \ldots, v_k\}$ and $\{w_1, \ldots, w_\ell\}$, and it thereby also naturally extends to sets of tuples V and W.

▶ **Theorem 2.2** (Uniform quasi-wideness for tuples [14, Theorem 2.9])**.** *Let $r, t \in \mathbb{N}$, and let $\mathcal{C}$ be a class of $(\sigma, \mathbf{W})$-graphs G whose Gaifman graph $(V(G), E(G))$ does not include K_t as a depth-$18r$ minor. For every $d \in \mathbb{N}$, there is a number s and a polynomial $N\colon \mathbb{N} \to \mathbb{N}$ computable from r, t, and d with the following property.*

For every $G \in \mathcal{C}$, every $m \in \mathbb{N}$, and every set $X \subseteq (V(G))^d$ with $|X| \geqslant N(m)$, there are sets $S \subseteq V(G)$ and $Y \subseteq X$ with $|S| \leqslant s$ and $|Y| \geqslant m$ such that all distinct $\bar{v}, \bar{v}' \in Y$ are r-separated by S in G.

The Weight Aggregation Logic FOWA

Fix a countably infinite set vars of *variables*. A $(\sigma, \mathbf{W})$-*interpretation* $\mathcal{I} = (G, \beta)$ consists of a $(\sigma, \mathbf{W})$-graph G and an *assignment* $\beta\colon \mathsf{vars} \to V(G)$. For $k \in \mathbb{N}_{\geqslant 1}$, elements $a_1, \ldots, a_k \in V(G)$, and k distinct variables $y_1, \ldots, y_k$, we write $\mathcal{I}\frac{a_1,\ldots,a_k}{y_1,\ldots,y_k}$ for the interpretation $(G, \beta\frac{a_1,\ldots,a_k}{y_1,\ldots,y_k})$, where $\beta\frac{a_1,\ldots,a_k}{y_1,\ldots,y_k}$ is the assignment β' with $\beta'(y_i) = a_i$ for every $i \in [k]$ and $\beta'(z) = \beta(z)$ for all $z \in \mathsf{vars} \setminus \{y_1, \ldots, y_k\}$.

Recall that $\mathbb{S}$ is a collection of rings and/or abelian groups. An $\mathbb{S}$-*predicate collection* is a 4-tuple $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$, where $\mathbb{P}$ is a countable set of *predicate names* and, to each $\mathsf{P} \in \mathbb{P}$, ar assigns an *arity* $\mathrm{ar}(\mathsf{P}) \in \mathbb{N}_{\geqslant 1}$, type assigns a *type* $\mathrm{type}(\mathsf{P}) \in \mathbb{S}^{\mathrm{ar}(\mathsf{P})}$, and $[\![\cdot]\!]$ assigns a *semantics* $[\![\mathsf{P}]\!] \subseteq \mathrm{type}(\mathsf{P})$. For the remainder of this paper, fix an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$.

For every $S \in \mathbb{S}$ that is not a ring but just an abelian group, a $\mathbf{W}$-*product of type* S is either an element in S or an expression of the form $\mathtt{w}(\bar{z})$, where $\mathtt{w} \in \mathbf{W}$ is of type S and either $\mathrm{ar}(\mathtt{w}) = 1$ and $\bar{z}$ is a single variable, or $\mathrm{ar}(\mathtt{w}) = 2$ and $\bar{z} = (z_1, z_2)$ for distinct variables z_1, z_2.

For every ring $S \in \mathbb{S}$, a $\mathbf{W}$-*product of type* S is an expression of the form $t_1 \cdot \cdots \cdot t_\ell$, where $\ell \in \mathbb{N}_{\geqslant 1}$, and for each $i \in [\ell]$, either $t_i \in S$ or there exists a $\mathtt{w} \in \mathbf{W}$ with $\mathrm{type}(\mathtt{w}) = S$ and either $\mathrm{ar}(\mathtt{w}) = 1$ and t_i is of the form $\mathtt{w}(z)$ for a variable z or $\mathrm{ar}(\mathtt{w}) = 2$ and t_i is of the form $\mathtt{w}(z_1, z_2)$ for distinct variables z_1, z_2. By $\mathrm{vars}(p)$, we denote the set of all variables that occur in a $\mathbf{W}$-product p. The syntax and semantics of first-order logic with weight aggregation FOWA is defined as follows.

▶ **Definition 2.3.** *For* $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*, the set of* formulas *and* $\mathbb{S}$-terms *is built according to the following rules.*

(1) $x_1{=}x_2$ *and* $R(x_1, \ldots, x_k)$ *are formulas for* $x_1, \ldots, x_k \in \mathsf{vars}$ *and* $R \in \sigma$ *with* $\mathrm{ar}(R) = k$.

(2) *If* $\mathtt{w} \in \mathbf{W}$*,* $S = \mathrm{type}(\mathtt{w})$*,* $s \in S$*,* $k = \mathrm{ar}(\mathtt{w})$*, and* $\bar{x} = (x_1, \ldots, x_k)$ *is a tuple of* k *pairwise distinct variables, then* $\big(s = \mathtt{w}(\bar{x})\big)$ *is a formula.*

(3) *If* φ *and* ψ *are formulas, then* $\neg\varphi$ *and* $(\varphi \vee \psi)$ *are also formulas.*

(4) *If* φ *is a formula and* $x \in \mathsf{vars}$*, then* $\exists x\, \varphi$ *is a formula.*

(5) *If* φ *is a formula,* $\mathtt{w} \in \mathbf{W}$*,* $S = \mathrm{type}(\mathtt{w})$*,* $s \in S$*,* $k = \mathrm{ar}(\mathtt{w})$*, and* $\bar{x} = (x_1, \ldots, x_k)$ *is a tuple of* k *pairwise distinct variables, then* $\big(s = \sum \mathtt{w}(\bar{x}).\varphi\big)$ *is a formula.*

(6) *If* $\mathsf{P} \in \mathbb{P}$*,* $m = \mathrm{ar}(\mathsf{P})$*, and* $t_1, \ldots, t_m$ *are* $\mathbb{S}$*-terms with* $\mathrm{type}(\mathsf{P}) = \big(\mathrm{type}(t_1), \ldots, \mathrm{type}(t_m)\big)$*, then* $\mathsf{P}(t_1, \ldots, t_m)$ *is a formula.*

(7) *For every* $S \in \mathbb{S}$ *and every* $s \in S$*,* s *is an* $\mathbb{S}$*-term of type* S*.*

(8) *For every* $S \in \mathbb{S}$*, every* $\mathtt{w} \in \mathbf{W}$ *of type* S*, and every tuple* $(x_1, \ldots, x_k)$ *of* $k := \mathrm{ar}(\mathtt{w})$ *pairwise distinct variables in* vars*,* $\mathtt{w}(x_1, \ldots, x_k)$ *is an* $\mathbb{S}$*-term of type* S*.*

(9) *If* t_1 *and* t_2 *are* $\mathbb{S}$*-terms of the same type* S*, then* $(t_1 + t_2)$ *and* $(t_1 - t_2)$ *are also* $\mathbb{S}$*-terms of type* S*; furthermore, if* S *is a ring (and not just an abelian group), then also* $(t_1{\cdot}t_2)$ *is an* $\mathbb{S}$*-term of type* S*.*

(10) *If* φ *is a formula,* $S \in \mathbb{S}$*, and* p *is a* $\mathbf{W}$*-product of type* S*, then* $\sum p.\varphi$ *is an* $\mathbb{S}$*-term of type* S*.*

Let $\mathcal{I} = (G, \beta)$ *be a* $(\sigma, \mathbf{W})$*-interpretation. For a formula or* $\mathbb{S}$*-term* ξ *from* $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*, the semantics* $[\![\xi]\!]^{\mathcal{I}}$ *is defined as follows.*

(1) $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 1$ *if* $\beta(x_1) = \beta(x_2)$*, and* $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 0$ *otherwise;* $[\![E(x_1, x_2)]\!]^{\mathcal{I}} = 1$ *if* $\{\beta(x_1), \beta(x_2)\} \in E(G)$*, and* $[\![E(x_1, x_2)]\!]^{\mathcal{I}} = 0$ *otherwise; for all* $R \in \sigma$ *with* $\mathrm{ar}(R) = 1$*, we have* $[\![R(x_1)]\!]^{\mathcal{I}} = 1$ *if* $\beta(x_1) \in R(G)$*, and* $[\![R(x_1)]\!]^{\mathcal{I}} = 0$ *otherwise; for all* $R \in \sigma$ *with* $\mathrm{ar}(R) = 0$*, we have* $[\![R()]\!]^{\mathcal{I}} = 1$ *if* $() \in R(G)$*, and* $[\![R()]\!]^{\mathcal{I}} = 0$ *otherwise.*

(2) $\big[\!\big[\big(s = \mathtt{w}(\bar{x})\big)\big]\!\big]^{\mathcal{I}} = 1$ *if* $s = \mathtt{w}^G\big(\beta(x_1), \ldots, \beta(x_k)\big)$*, and* $\big[\!\big[\big(s = \mathtt{w}(\bar{x})\big)\big]\!\big]^{\mathcal{I}} = 0$ *otherwise.*

(3) $[\![\neg\varphi]\!]^{\mathcal{I}} = 1 - [\![\varphi]\!]^{\mathcal{I}}$ *and* $[\![(\varphi \vee \psi)]\!] = \max\big\{[\![\varphi]\!]^{\mathcal{I}}, [\![\psi]\!]^{\mathcal{I}}\big\}$.

(4) $[\![\exists x\, \varphi]\!]^{\mathcal{I}} = \max\big\{[\![\varphi]\!]^{\mathcal{I}\frac{v}{x}} \;:\; v \in V(G)\big\}$.

(5) $[\![(s = \sum \mathtt{w}(\bar{x}).\varphi)]\!]^{\mathcal{I}} = 1$ *if* $s = \sum_S \{\mathtt{w}^G(\bar{v}) : \bar{v} = (v_1,\dots,v_k) \in (V(G))^k$ *with* $[\![\varphi]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} = 1\}$, *and* $[\![(s = \sum \mathtt{w}(\bar{x}).\varphi)]\!]^{\mathcal{I}} = 0$ *otherwise. As usual,* $\sum_S X = 0_S$ *if* $X = \emptyset$.
(6) $[\![\mathsf{P}(t_1,\dots,t_m)]\!]^{\mathcal{I}} = 1$ *if* $([\![t_1]\!]^{\mathcal{I}},\dots,[\![t_m]\!]^{\mathcal{I}}) \in [\![\mathsf{P}]\!]$, *and* $[\![\mathsf{P}(t_1,\dots,t_m)]\!]^{\mathcal{I}} = 0$ *otherwise.*
(7) $[\![s]\!]^{\mathcal{I}} = s$ *for* $s \in S$ *for some* $S \in \mathbb{S}$.
(8) $[\![\mathtt{w}(x_1,\dots,x_k)]\!]^{\mathcal{I}} = \mathtt{w}^G(\beta(x_1),\dots,\beta(x_k))$.
(9) $[\![(t_1 * t_2)]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} *_S [\![t_2]\!]^{\mathcal{I}}$, *for* $* \in \{+,-,\cdot\}$.
(10) $[\![\sum p.\varphi]\!]^{\mathcal{I}} = \sum_S \{[\![p]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} : v_1,\dots,v_k \in V(G), [\![\varphi]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} = 1\}$, *where* $\mathrm{vars}(p) = \{x_1,\dots,x_k\}$, $k = |\mathrm{vars}(p)|$ *and* $[\![p]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} \cdot_S \cdots \cdot_S [\![t_\ell]\!]^{\mathcal{I}}$ *if* $p = t_1 \cdot \cdots \cdot t_\ell$ *is of type* S.

An *expression* is a formula or an $\mathbb{S}$-term. The set $\mathrm{vars}(\xi)$ of an expression ξ is defined as the set of all variables in vars that occur in ξ. The *free variables* $\mathrm{free}(\xi)$ of ξ are inductively defined as follows.

(1) $\mathrm{free}(x_1{=}x_2) = \{x_1,x_2\}$ and $\mathrm{free}\big(R(x_1,\dots,x_k)\big) = \{x_1,\dots,x_k\}$ for $R \in \sigma$.
(2) $\mathrm{free}\Big(\big(s = \mathtt{w}(x_1,\dots,x_k)\big)\Big) = \{x_1,\dots,x_k\}$.
(3) $\mathrm{free}(\neg\varphi) = \mathrm{free}(\varphi)$ and $\mathrm{free}(\varphi \vee \psi) = \mathrm{free}(\varphi) \cup \mathrm{free}(\psi)$.
(4) $\mathrm{free}(\exists x\, \varphi) = \mathrm{free}(\varphi) \setminus \{x\}$.
(5) $\mathrm{free}\Big(\big(s = \sum \mathtt{w}(x_1,\dots,x_k).\varphi\big)\Big) = \mathrm{free}(\varphi) \setminus \{x_1,\dots,x_k\}$,
(6) $\mathrm{free}\big(\mathsf{P}(t_1,\dots,t_m)\big) = \bigcup_{i=1}^m \mathrm{free}(t_i)$.
(7) $\mathrm{free}(s) = \emptyset$ for $s \in S$ for some $S \in \mathbb{S}$.
(8) $\mathrm{free}\big(\mathtt{w}(x_1,\dots,x_k)\big) = \{x_1,\dots,x_k\}$.
(9) $\mathrm{free}\big((t_1 * t_2)\big) = \mathrm{free}(t_1) \cup \mathrm{free}(t_2)$ for $* \in \{+,-,\cdot\}$.
(10) $\mathrm{free}(\sum p.\varphi) = \mathrm{free}(\varphi) \setminus \mathrm{vars}(p)$.

We write $\xi(x_1,\dots,x_k)$ to indicate that $\mathrm{free}(\xi) \subseteq \{x_1,\dots,x_k\}$. A *sentence* is a formula without free variables, and a *ground* $\mathbb{S}$*-term* is an $\mathbb{S}$-term without free variables.

For a formula φ and a $(\sigma,\mathbf{W})$-interpretation $\mathcal{I}$, we write $\mathcal{I} \models \varphi$ to indicate that $[\![\varphi]\!]^{\mathcal{I}} = 1$. Likewise, $\mathcal{I} \not\models \varphi$ indicates that $[\![\varphi]\!]^{\mathcal{I}} = 0$. For a formula φ, a $(\sigma,\mathbf{W})$-graph G, and a tuple $\bar{v} = (v_1,\dots,v_k) \in \big(V(G)\big)^k$, we write $G \models \varphi[\bar{v}]$ or $(G,\bar{v}) \models \varphi$ to indicate that $(G,\beta) \models \varphi$ for one (and hence every) assignment β with $\beta(x_i) = v_i$ for all $i \in [k]$. Furthermore, we set $[\![\varphi(\bar{v})]\!]^G := 1$ if $G \models \varphi[\bar{v}]$, and $[\![\varphi(\bar{v})]\!]^G := 0$ otherwise. Similarly, for an $\mathbb{S}$-term $t(\bar{x})$, we write $t^G[\bar{v}]$ to denote $[\![t]\!]^{\mathcal{I}}$. The fragments FOWA_n and FOW_1 of FOWA are defined as follows.

▶ **Definition 2.4.** *For every* $n \in \mathbb{N}$, *the set of expressions of* $\mathsf{FOWA}_n(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *is built according to the same rules as for the logic* $\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$, *with the following restrictions:*

- *rule* (5) *can only be applied if* S *is finite,*
- *rule* (6) *can only be applied if* $|\mathrm{free}(t_1) \cup \cdots \cup \mathrm{free}(t_m)| \leqslant n$.

$\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *is the restriction of* $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *where rule* (10) *cannot be applied.*

As pointed out in [5], FOW_1 can be viewed as an extension of *first-order logic with modulo-counting quantifiers*, and FOWA and FOWA_1 can be viewed as extensions of the counting logics FOC and FOC_1 of [10] and [8]. In fact, every formula in FOC can be viewed as a formula in FOWA.

Note that first-order logic FO is the restriction of FOW_1 where only rules (1), (3), and (4) can be applied. As usual, we write $(\varphi \wedge \psi)$ and $\forall x\, \varphi$ as shorthands for $\neg(\neg\varphi \vee \neg\psi)$ and $\neg\exists x\, \neg\varphi$. The *quantifier rank* $\mathrm{qr}(\xi)$ of an $\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ expression ξ is defined as the maximum nesting depth of constructs using rules (4) and (5) in order to construct ξ. The *aggregation depth* $\mathrm{d_{ag}}(\xi)$ of ξ is defined as the maximum nesting depth of term constructions using rule (10) in order to construct ξ.

▶ **Example 2.5.** Consider the following setting. $\mathbb{S}$ consists of a single ring, the ring $(\mathbb{Z}, +, \cdot)$ of integers with the natural addition and multiplication. $\mathbb{P}$ consists of a single predicate, the binary *equality predicate* $\mathsf{P}_=$ with $[\![\mathsf{P}_=]\!] = \{(i,i) : i \in \mathbb{Z}\}$. $\mathbf{W}$ consists of a single weight symbol $\mathtt{w}$, and $\mathrm{ar}(\mathtt{w}) = 2$. Furthermore, $\sigma = \{E\}$. We interpret a $(\sigma, \mathbf{W})$-graph $G = (V(G), E(G), \mathtt{w}^G)$ as a *flow network*, where $\mathtt{w}^G(u,v)$ indicates the flow through edge $\{u,v\}$ in the direction from u to v, and $\mathtt{w}^G(v,u)$ indicates the flow through edge $\{u,v\}$ in the direction from v to u.

The fact that a node x is a *source node*, i.e., all edges incident with x have weight 0 in the direction into x, can be described by the $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula $\mathit{source}(x) \coloneqq \forall z\ (0 = \mathtt{w}(z,x))$. Similarly, $\mathit{target}(y) \coloneqq \forall z\ (0 = \mathtt{w}(y,z))$ is an $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula expressing that node y is a *target node*, i.e., all edges incident with y have weight 0 in the direction outgoing from y. Furthermore, $t_{in}(z) \coloneqq \sum \mathtt{w}(u,z').(z'{=}z \wedge E(u,z'))$ is a term of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ which specifies the total flow through edges incoming into node z. Moreover, $t_{out}(z) \coloneqq \sum \mathtt{w}(z',u).(z'{=}z \wedge E(z',u))$ is a term of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ which specifies the total flow through edges going out of node z. Thus, $\psi(z) \coloneqq \mathsf{P}_=(t_{in}(z), t_{out}(z))$ is a $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula expressing that for node z, the incoming flow is equal to its outgoing flow. Finally, $\varphi(x,y) \coloneqq ((\mathit{source}(x) \wedge \mathit{target}(y)) \wedge \forall z\ ((z{=}x \vee z{=}y) \vee \psi(z)))$ is a $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula expressing the following: $G \models \varphi[s,t]$ for nodes $s, t \in V(G)$ if and only if $\mathtt{w}^G$ is a *feasible flow* for the flow network G with source and sink nodes s and t, i.e., for all vertices $v \in V(G) \setminus \{s,t\}$ the incoming flow is equal to its outgoing flow.

Locality Results

For proving the main results (2)–(5) stated in Section 1, we heavily rely on the following two locality results achieved in [5].

▶ **Theorem 2.6** (Feferman–Vaught decompositions for FOW_1 [5, Theorem 4.3])**.** *Let $k, \ell \in \mathbb{N}$, and let $\bar{x} = (x_1, \ldots, x_k)$, $\bar{y} = (y_1, \ldots, y_\ell)$ be tuples of $k + \ell$ pairwise distinct variables. For every $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula φ with free variables among $\{x_1, \ldots, x_k, y_1, \ldots, y_\ell\}$, there is a finite, non-empty set Δ of pairs (α, β) of $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formulas with $\mathrm{free}(\alpha) \subseteq \{x_1, \ldots, x_k\}$ and $\mathrm{free}(\beta) \subseteq \{y_1, \ldots, y_\ell\}$ such that the following holds. For all $(\sigma, \mathbf{W})$-graphs G and H with $V(G) \cap V(H) = \emptyset$ and all $\bar{v} \in (V(G))^k$ and $\bar{w} \in (V(H))^\ell$, we have $G \uplus H \models \varphi[\bar{v}, \bar{w}]$ if and only if there is a pair $(\alpha, \beta) \in \Delta$ with $G \models \alpha[\bar{v}]$ and $H \models \beta[\bar{w}]$.*

Furthermore, all formulas occurring in Δ have quantifier rank at most $\mathrm{qr}(\varphi)$, and they only use those $\mathsf{P} \in \mathbb{P}$ and $S \in \mathbb{S}$ that occur in φ and only those $\mathbb{S}$-terms that occur in φ or that are of the form s for an $s \in S$ with $S \in \mathbb{S}$ where S is finite and occurs in φ.

Moreover, there is an algorithm that computes Δ upon input of φ, $\bar{x}$, and $\bar{y}$.

For stating the second locality result, we need the following notation of *local formulas*. Let $r \in \mathbb{N}$. A FOWA formula $\varphi(\bar{x})$ with free variables $\bar{x} = (x_1, \ldots, x_d)$ is *r-local (around $\bar{x}$)* if for every $(\sigma, \mathbf{W})$-graph G and all $\bar{a} \in V(G)^d$, we have $G \models \varphi[\bar{a}] \iff \mathcal{N}_r^G(\bar{a}) \models \varphi[\bar{a}]$. A formula is *local* if it is r-local for some $r \in \mathbb{N}$.

▶ **Theorem 2.7** (Localisation Theorem for FOWA_1 [5, Theorem 4.7])**.** *Let $d \in \mathbb{N}$. For every formula $\varphi(x_1, \ldots, x_d)$ of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, there is an $r \in \mathbb{N}$, an extension σ' of σ with relation symbols of arity $\leqslant 1$, and an $\mathsf{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$ formula $\varphi'(x_1, \ldots, x_d)$ that is a Boolean combination of r-local formulas and statements of the form $R()$ for a 0-ary relation symbol $R \in \sigma'$ such that the following holds. There is an algorithm that, upon input of a $(\sigma, \mathbf{W})$-graph G, computes in time $|V(G)| \cdot (\deg(G))^{\mathcal{O}(1)}$ a $(\sigma', \mathbf{W})$-expansion G' of G such that, for all $\bar{v} \in V(G)^d$, it holds that $G' \models \varphi'[\bar{v}]$ if and only if $G \models \varphi[\bar{v}]$. Furthermore, r, σ', and φ' are computable from φ.*

3 FOC_2 has Unbounded VC Dimension

This section proves main result (1) stated in Section 1. Let $\sigma := \{E\}$. Let $\mathbb{S}$ consist of the integer ring $(\mathbb{Z}, +, \cdot)$, and let $\mathbf{W}$ consist of a unary weight symbol $\mathtt{one}$. We identify a graph $G = (V(G), E(G))$ with a $(\sigma, \mathbf{W})$-graph by letting $\mathtt{one}^G(v) = 1$ for all $v \in V(G)$. For a formula φ, we write $\#(y_1, \ldots, y_j).\varphi$ for the weight aggregation term $\sum p.\varphi$ for $p := \mathtt{one}(y_1) \cdots \mathtt{one}(y_j)$. Note that this term evaluates to the number of tuples $(a_1, \ldots, a_j) \in V(G)^j$ for which the formula φ is satisfied when assigning the variables $y_1, \ldots, y_j$ the vertices $a_1, \ldots, a_j$. Let $\mathbb{P}$ be the predicate collection consisting only of the *equality predicate* $\mathsf{P}_=$, where $[\![\mathsf{P}_=]\!] = \{(i, i) : i \in \mathbb{Z}\}$. The logic $\mathsf{FOC}(\mathbb{P})[\sigma]$ considered in [8] precisely corresponds to the logic $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, and $\mathsf{FOC}_n(\mathbb{P})[\sigma]$ corresponds to $\mathsf{FOWA}_n(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, for $n \in \mathbb{N}$.

▶ **Theorem 3.1.** *Let $\mathcal{T}_3$ be the class of undirected, unranked trees of height at most 3. There is an $\mathsf{FOC}_2(\mathbb{P})[\sigma]$ formula $\psi(x, y)$ such that, for every $n \in \mathbb{N}$, there exist $H \in \mathcal{T}_3$ and $W' \subseteq V(H)$ with $|W'| = n$ and $|S^{\varphi}_H(V(H)/W')| = 2^{|W'|}$. In particular, this implies that $\psi(x, y)$ has unbounded VC dimension on $\mathcal{T}_3$.*

Proof. Recall the notions introduced at the beginning of Section 1. In particular, we write $S^{\varphi}(G/W)$ as a shorthand for $S^{\varphi}_G(V(G)/W)$.

Let $\mathcal{C}_{all}$ be the class of all graphs. The proof of [8, Theorem 4.1] associates with every $G \in \mathcal{C}_{all}$ a tree $H_G \in \mathcal{T}_3$ and an injective mapping π_G from $V(G)$ to $V(H_G)$. Furthermore, the construction presented there allows associating with every $\mathsf{FO}[\sigma]$ formula $\varphi(x, y)$ an $\mathsf{FOC}_2(\mathbb{P})[\sigma]$ formula $\hat{\varphi}(x, y)$ such that the following is true for every $G \in \mathcal{C}_{all}$:

1. For all $v, w \in V(G)$, we have: $G \models \varphi[v, w] \iff H_G \models \hat{\varphi}[\pi_G(v), \pi_G(w)]$.
2. For all $v', w' \in V(H_G)$ with $v' \notin \mathrm{img}(\pi_G)$ or $w' \notin \mathrm{img}(\pi_G)$, we have: $H_G \not\models \hat{\varphi}[v', w']$.

This implies that for all $W \subseteq V(G)$ and all $v \in V(G)$ we have:
$\pi_G(\mathrm{tp}^{\varphi}_G(v/W)) = \{w' \in \pi_G(W) : H_G \models \hat{\varphi}[\pi_G(v), w']\} = \mathrm{tp}^{\hat{\varphi}}_{H_G}(\pi_G(v)/\pi_G(W))$.
Hence, $\pi_G(S^{\varphi}(G/W)) \subseteq S^{\hat{\varphi}}(H_G/\pi_G(W))$, and thus

$$|\, S^{\varphi}(G/W)\,| \;\leqslant\; |\, S^{\hat{\varphi}}(H_G/\pi_G(W))\,|. \tag{1}$$

Consider the FO formula $\varphi(x, y) := E(x, y)$. For every $n \in \mathbb{N}$, there is a graph $G \in \mathcal{C}_{all}$ and a set $W \subseteq V(G)$ with $|W| = n$ and $|S^{\varphi}(G/W)| = 2^{|W|}$. For example, we could use the graph G with $V(G) := [n] \uplus \{0, 1\}^n$, $E(G) := \big\{\{i, \bar{w}\} \;:\; i \in \mathbb{N}, \bar{w} \in \{0, 1\}^n, w_i = 1\big\}$, and $W := [n]$. Let $W' := \pi_G(W)$, and note that $|W'| = |W| = n$. From Equation (1), we obtain that $2^{|W|} = |S^{\varphi}(G/W)| \leqslant |S^{\hat{\varphi}}(H_G/W')| \leqslant 2^{|W'|} = 2^{|W|}$. Therefore, $|S^{\hat{\varphi}}(H_G/W')| = 2^{|W'|}$. Choosing $\psi(x, y)$ to be the formula $\hat{\varphi}(x, y)$ thus proves the first statement of Theorem 3.1. The second statement of the theorem is an immediate consequence of its first statement and the definition of the notion of VC dimension. ◀

4 Bound on the Number of Types

In this section, we prove the main technical tool for this paper. For that, we use the following notation. For every $k \in \mathbb{N}$, $I = \{i_1, \ldots, i_\ell\} \subseteq [k]$ with $i_1 < i_2 < \cdots < i_\ell$, and for a tuple $\bar{v} = (v_1, \ldots, v_k)$, we let $\bar{v}_I := (v_{i_1}, v_{i_2}, \ldots, v_{i_\ell})$ be the tuple obtained from $\bar{v}$ by keeping only entries at positions contained in I.

▶ **Lemma 4.1.** *There are computable functions $T\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ and $r\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \to \mathbb{N}$ such that, for every $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula $\varphi(\bar{x}, \bar{y})$, every $m \in \mathbb{N}$, every $(\sigma, \mathbf{W})$-graph G, and all $V, W \subseteq V(G)$ that are $r(\varphi)$-separated by a set of size at most m, we have $|S^{\varphi}_G(V/W)| \leqslant T(\varphi, m)$.*

Proof. Let $\varphi(\bar{x}, \bar{y}) \in \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, $k \coloneqq |\bar{x}|$, and $\ell \coloneqq |\bar{y}|$. W.l.o.g., we assume that $\mathbb{P}$, σ, $\mathbb{S}$, and $\mathbf{W}$ only contain elements that occur in φ. Using Theorem 2.7, from φ, we can compute an $r' \in \mathbb{N}$, an extension σ' of σ with relation symbols of arity $\leqslant 1$, and an $\mathsf{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$ formula $\varphi'(\bar{x}, \bar{y})$ that is a Boolean combination of r'-local formulas and statements of the form $R()$ for a 0-ary relation symbol $R \in \sigma'$ such that the following holds. For every $(\sigma, \mathbf{W})$-graph G, there is a $(\sigma', \mathbf{W})$-expansion G' of G such that for all $\bar{v} \in \big(V(G)\big)^k$ and $\bar{w} \in \big(V(G)\big)^\ell$, it holds that $G \models \varphi[\bar{v}, \bar{w}]$ if and only if $G' \models \varphi'[\bar{v}, \bar{w}]$. We set $r(\varphi) \coloneqq 2r' + 1$. Note that, for all $V, W \subseteq V(G)$, we have that $S^G_\varphi(V/W) = S^{G'}_{\varphi'}(V/W)$.

Let $m \in \mathbb{N}$. We extend σ' and $\mathbf{W}$ to be able to remove a set of vertices of size at most m from G' and encode the missing information in the remaining graph. For that, for every $i, j \in [m]$, we introduce a new 0-ary relation symbol R_i for every unary relation symbol $R \in \sigma'$, we introduce the new unary relation symbol E_i, and we introduce the new 0-ary relation symbol $E_{i,j}$. Analogously, for every $i \in [m]$, we introduce two new unary weight symbols $\mathtt{w}_{i,1}, \mathtt{w}_{i,2}$ for every binary weight symbol $\mathtt{w} \in \mathbf{W}$. In addition, for all $i, j \in [m]$, for all weight symbols $\mathtt{w} \in \mathbf{W}$, for all $s \in \mathrm{type}(\mathtt{w})$ that occur in φ' (and $\mathrm{type}(\mathtt{w})$ may be infinite) and all $s \in \mathrm{type}(\mathtt{w})$ if $\mathrm{type}(\mathtt{w})$ is finite, we add the new 0-ary relation symbol $R_{\mathtt{w},i,s}$ if $\mathtt{w}$ is a unary weight symbol, and we add the new 0-ary relation symbol $R_{\mathtt{w},i,j,s}$ if $\mathtt{w}$ is a binary weight symbol. Let σ_m and $\mathbf{W}_m$ denote the resulting signature and the resulting set of weight symbols, respectively. Note that both σ_m and $\mathbf{W}_m$ are finite.

▷ Claim 4.2. Let H be a $(\sigma', \mathbf{W})$-graph, let $z_1, \dots, z_t \in V(H)$ be pairwise distinct vertices with $t \leqslant m$, let $Z \coloneqq \{z_1, \dots, z_t\}$, and let $\psi(x'_1, \dots, x'_p) \in \mathsf{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$ for some $p \in \mathbb{N}$.

There is a $(\sigma_m, \mathbf{W}_m)$-expansion $H_{\bar{z},\psi}$ of $H \setminus Z$ such that for every mapping $f\colon [p] \to [0, t]$, there is a $\mathsf{FOW}_1(\mathbb{P})[\sigma_m, \mathbb{S}, \mathbf{W}_m]$ formula $\psi_{H,\bar{z},f}(\bar{x}'')$, where $\bar{x}''$ is obtained from $\bar{x}'$ by dropping all variables x'_i with $f(i) \neq 0$, with the following properties.

For all $\bar{v} \in (V(H))^p$, we have that $H \models \psi[\bar{v}]$ if and only if $H_{\bar{z},\psi} \models \psi_{H,\bar{z},f}[\bar{v}']$, where $\bar{v}'$ is obtained from $\bar{v}$ by dropping all elements that are contained in Z, and $f\colon [p] \to [0, t]$ maps $i \in [p]$ to $j \in [t]$ if $v_i = z_j$, and it maps $i \in [p]$ to 0 if $v_i \notin Z$.

Further, for a fixed formula ψ and a fixed mapping f, the formulas $\psi_{H,\bar{z},f}$ are structurally identical. That is, the syntax trees of all the formulas $\psi_{H,\bar{z},f}$ have the same inner nodes, and the leaf nodes that do not represent constants from rule (7) coincide. Hence, the dependence on H and $\bar{z}$ is only reflected in the use of different constants for rule (7).

Proof. Let H be a $(\sigma', \mathbf{W})$-graph, let $z_1, \dots, z_t \in V(H)$ be pairwise distinct vertices with $t \leqslant m$, let $Z \coloneqq \{z_1, \dots, z_t\}$, and let $\psi(x'_1, \dots, x'_p) \in \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ for some $p \in \mathbb{N}$.

We use the new relation symbols R_i and $E_{i,j}$ to encode whether $z_i \in R(H)$ and $\{z_i, z_j\} \in E(H)$, and we let E_i include all vertices v such that $\{z_i, v\} \in E(H)$. The relation symbols $R_{\mathtt{w},i,s}$ and $R_{\mathtt{w}',i,j,s}$ are used to encode whether $\mathtt{w}^H(z_i) = s$ and $(\mathtt{w}')^H(z_i, z_j) = s$. Finally, the unary weight symbols $\mathtt{w}_{i,1}$ and $\mathtt{w}_{i,2}$ are used to encode the weights $\mathtt{w}^H(z_i, v)$ and $\mathtt{w}^H(v, z_i)$ for all $v \in V(H) \setminus Z$. Formally, we let $H_{\bar{z},\psi}$ be the $(\sigma_m, \mathbf{W}_m)$-expansion of $H \setminus Z$ with

- $R_i(H_{\bar{z},\psi}) \coloneqq \top$ if and only if $i \in [t]$ and $z_i \in R$, for all unary $R \in \sigma'$ and $i \in [m]$,
- $E_i(H_{\bar{z},\psi}) \coloneqq N_1^H(z_i)$, for all $i \in [t]$,
- $E_i(H_{\bar{z},\psi}) \coloneqq \emptyset$, for all $i \in [m] \setminus [t]$,
- $E_{i,j}(H_{\bar{z},\psi}) \coloneqq \top$ if and only if $i, j \in [t]$ and $\{z_i, z_j\} \in E(H)$, for all $i, j \in [m]$,
- $R_{\mathtt{w},i,s}(H_{\bar{z},\psi}) \coloneqq \top$ if and only if $i \in [t]$ and $\mathtt{w}^H(z_i) = s$, for all unary $\mathtt{w} \in \mathbf{W}$ and all $s \in \mathrm{type}(\mathtt{w})$ that occur in ψ and all $s \in \mathrm{type}(\mathtt{w})$ if $\mathrm{type}(\mathtt{w})$ is finite,
- $R_{\mathtt{w},i,j,s}(H_{\bar{z},\psi}) \coloneqq \top$ if and only if $i, j \in [t]$ and $\mathtt{w}^H(z_i, z_j) = s$, for all binary $\mathtt{w} \in \mathbf{W}$ and all $s \in \mathrm{type}(\mathtt{w})$ that occur in ψ and all $s \in \mathrm{type}(\mathtt{w})$ if $\mathrm{type}(\mathtt{w})$ is finite,

- $\mathtt{w}_{i,1}^{H_{\bar{z},\psi}} : V(H_{\bar{z},\psi}) \to \mathrm{type}(\mathtt{w}), v \mapsto \mathtt{w}^H(z_i, v)$, for all binary $\mathtt{w} \in \mathbf{W}$ and $i \in [t]$,
- $\mathtt{w}_{i,2}^{H_{\bar{z},\psi}} : V(H_{\bar{z},\psi}) \to \mathrm{type}(\mathtt{w}), v \mapsto \mathtt{w}^H(v, z_i)$, for all binary $\mathtt{w} \in \mathbf{W}$ and $i \in [t]$, and
- $\mathtt{w}_{i,j}^{H_{\bar{z},\psi}} : V(H_{\bar{z},\psi}) \to \mathrm{type}(\mathtt{w}), v \mapsto 0$, for all binary $\mathtt{w} \in \mathbf{W}$, $j \in [2]$, and $i \in [m] \setminus [t]$.

Next, for every mapping $f\colon [p] \to [0,t]$, we recursively construct a $\mathsf{FOW}_1(\mathbb{P})[\sigma_m, \mathbb{S}, \mathbf{W}_m]$ formula $\psi_{H,\bar{z},f}(\bar{x}'')$, where $\bar{x}''$ is obtained from $\bar{x}'$ by dropping all variables x'_i with $f(i) \neq 0$. Intuitively, if $f(i) \neq 0$, then this indicates that the variable x'_i should be replaced by the vertex $z_{f(i)}$. For all $i \in [p]$ and $j \in [0,t]$, we let $f_{i\to j}\colon [p] \to [0,t]$ be the mapping with $f_{i\to j}(i') := f(i')$ for all $i' \neq i$ and $f_{i\to j}(i) := j$. Moreover, for $i, i' \in [p]$ and $j, j' \in [0,t]$, we analogously define $f_{i\to j, i'\to j'}\colon [p] \to [0,t]$.

(1) If ψ is of the form $x'_i{=}x'_j$, then we let $\psi_{H,\bar{z},f} := \psi$ if $f(i) = f(j) = 0$, $\psi_{H,\bar{z},f} := \top$ if $f(i) = f(j) \neq 0$, and $\psi_{H,\bar{z},f} := \bot$ else. If ψ is of the form $R()$, or ψ is of the form $R(x'_i)$ and $f(i) = 0$, or ψ is of the form $E(x'_i, x'_j)$ and $f(i) = f(j) = 0$, then we let $\psi_{H,\bar{z},f} := \psi$. If ψ is of the form $R(x'_i)$ and $f(i) \neq 0$, then we let $\psi_{H,\bar{z},f} := R_{f(i)}()$. If ψ is of the form $E(x'_i, x'_j)$ and $f(i) \neq 0$ and $f(j) = 0$, then we let $\psi_{H,\bar{z},f} := E_{f(i)}(x'_j)$. If ψ is of the form $E(x'_i, x'_j)$ and $f(i) = 0$ and $f(j) \neq 0$, then we let $\psi_{H,\bar{z},f} := E_{f(j)}(x'_i)$. If ψ is of the form $E(x'_i, x'_j)$ and $f(i), f(j) \neq 0$, then we let $\psi_{H,\bar{z},f} := E_{f(i),f(j)}()$.

(2) If ψ is of the form $\big(s{=}\mathtt{w}(x'_i)\big)$ and $f(i) = 0$, or ψ is of the form $\big(s{=}\mathtt{w}(x'_i, x'_j)\big)$ and $f(i) = f(j) = 0$, then we let $\psi_{H,\bar{z},f} := \psi$. If ψ is of the form $\big(s{=}\mathtt{w}(x'_i)\big)$ and $f(i) \neq 0$, then we let $\psi_{H,\bar{z},f} := R_{\mathtt{w},f(i),s}$. If ψ is of the form $\big(s{=}\mathtt{w}(x'_i, x'_j)\big)$ and $f(i), f(j) \neq 0$, then we let $\psi_{H,\bar{z},f} := R_{\mathtt{w},f(i),f(j),s}$. If ψ is of the form $\big(s{=}\mathtt{w}(x'_i, x'_j)\big)$ and $f(i) \neq 0$ and $f(j) = 0$, then we let $\psi_{H,\bar{z},f} := \big(s{=}\mathtt{w}_{f(i),1}(x'_j)\big)$. If ψ is of the form $\big(s{=}\mathtt{w}(x'_i, x'_j)\big)$ and $f(i) = 0$ and $f(j) \neq 0$, then we let $\psi_{H,\bar{z},f} := \big(s{=}\mathtt{w}_{f(j),2}(x'_i)\big)$.

(3) If ψ is of the form $(\psi' \vee \psi'')$, then we recursively construct $\psi'_{H,\bar{z},f}$ and $\psi''_{H,\bar{z},f}$, and we let $\psi_{H,\bar{z},f} := (\psi'_{H,\bar{z},f} \vee \psi''_{H,\bar{z},f})$. If ψ is of the form $\neg\psi'$, then we recursively construct $\psi'_{H,\bar{z},f}$, and we let $\psi_{H,\bar{z},f} := \neg\psi'_{H,\bar{z},f}$.

(4) If ψ is of the form $\exists x'_i\, \psi'$, then we recursively construct $\psi'_{H,\bar{z},f_{i\to j}}$ for all $j \in [0,t]$. We let $\psi_{H,\bar{z},f} := (\exists x'_i\, \psi'_{H,\bar{z},f_{i\to 0}} \vee \bigvee_{j=1}^{t} \psi'_{H,\bar{z},f_{i\to j}})$.

(5) If ψ is of the form $\big(s{=}\sum \mathtt{w}(x'_i).\psi'\big)$ for a unary weight symbol $\mathtt{w} \in \mathbf{W}$ of finite type $S := \mathrm{type}(\mathtt{w})$, then we recursively construct $\psi'_{H,\bar{z},f_{i\to j}}$ for all $j \in [0,t]$. We let

$$\psi_{H,\bar{z},f} := \bigvee_{\substack{s_0,s_1,\ldots,s_t \in S \\ s_0+s_1+\cdots+s_t=s}} \Big(s_0{=}\sum \mathtt{w}(x'_i).\big(\psi'_{H,\bar{z},f_{i\to 0}} \wedge \bigwedge_{j=1}^{t} (R_{\mathtt{w},j,s_j} \wedge \psi'_{H,\bar{z},f_{i\to j}})\big)\Big).$$

If ψ is of the form $\big(s{=}\sum \mathtt{w}(x'_i, x'_{i'}).\psi'\big)$ for a binary weight symbol $\mathtt{w} \in \mathbf{W}$ of finite type $S := \mathrm{type}(\mathtt{w})$, then we recursively construct $\psi'_{H,\bar{z},f_{i\to j,i'\to j'}}$ for all $j, j' \in [0,t]$. We let

$$\psi_{H,\bar{z},f} := \bigvee_{\substack{s_{0,0},s_{0,1},\ldots,s_{0,t},s_{1_0},\ldots,s_{t,t} \in S \\ s_{0_0}+s_{0,1}+\cdots+s_{t,t}=s}} \Big(s_0{=}\sum \mathtt{w}(x'_i, x'_{i'}).\big(\psi'_{H,\bar{z},f_{i\to 0,i'\to 0}} \wedge \bigwedge_{j=1}^{t}\bigwedge_{j'=1}^{t} (R_{\mathtt{w},j,j',s_{j,j'}} \wedge \psi'_{H,\bar{z},f_{i\to j,i'\to j'}})\big)\Big).$$

(6) Finally, if ψ is of the form $\mathsf{P}(t_1, \ldots, t_j)$, then $t_1, \ldots, t_j$ are terms according to rules (7)–(9), and they have at most one free variable, say x'_i. If $f(i) = 0$, then we let $\psi_{H,\bar{z},f} := \psi$. Otherwise, we let $t'_1, \ldots, t'_j$ be the terms obtained from $t_1, \ldots, t_j$ by replacing every occurrence of a term of the form $\mathtt{w}(x'_i)$ by the constant $\mathtt{w}^H(z_{f(i)})$ and every occurrence of a term of the form $\mathtt{w}(x'_i, x'_i)$ by the constant $\mathtt{w}^H(z_{f(i)}, z_{f(i)})$. We set $\psi_{H,\bar{z},f} := \mathsf{P}(t'_1, \ldots, t'_j)$.

It follows from the construction that, for all $\bar{v} \in (V(H))^p$, we have $H \models \psi[\bar{v}]$ if and only if $H_{\bar{z},\psi} \models \psi_{H,\bar{z},f}[\bar{v}']$, where $\bar{v}'$ is obtained from $\bar{v}$ by dropping all elements that are contained in Z, and $f\colon [p] \to [0,t]$ maps $i \in [p]$ to $j \in [t]$ if $v_i = z_j$, and it maps $i \in [p]$ to 0 if $v_i \notin Z$.

Moreover, for a fixed formula ψ and a fixed mapping f, the formulas $\psi_{H,\bar{z},f}$ are structurally identical. That is, the syntax trees of all the formulas $\psi_{H,\bar{z},f}$ have the same inner nodes, and even the leaf nodes that do not represent constants from some abelian group or ring (rule (7)) coincide. Hence, the dependence on H and $\bar{z}$ is only reflected in the use of different constants for rule (7). ◁

Let $V, W \subseteq V(G)$, let $z_1, \dots, z_t \in V(G)$ be pairwise distinct vertices with $t \leqslant m$ such that V and W are $r(\varphi)$-separated in G (and thus also in G') by the set $Z \coloneqq \{z_1, \dots, z_t\}$, and let $\bar{z} \coloneqq (z_1, \dots, z_t)$. W.l.o.g., we may assume that every vertex from Z is contained in some path from V to W in G of length at most $r(\varphi) = 2r' + 1$, so $Z \subseteq V\big(\mathcal{N}_{r'}^{G'}(V \cup W)\big)$.

By applying Claim 4.2 to $H \coloneqq \mathcal{N}_{r'}^{G'}(V \cup W)$, $z_1, \dots, z_t$, and φ', we obtain a $(\sigma_m, \mathbf{W}_m)$-expansion $H_{\bar{z},\varphi'}$ of $H \setminus Z$ and, for every mapping $f\colon [k+\ell] \to [0,t]$, a $\mathsf{FOW}_1(\mathbb{P})[\sigma_m, \mathbb{S}, \mathbf{W}_m]$ formula $\varphi'_{H,\bar{z},f}$. Since V is $(2r'+1)$-separated from W by Z, there is no path from $V \setminus Z$ to $W \setminus Z$ in $H \setminus Z = \mathcal{N}_{r'}^{G'}(V \cup W) \setminus Z$. Hence, there are $(\sigma_m, \mathbf{W}_m)$-graphs H_V and H_W such that $V \setminus Z \subseteq V(H_V)$, $W \setminus Z \subseteq V(H_W)$, and $H_{\bar{z},\varphi'} = H_V \uplus H_W$.

Let $\bar{v} \in V^k$ and $\bar{w} \in W^\ell$. We have $G \models \varphi[\bar{v}, \bar{w}]$ if and only if $G' \models \varphi'[\bar{v}, \bar{w}]$. Moreover, since φ' is a Boolean combination of r'-local formulas and statements of the form $R()$ for a 0-ary relation symbol $R \in \sigma'$, we have that $G' \models \varphi'[\bar{v}, \bar{w}]$ if and only if $\mathcal{N}_{r'}^{G'}(\bar{v}\bar{w}) \models \varphi'[\bar{v}, \bar{w}]$ if and only if $\mathcal{N}_{r'}^{G'}(V \cup W) \models \varphi'[\bar{v}, \bar{w}]$. Furthermore, by Claim 4.2, it holds that $\mathcal{N}_{r'}^{G'}(V \cup W) \models \varphi'[\bar{v}, \bar{w}]$ if and only if $H_{\bar{z},\varphi'} \models \varphi'_{H,\bar{z},f}[\bar{v}', \bar{w}']$, where $\bar{v}'$ and $\bar{w}'$ are obtained from $\bar{v}$ and $\bar{w}$, respectively, by dropping all entries that are contained in Z, and $f\colon [k+\ell] \to [0,t]$ is defined by $f(i) \coloneqq j$ if $i \leqslant k$ and $v_i = z_j$ or $i > k$ and $w_{i-k} = z_j$, and $f(i) \coloneqq 0$ if $i \leqslant k$ and $v_i \notin Z$ or $i > k$ and $w_{i-k} \notin Z$. Let $\bar{x}'$ and $\bar{y}'$ be the tuples of variables obtained analogously from $\bar{x}$ and $\bar{y}$, respectively.

Using Theorem 2.6, we obtain a Feferman–Vaught decomposition $\Delta_{\varphi'_{H,\bar{z},f}}$ of $\varphi'_{H,\bar{z},f}(\bar{x}', \bar{y}')$ w.r.t. $(\bar{x}'; \bar{y}')$, that is, a set of pairs $\big(\alpha(\bar{x}'), \beta(\bar{y}')\big)$ of $\mathsf{FOW}_1(\mathbb{P})[\sigma_m, \mathbb{S}, \mathbf{W}_m]$ formulas such that $H_{\bar{z},\varphi'} \models \varphi'_{H,\bar{z},f}[\bar{v}', \bar{w}']$ if and only if there is a pair $\big(\alpha(\bar{x}'), \beta(\bar{y}')\big)$ in $\Delta_{\varphi'_{H,\bar{z},f}}$ such that $H_V \models \alpha[\bar{v}']$ and $H_W \models \beta[\bar{w}']$. Since the structure of $\varphi'_{H,\bar{z},f}$ is independent of H and $\bar{z}$, and only the used constants might differ, it is easy to see from the proof of Theorem 2.6 (see [5] for details) that the size of $\Delta_{\varphi'_{H,\bar{z},f}}$ only depends on φ and f, and that it is independent of H and $\bar{z}$. Furthermore, the number of mappings $f\colon [k+\ell] \to [0,t]$ only depends on φ and m (recall that $t \leqslant m$), so we can let $T'(\varphi, m)\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ be an upper bound on the number of pairs in the decomposition $\Delta_{\varphi'_{H,\bar{z},f}}$ for all H, $\bar{z}$, and f.

All in all, we have $G \models \varphi[\bar{v}, \bar{w}]$ if and only if there is a pair $\big(\alpha(\bar{x}'), \beta(\bar{y}')\big)$ in $\Delta_{\varphi'_{H,\bar{z},f}}$ such that $H_V \models \alpha[\bar{v}']$ and $H_W \models \beta[\bar{w}']$. Hence, for every $\bar{v} \in V^k$, $\mathrm{tp}_G^\varphi(\bar{v}/W)$ only depends on

- which vertices of $\bar{v}$ are contained in Z and
- which formulas α of pairs (α, β) in any of the $\Delta_{\varphi'_{H,\bar{z},f}}$ are satisfied by $\bar{v}'$, where $\bar{v}'$ is obtained from $\bar{v}$ by dropping all entries that are contained in Z, and f ranges over all mappings $f\colon [k+\ell] \to [0,t]$ with, for all $i \in [k]$, $f(i) = j$ if $v_i = z_j$, and $f(i) = 0$ if $v_i \notin Z$.

Since the number of possibilities for both can be bounded in terms of φ and m, there is a function $T\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ such that $|S_G^\varphi(V/W)| = \big|\{\mathrm{tp}_G^\varphi(\bar{v}/W) : \bar{v} \in V^k\}\big| \leqslant T(\varphi, m)$. This is the statement of Lemma 4.1. ◀

5 VC Density and VC Dimension

In this section, we prove Results (2)–(4) stated in Section 1. Our main result of this section is the following.

▶ **Theorem 5.1.** *Let $\mathcal{C}$ be a nowhere dense class of $(\sigma, \mathbf{W})$-graphs, and let $\varphi(\bar{x}, \bar{y})$ be a $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula. For every $\varepsilon > 0$, there exists a constant $c \in \mathbb{N}$ such that for every $G \in \mathcal{C}$ and every non-empty $W \subseteq V(G)$, we have $|S^{\varphi}(G/W)| \leqslant c \cdot |W|^{|\bar{x}|+\varepsilon}$.*

As discussed in the introduction, this immediately implies the following bound on the VC density of FOWA_1 formulas.

▶ **Corollary 5.2.** *Let $\mathcal{C}$ be a nowhere dense class of $(\sigma, \mathbf{W})$-graphs, and let $\varphi(\bar{x}, \bar{y})$ be a $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula. The VC density of $\varphi(\bar{x}, \bar{y})$ on $\mathcal{C}$ is at most $|\bar{x}|$.*

Moreover, this implies that the VC dimension of FOWA_1 formulas on nowhere dense classes is bounded.

▶ **Corollary 5.3.** *Let $\mathcal{C}$ be a nowhere dense class of $(\sigma, \mathbf{W})$-graphs, and let $\varphi(\bar{x}, \bar{y})$ be a $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ formula. It holds that $\varphi(\bar{x}, \bar{y})$ has bounded VC dimension on $\mathcal{C}$.*

Proof. As described in the introduction, Corollary 5.2 already implies Corollary 5.3, since the VC dimension is finite if and only if the VC density is finite (see, e. g., [2]). However, since we find it short and instructive, we also give a proof of Corollary 5.3 based on Theorem 5.1.

Let $k := |\bar{x}|$, $\ell := |\bar{y}|$, let $\varepsilon > 0$, and let $c \in \mathbb{N}$ be the constant from Theorem 5.1 applied to $\mathcal{C}$, $\varphi(\bar{x}, \bar{y})$, and ε. Moreover, let $m_0 \in \mathbb{N}$ be such that $c \cdot (\ell m)^{k+\varepsilon} < 2^m$ for all $m \geqslant m_0$.

Let $G \in \mathcal{C}$ and $Y \subseteq \big(V(G)\big)^{\ell}$ such that $|Y| =: m \geqslant m_0$. Let $W \subseteq V(G)$ be the set of vertices appearing in any tuple in Y. We have $|W| \leqslant \ell \cdot |Y| = \ell m$. Moreover, we have $\{Y \cap F : F \in S^{\varphi}(G/V(G))\} \subseteq S^{\varphi}(G/W)$. Hence, by Theorem 5.1, we have $|\{Y \cap F : F \in S^{\varphi}(G/V(G))\}| \leqslant |S^{\varphi}(G/W)| \leqslant c \cdot (\ell m)^{k+\varepsilon} < 2^m$. This shows that $\{Y \cap F : F \in S^{\varphi}(G/V(G))\} \neq 2^Y$, so Y is not shattered by $S^{\varphi}(G/V(G))$. Thus, the VC dimension of $S^{\varphi}(G/V(G))$ is less than m_0. Since m_0 does not depend on G, this proves that $\varphi(\bar{x}, \bar{y})$ has bounded VC dimension on $\mathcal{C}$. ◀

For the proof of Theorem 5.1, we rely on the following lemma on the neighbourhood complexity in nowhere dense graph classes. Let G be a $(\sigma, \mathbf{W})$-graph, and let $X \subseteq V(G)$. For vertices $v \in X$ and $w \in V(G)$, a path P from v to w in G is called *X-avoiding* if all vertices on the path except for v are not contained in X. For an $r \in \mathbb{N}$ and $w \in V(G)$, the *r-projection of w on X*, denoted by $M_r^G(w, X)$, is the set of all vertices $v \in X$ that are connected to w by an X-avoiding path of length at most r.

▶ **Lemma 5.4** ([6, Lemmas 21 and 22])**.** *Let $\mathcal{C}$ be a nowhere dense class of graphs. There is a function $f_{\mathrm{cl}} \colon \mathbb{N} \times \mathbb{Q}_{>0} \to \mathbb{N}$ and an algorithm[1] that, given a graph $G \in \mathcal{C}$, $X \subseteq V(G)$, $r \in \mathbb{N}$, and $\delta \in \mathbb{Q}_{>0}$, computes a set $\mathrm{cl}_{r,\delta}(X)$, called the r-*closure of X w.r.t. δ, *with the following properties.*

1. $X \subseteq \mathrm{cl}_{r,\delta}(X) \subseteq V(G)$,
2. $|\mathrm{cl}_{r,\delta}(X)| \leqslant f_{\mathrm{cl}}(r, \delta) \cdot |X|^{1+\delta}$, *and*
3. $\big|M_r^G\big(u, \mathrm{cl}_{r,\delta}(X)\big)\big| \leqslant f_{\mathrm{cl}}(r, \delta) \cdot |X|^{\delta}$ *for all* $u \in V(G) \setminus \mathrm{cl}_{r,\delta}(X)$.

Moreover, for all $X \subseteq V(G)$, it holds that

4. $\big|\big\{M_r^G(u, X) \ : \ u \in V(G)\big\}\big| \leqslant f_{\mathrm{cl}}(r, \delta) \cdot |X|^{1+\delta}$.

[1] In [6], the authors even show that this can be computed by a polynomial-time algorithm. However, running-time bounds are not relevant for our purposes.

We can now prove Theorem 5.1.

Proof of Theorem 5.1. The proof is similar to the proof of the analogous result for first-order logic in [14], using Lemma 4.1 instead of the corresponding result for FO.

Let $\mathcal{C}$ be a nowhere dense class of $(\sigma, \mathbf{W})$-graphs, let $\varphi(\bar{x}, \bar{y})$ be a FOWA_1 formula, and let $\varepsilon > 0$. Let $k \coloneqq |\bar{x}|$, $\ell \coloneqq |\bar{y}|$, let $r\colon \mathsf{FOWA}_1 \to \mathbb{N}$ and $T\colon \mathsf{FOWA}_1 \times \mathbb{N} \to \mathbb{N}$ be the functions from Lemma 4.1, let $t\colon \mathbb{N} \to \mathbb{N}$ be the function from Definition 2.1, and let $r \coloneqq r(\varphi)$ and $t \coloneqq t(36r)$. We have that no graph $G \in \mathcal{C}$ contains K_t as a depth-$36r$ minor.

By Theorem 2.2, there is a number $s \in \mathbb{N}$ and a polynomial $N\colon \mathbb{N} \to \mathbb{N}$ such that, for every graph $G \in \mathcal{C}$, every $m \in \mathbb{N}$, and every set $X \subseteq (V(G))^k$ with $|X| \geqslant N(m)$, there are sets $S \subseteq V(G)$ and $Y \subseteq X$ with $|S| \leqslant s$ and $|Y| \geqslant m$ such that all distinct $\bar{v}, \bar{v}' \in Y$ are $2r$-separated by S in G. Let d be the degree of N.

Let $G \in \mathcal{C}$, and let $W \subseteq V(G)$ be a non-empty set of vertices. We set $\delta \coloneqq \frac{\varepsilon}{4k+4d}$, and we let $W' \coloneqq \mathrm{cl}_{r,\delta}(W)$ be the r-closure of W w.r.t. δ, obtained via Lemma 5.4. We shall prove that

$$|S^{\varphi}(G/W')| \in \mathcal{O}_{\varepsilon,\varphi}\big(|W'|^{k+\varepsilon'}\big) \qquad \text{for } \varepsilon' \coloneqq \varepsilon/2 > 0, \tag{$\star$}$$

where $\mathcal{O}_{\varepsilon,\varphi}(\cdot)$ omits factors depending only on ε and φ. Since $W \subseteq W'$, we have $|S^{\varphi}(G/W)| \leqslant |S^{\varphi}(G/W')|$. Moreover, by Lemma 5.4, we have $|W'| = |\mathrm{cl}_{r,\delta}(W)| \leqslant f_{\mathrm{cl}}(r,\delta) \cdot |W|^{1+\delta}$, and we have $(1+\delta)(k+\varepsilon') = (1+\delta)(k+\varepsilon/2) \leqslant k + \varepsilon$ by the choice of δ, so

$$|S^{\varphi}(G/W)| \in \mathcal{O}_{\varepsilon,\varphi}\Big(\big(f_{\mathrm{cl}}(r,\delta) \cdot |W|^{1+\delta}\big)^{k+\varepsilon'}\Big) \subseteq \mathcal{O}_{\varepsilon,\varphi}\big(|W|^{k+\varepsilon}\big),$$

which is the statement of Theorem 5.1.

It remains to prove ($\star$). Recall that $S^{\varphi}(G/W') = \big\{\mathrm{tp}^{\varphi}_G(\bar{v}/W') \;:\; \bar{v} \in \big(V(G)\big)^k\big\}$. We partition the tuples $\bar{v} = (v_1, \ldots, v_k) \in \big(V(G)\big)^k$ based on their projection $M^G_r(\bar{v}, W') \coloneqq \bigcup_{i=1}^{k} M_r(v_i, W')$ into sets $V_1, \ldots, V_p$. That is, two tuples $\bar{v}, \bar{v}' \in \big(V(G)\big)^k$ are contained in the same set V_j for some $j \in [p]$ if and only if $M^G_r(\bar{v}, W') = M^G_r(\bar{v}', W')$. By Item 4 of Lemma 5.4, there are at most $f_{\mathrm{cl}}(r,\delta) \cdot |W'|^{1+\delta}$ different projections of vertices in $V(G)$ on W', so we have $p \in \mathcal{O}_{\varepsilon,\varphi}\big(|W'|^{(1+\delta)k}\big)$. Hence, to prove ($\star$), it suffices to show that

$$\big|\big\{\mathrm{tp}^{\varphi}_G(\bar{v}/W') \;:\; \bar{v} \in V_j\big\}\big| \in \mathcal{O}_{\varepsilon,\varphi}\big(|W'|^{\varepsilon''}\big) \qquad \text{for } \varepsilon'' \coloneqq \varepsilon' - k\delta > 0, \tag{$\star\star$}$$

for all $j \in [p]$, since then $|S^{\varphi}(G/W')| \in \mathcal{O}_{\varepsilon,\varphi}\big(|W'|^{(1+\delta)k}\,|W'|^{\varepsilon'-k\delta}\big) = \mathcal{O}_{\varepsilon,\varphi}\big(|W'|^{k+\varepsilon'}\big)$.

Let $j \in [p]$, and let $X \coloneqq M^G_r(\bar{v}, W')$ be the r-projection of $\bar{v}$ on W' for any (and, due to the definition of V_j, for all) $\bar{v} \in V_j$. By Item 3 of Lemma 5.4, we have $|X| \leqslant k \cdot f_{\mathrm{cl}}(r,\delta) \cdot |W|^{\delta} \in \mathcal{O}_{\varepsilon,\varphi}\big(|W|^{\delta}\big)$.

Let V'_j be a maximal subset of V_j such that all pairwise distinct tuples $\bar{v}, \bar{v}'$ from V'_j have different types $\mathrm{tp}^{\varphi}_G(\bar{v}/W') \neq \mathrm{tp}^{\varphi}_G(\bar{v}'/W')$. Note that $\big|\big\{\mathrm{tp}^{\varphi}_G(\bar{v}/W') \;:\; \bar{v} \in V_j\big\}\big| = \big|V'_j\big|$. Now let $m \in \mathbb{N}$ be the maximum number with $\big|V'_j\big| \geqslant N(m)$. Then $\big|V'_j\big| < N(m+1) \in \mathcal{O}_{\varepsilon,\varphi}(m^d)$.

By Theorem 2.2, as described above, there are sets $S \subseteq V(G)$ and $Y \subseteq V'_j$ with $|S| \leqslant s$ and $|Y| \geqslant m$ such that all distinct $\bar{v}, \bar{v}' \in Y$ are $2r$-separated by S in G.

We partition Y into two sets $Y_1 \uplus Y_2$, where Y_1 contains all tuples that are r-separated by S from W', and Y_2 contains the remaining tuples. By Lemma 4.1, since all tuples in Y_1 are r-separated by S from W', and all tuples in Y_1 have distinct types, we know that $|Y_1| \leqslant T(\varphi, s) \in \mathcal{O}_{\varepsilon,\varphi}(1)$. Moreover, for every tuple $\bar{v} \in Y_2$, there is a vertex $w \in W'$ such that $\bar{v}$ and w are not r-separated by S in G. Note that we can choose w to be contained in X. Moreover, since all tuples in Y_2 are mutually $2r$-separated by S in G, we know that

for two distinct tuples $\bar{v}, \bar{v}' \in Y_2$, the vertices in C connected to them by paths of length at most r avoiding S must also be distinct. This shows that $|Y_2| \leqslant |X|$. Combined, we obtain that $|Y| \in \mathcal{O}_{\varepsilon,\delta}(|X|)$. Furthermore, since $|Y| \geqslant m$, we have

$$|V'_j| \in \mathcal{O}_{\varepsilon,\varphi}(m^d) \subseteq \mathcal{O}_{\varepsilon,\varphi}(|Y|^d) \subseteq \mathcal{O}_{\varepsilon,\varphi}(|X|^d) \subseteq \mathcal{O}_{\varepsilon,\varphi}(|W'|^{d\delta}) \subseteq \mathcal{O}_{\varepsilon,\varphi}(|W'|^{\varepsilon''}),$$

where the last inclusion holds because $\varepsilon'' = \varepsilon/2 - k\delta \leqslant \varepsilon/4 \leqslant d\delta$ by the choice of δ. This proves $(\star\star)$, which, as discussed above, implies the statement of Theorem 5.1. ◀

6 Stability

In this section, we provide the following bound on the ladder index of FOC_1 formulas and FOWA_1 formulas on nowhere dense classes of weighted graphs. Based on this, we prove Result (5) stated in Section 1.

▶ **Theorem 6.1.** *There are computable functions $f\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ and $g\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}] \to \mathbb{N}$ such that, for every $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ formula φ, for every $t \in \mathbb{N}$, and for every $(\sigma,\mathbf{W})$-graph G excluding K_t as a depth-$g(\varphi)$ minor, the ladder index of φ in G is at most $f(\varphi,t)$.*

Proof. The proof is similar to the proof of the analogous statement in [14] for first-order formulas. Let $r\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}] \to \mathbb{N}$ and $T\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ be the functions from Lemma 4.1. We set $g\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}] \to \mathbb{N}, \varphi \mapsto 18r(\varphi)$.

Let $\varphi(\bar{x},\bar{y})$ be a $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ formula, let $t \in \mathbb{N}$, and let $\mathcal{C}$ be the class of $(\sigma,\mathbf{W})$-graphs excluding K_t as a depth-$g(\varphi)$ minor. Let $d := |\bar{x}| + |\bar{y}|$, and let $s \in \mathbb{N}$ be the number and $N\colon \mathbb{N} \to \mathbb{N}$ be the polynomial computed from $r(\varphi)$, t, and d using Theorem 2.2. Moreover, let $L := f(\varphi,t) := N\big(2T(\varphi,s)+1\big)$. (Note that N and s can be computed from φ and t.) We show that every φ-ladder in every graph $G \in \mathcal{C}$ has length less than L.

Towards a contradiction, suppose there are a graph $G \in \mathcal{C}$ and tuples $\bar{v}_1,\dots,\bar{v}_L \in \big(V(G)\big)^{|x|}$ and $\bar{w}_1,\dots,\bar{w}_L \in \big(V(G)\big)^{|y|}$ that form a φ-ladder in G, that is, $G \models \varphi[\bar{v}_i,\bar{w}_j]$ if and only if $i \leqslant j$. In particular, the tuples $\bar{v}_1,\dots,\bar{v}_L$ are pairwise distinct, and the same holds for the tuples $\bar{w}_1,\dots,\bar{w}_L$. Let $X := \{\bar{v}_i\bar{w}_i : i \in [L]\} \subseteq \big(V(G)\big)^d$. By Theorem 2.2, for $m := 2T(\varphi,s)+1$, since $|X| \geqslant N(m)$, there are sets $S \subseteq V(G)$ and $Y \subseteq X$ with $|S| \leqslant s$ and $|Y| \geqslant m$ such that all distinct $\bar{u},\bar{u}' \in Y$ are $r(\varphi)$-separated by S in G. Let $I := \{i \in [\ell] : \bar{v}_i\bar{w}_i \in Y\}$. Let I_1, I_2 be an alternating partition of I, that is, for all successive $i,j \in I_1$, there is exactly one $k \in I_2$ with $i < k < j$. Note that $|I_1| \geqslant T(\varphi,s)+1$. Let $V \subseteq V(G)$ be the set of vertices appearing in a tuple $\bar{v}_i\bar{w}_i$ with $i \in I_1$, and let $W \subseteq V(G)$ be the set of vertices appearing in a tuple $\bar{v}_i\bar{w}_i$ with $i \in I_2$. Since all distinct $\bar{u},\bar{u}' \in Y$ are $r(\varphi)$-separated by S in G, it also holds that the sets V and W are $r(\varphi)$-separated by S in G.

Now we can apply Lemma 4.1 to V and W, and we obtain $|S^{\varphi}_G(V/W)| \leqslant T(\varphi,s) < |I_1|$. Hence, there are two indices $i,j \in I_1$ with $i < j$ such that $\mathrm{tp}^{\varphi}_G(\bar{v}_i/W) = \mathrm{tp}^{\varphi}_G(\bar{v}_j/W)$. Let $k \in I_2$ with $i < k < j$. Then $\bar{w}_k \in \mathrm{tp}^{\varphi}_G(\bar{v}_i/W)$ if and only if $\bar{w}_k \in \mathrm{tp}^{\varphi}_G(\bar{v}_j/W)$, so $G \models \varphi[\bar{v}_i,\bar{w}_k]$ if and only if $G \models \varphi[\bar{v}_j,\bar{w}_k]$. However, this contradicts $\bar{v}_1,\dots,\bar{v}_\ell$ and $\bar{w}_1,\dots,\bar{w}_\ell$ being a φ-ladder, because we need to have $G \models \varphi[\bar{v}_i,\bar{w}_k]$ (since $i < k$) and $G \not\models \varphi[\bar{v}_j,\bar{w}_k]$ (since $j > k$). This shows that there is no φ-ladder in G of size at least $L = f(\varphi,t)$, so the ladder index of φ in G is at most $f(\varphi,t)$. ◀

We call a class $\mathcal{C}$ of weighted graphs FOWA_1-*stable* (FOC_1-*stable*) if the ladder index of every FOWA_1 (FOC_1) formula φ in every weighted graph from $\mathcal{C}$ is bounded by a constant depending only on φ and $\mathcal{C}$.

▶ **Corollary 6.2.** *Every nowhere dense class of weighted graphs is* FOC_1*-stable and* FOWA_1*-stable.*

Proof. Let $\mathcal{C}$ be a nowhere dense class of $(\sigma, \mathbf{W})$-graphs, let $\varphi(\bar{x}, \bar{y})$ be a formula in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, and let $k := |\bar{x}|$ and $\ell := |\bar{y}|$. By Definition 2.1, there is a function $t\colon \mathbb{N} \to \mathbb{N}$ such that for all $r \in \mathbb{N}$ and $G \in \mathcal{C}$, it holds that G does not contain $K_{t(r)}$ as a depth-r minor.

Let $f\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \times \mathbb{N} \to \mathbb{N}$ and $g\colon \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}] \to \mathbb{N}$ be the functions from Theorem 6.1. For all $G \in \mathcal{C}$, we have that G does not contain $K_{t(g(\varphi))}$ as a depth-$g(\varphi)$ minor. Thus, by Theorem 6.1, for every $G \in \mathcal{C}$, the ladder index of φ in G is at most $L := f\big(\varphi, t(g(\varphi))\big)$, which only depends on φ and $\mathcal{C}$. ◀

7 Final Remarks

In this paper, we have presented upper bounds on the VC dimension and the ladder index as well as optimal bounds on the VC density of formulas in the first-order logic with counting FOC_1 and the first-order logic with weight aggregation FOWA_1 on nowhere dense classes of vertex- and edge-weighted graphs. This lifts results of Adler and Adler [1] and results of Pilipczuk, Siebertz, and Toruńczyk [14] from first-order logic to substantially more expressive logics.

In [4], van Bergerem, Grohe, and Ritzert combined the result by Adler and Adler with the fixed-parameter tractable (fpt) model-checking result for FO on nowhere dense graph classes [7] to prove learnability results for FO on nowhere dense graph classes in the Probably Approximately Correct (PAC) learning framework. We remark that, by combining our results on the VC dimension for FOC_1 formulas with the fpt model-checking result for FOC_1 by Grohe and Schweikardt [8], we also obtain fpt PAC learnability for FOC_1-definable concepts over nowhere dense graph classes. We are currently working on lifting these model-checking and learnability results from FOC_1 to FOWA_1.

References

1 Hans Adler and Isolde Adler. Interpreting nowhere dense graph classes as a classical notion of model theory. *Eur. J. Comb.*, 36:322–330, 2014. doi:10.1016/j.ejc.2013.06.048.

2 Matthias Aschenbrenner, Alf Dolich, Deirdre Haskell, Dugald Macpherson, and Sergei Starchenko. Vapnik–Chervonenkis density in some theories without the independence property, I. *Transactions of the American Mathematical Society*, 368(8):5889–5949, August 2016. doi:10.1090/tran/6659.

3 Steffen van Bergerem. Learning concepts definable in first-order logic with counting. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24–27, 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785811.

4 Steffen van Bergerem, Martin Grohe, and Martin Ritzert. On the parameterized complexity of learning first-order logic. In *PODS 2022: International Conference on Management of Data, Philadelphia, PA, USA, June 12–17, 2022*, pages 337–346. ACM, 2022. doi:10.1145/3517804.3524151.

5 Steffen van Bergerem and Nicole Schweikardt. Learning concepts described by weight aggregation logic. In *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, Ljubljana, Slovenia (Virtual Conference), January 25–28, 2021*, volume 183 of *LIPIcs*, pages 10:1–10:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CSL.2021.10.

6 Kord Eickmeyer, Archontia C. Giannopoulou, Stephan Kreutzer, O-joung Kwon, Michał Pilipczuk, Roman Rabinovich, and Sebastian Siebertz. Neighborhood complexity and kernelization for nowhere dense classes of graphs. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10–14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 63:1–63:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ICALP.2017.63.

7 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. *J. ACM*, 64(3):17:1–17:32, 2017. doi:10.1145/3051095.

8 Martin Grohe and Nicole Schweikardt. First-order query evaluation with cardinality conditions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2018, Houston, TX, USA, June 10–15, 2018*, pages 253–266. ACM, 2018. doi:10.1145/3196959.3196970.

9 Martin Grohe and György Turán. Learnability and definability in trees and similar structures. *Theory Comput. Syst.*, 37(1):193–220, 2004. doi:10.1007/s00224-003-1112-8.

10 Dietrich Kuske and Nicole Schweikardt. First-order logic with counting. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavík, Iceland, June 20–23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005133.

11 Jaroslav Nešetřil and Patrice Ossona de Mendez. First order properties on nowhere dense structures. *J. Symb. Log.*, 75(3):868–887, 2010. doi:10.2178/jsl/1278682204.

12 Jaroslav Nešetřil and Patrice Ossona de Mendez. On nowhere dense graphs. *Eur. J. Comb.*, 32(4):600–617, 2011. doi:10.1016/j.ejc.2011.01.006.

13 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

14 Michał Pilipczuk, Sebastian Siebertz, and Szymon Toruńczyk. On the number of types in sparse graphs. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09–12, 2018*, pages 799–808. ACM, 2018. doi:10.1145/3209108.3209178.

15 Klaus-Peter Podewski and Martin Ziegler. Stable graphs. *Fundamenta Mathematicae*, 100(2):101–107, 1978. URL: http://eudml.org/doc/210953.

16 Norbert Sauer. On the density of families of sets. *J. Comb. Theory A*, 13(1):145–147, 1972. doi:10.1016/0097-3165(72)90019-2.

17 Saharon Shelah. A combinatorial problem: stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, 1972. doi:10.2140/pjm.1972.41.247.

18 Katrin Tent and Martin Ziegler. *A Course in Model Theory*. Lecture Notes in Logic. Cambridge University Press, 2012. doi:10.1017/CBO9781139015417.

19 Vladimir Naumovich Vapnik and Alexey Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971. doi:10.1137/1116025.

Undefinability of Approximation of 2-To-2 Games

Anuj Dawar
Department of Computer Science and Technology, University of Cambridge, UK

Bálint Molnár
Department of Computer Science and Technology, University of Cambridge, UK

Abstract

Recent work by Atserias and Dawar [6] and Tucker-Foltz [26] has established undefinability results in fixed-point logic with counting (FPC) corresponding to many classical complexity results from the hardness of approximation. In this line of work, NP-hardness results are turned into unconditional FPC undefinability results. We extend this work by showing the FPC undefinability of any constant factor approximation of weighted 2-to-2 games, based on the NP-hardness results of Khot, Minzer and Safra. Our result shows that the completely satisfiable 2-to-2 games are not FPC-separable from those that are not ϵ-satisfiable, for arbitrarily small ϵ. The perfect completeness of our inseparability is an improvement on the complexity result, as the NP-hardness of such a separation is still only conjectured. This perfect completeness enables us to show the FPC undefinability of other problems whose NP-hardness is conjectured. In particular, we are able to show that no FPC formula can separate the 3-colourable graphs from those that are not t-colourable, for any constant t.

2012 ACM Subject Classification Theory of computation → Finite Model Theory; Theory of computation → Complexity theory and logic; Theory of computation → Problems, reductions and completeness

Keywords and phrases Hardness of Approximation, Unique Games, Descriptive Complexity, Fixed-Point Logic with Counting

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.16

Funding *Anuj Dawar*: Funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee: grant number EP/X028259/1.

1 Introduction

The study of the hardness of approximation of NP-optimization problems began in earnest with the PCP theorem in the 1990s. This theorem showed that for many problems (such as MAX 3SAT), where there are polynomial-time algorithms that can approximate the optimum solution within a constant factor, there is nonetheless a constant c such that no efficient algorithm can approximate the optimum value within a factor c unless P = NP. Indeed, Håstad [17] established tight bounds for MAX 3SAT: there is a trivial algorithm that achieves an $\frac{8}{7}$ approximation, but none that achieves an $\frac{8}{7} - \epsilon$ approximation for any ϵ, unless P = NP. Such tight bounds are known for many NP-optimization problems, while for others there is a gap in the approximation ratio between the best known algorithm and the strongest known lower bound. An important problem in the latter category is the *minimum vertex cover* problem, where the best known polynomial-time algorithms yield an approximation ratio of 2, while the strongest proved lower bound is $\sqrt{2}$.

Perhaps the most important open question in the field of the hardness of approximation is the *unique games conjecture* of Khot. This states that for any $\epsilon, \delta > 0$, there is a set of labels Σ such that it is NP-hard to separate the $(1 - \epsilon)$-satisfiable instances of Σ-unique games (the precise definitions follow below) from those that are not even δ-satisfiable. The strongest result obtained so far in this direction shows that there is a Σ for which it is NP-hard to separate the $(\frac{1}{2} - \epsilon)$-satisfiable instances from the δ-unsatisfiable ones. This result is a consequence of the 2-to-2 theorem due to Khot, Minzer and Safra [20, 11, 21].

The hardness of approximation has also been studied in recent years in the context of logical definability. In particular, Atserias and Dawar [6] showed that many of the NP-hardness results can be recast as *unconditional* undefinability results in *fixed-point logic with counting* (FPC). For example, there is an FPC formula which yields an $\frac{8}{7}$ approximation of the value of a MAX 3SAT instance and there is *provably* no formula that yields an $\frac{8}{7} - \epsilon$ approximation for any $\epsilon > 0$. Recall that FPC is a logic whose expressive power is contained within the complexity class P and which has been characterized as a natural *symmetric* fragment of that class [1]. Tucker-Foltz [26] established the first definability gap in FPC of unique games, by showing that no formula can distinguish the $\frac{1}{2}$-satisfiable instances from those that are not $\frac{1}{3} + \delta$-satisfiable and also showed that no constant factor approximation is FPC definable.

In the present paper, we consider the FPC definability of 2-to-2 games. The hardness of approximating the optimum value of such games was established through a series of results by Khot, Minzer and Safra [20, 25, 11]. At the core of their proof is a reduction from the problem MAX 3XOR of maximizing the number of satisfied clauses in a 3XOR instance. We show that the reductions used can be formulated, with some modification, as first-order definable reductions. As a consequence, we obtain the result that the completely satisfiable instances of 2-to-2 games cannot be separated by an FPC formula from those that are no more than δ-satisfiable. This $(1, \delta)$ separation is stronger (in terms of approximation ratios) than the known $(1 - \epsilon, \delta)$ NP-hardness result due to the fact that the FPC undefinability of approximating MAX 3XOR was proved with *perfect completeness* in [6]. A corollary of our result is the FPC undefinability of a $(\frac{1}{2}, \delta)$ separation for (a weighted version of) unique games. This improves, again in terms of the approximation ratios, the gap obtained by Tucker-Foltz, though it should be noted that the latter gap is for *unweighted* games.

A more striking consequence of our result is that no FPC sentence can separate the class of 3-colourable graphs from those that are not even t-colourable for any constant $t \geq 3$. The NP-hardness of such a separation has only been proved for t at most 5, though it is conjectured for larger values. Indeed, this is a central open problem in the rapidly growing study of *promise constraint satisfaction problems* (PCSP, see [7]).

The result on graph colouring should be compared with a recent result of Atserias and Dalmau [5] which shows that the promise graph colouring problem cannot be solved by a local consistency algorithm. In particular, this implies that for any constant t the 3-colourable graphs cannot be separated from those that are not t-colourable by a class (whose complement is) definable in Datalog. Since Datalog programs can be translated into sentences of FPC, our Theorem 5.3 can be seen as strengthening their result. It is worth examining this relationship more closely. It is known, from results of [4] and [8], that every class of bounded counting width (and therefore, in particular, any FPC definable class) that is the complement of a fixed-template constraint satisfaction problem (CSP) is already definable in Datalog. Hence, we can conclude from the result of Atserias and Dalmau that no FPC definable CSP separates the 3-colourable graphs from the non-t-colourable ones. However, since it is conceivable that a separating class for these two CSPs is FPC definable but not itself a CSP, our result is still a strengthening. But we can say still more. It can be deduced from the proof in [5] that the 3-colourable graphs and the non-t-colourable ones are not separable by any class definable in an existential positive infinitary logic ($\exists^{+,\omega}$). Moreover, it is a consequence of a very recent proof due to Rossman (published in the present volume [24]) that every class of bounded counting width that is preserved under homomorphisms is definable in $\exists^{+,\omega}$. Thus, we can conclude from these results that no homomorphism-closed class of bounded counting width separates the 3-colourable graphs from the non-t-colourbale ones. Since Theorem 5.3 easily

applies to all classes of bounded counting width and not just the FPC-definable ones; and it is conceivable that a separating class is not necessarily closed under homomorphisms, our result subsumes even this strengthened version of that of Atserias and Dalmau.

In Section 2 we introduce the problems, notation and provide background definitions. An outline of the steps involved in the reduction of Khot, Minzer and Safra is given in Section 3. The proof that the reductions involved are definable as first-order interpretations is given in Section 4 and certain consequences derived in Section 5.

2 Preliminaries

2.1 Hardness of Approximation in Optimization

We are interested in NP-hard optimization problems. A standard example is the problem MAX 3SAT, where the aim is to find, given a formula in 3CNF, an assignment of values to its variables that maximizes the number of clauses satisfied. Formally, consider a function problem M, which associates with every possible input instance I a value $M(I)$. In our example, MAX 3SAT maps a formula ϕ to the maximum number m of clauses of ϕ that can be simultaneously satisfied. While, in practice, we might be interested in finding an assignment that achieves this maximum, for the purpose of proving hardness, it suffices to show that it is hard to compute the number m. When finding $M(I)$ is hard, we may wish to approximate it, and we say that an algorithm computes a C-approximation (for a real number $C > 1$) of M if it produces a number $M'(I)$ with the guarantee that $M'(I) \leq M(I) \leq C \cdot M'(I)$.

For the sake of uniformity, we consider function problems that take values in $[0, 1]$. Thus, MAX 3SAT assigns to a 3CNF formula ϕ the maximum fraction of the clauses of ϕ that can be simultaneously satisfied. For MAX 3SAT, it is known that, unless P = NP, there is no polynomial-time algorithm that gives a C-approximation for any $C < 8/7$. Such hardness of approximation results are usually proved by means of a *hardness of separation*, which allows us to frame this in terms of the hardness of decision problems.

Formally, let A and B be two sets (i.e. decision problems) with $A \cap B = \emptyset$. We say that A and B are NP-hard to separate, if *every* set C with $A \subseteq C \subseteq \overline{B}$ is NP-hard, where $\overline{B}$ denotes the complement of B. For a function problem M, and a constant $c \in [0, 1]$, denote by c-M the set $\{I \mid M(I) \geq c\}$. Then, for constants c and s with $0 \leq s < c \leq 1$, we say that the *gap problem* Gap$M(c, s)$ is NP-hard if it is NP-hard to separate the sets c-M and $\overline{s\text{-}M}$. This implies, in particular, that unless P = NP, there is no polynomial-time algorithm giving a $\frac{c}{s}$-approximation of M. The value c in Gap$M(c, s)$ is called the *completeness parameter* and s the *soundness parameter*.

The first hardness of approximation results come from the PCP theorem [2, 15, 3]: one of its direct consequences is the NP-hardness of Gap3SAT$(1, \eta)$ for some constant η strictly less than 1. Håstad [17] obtained an optimum inapproximability result for MAX 3SAT. Namely, he showed that Gap3SAT$(1, \frac{7}{8} + \epsilon)$ is NP-hard for arbitrarily small ϵ. This is optimal since there is an easy $\frac{8}{7}$-approximation algorithm. Similarly, he also showed that Gap3XOR$(1 - \epsilon, \frac{1}{2} + \epsilon)$ is NP-hard for arbitrarily small ϵ. Again, this is optimal. Here, 3XOR is the problem where we are given a Boolean formula as a conjunction of clauses, each of which is the XOR of three literals and we aim to maximize the number of satisfied clauses. Note that the completeness parameter must be strictly less than 1, since the problem of determining whether such a formula is satisfiable or not is polynomial-time decidable. Thus 1-3XOR can be separated in polynomial time from $\overline{(1-\epsilon)\text{-3XOR}}$ for any ϵ.

Reductions

A common way of deriving further hardness of approximation results is via gap-reductions: given function problems A and B, a polynomial-time computable function f taking instances of A to instances of B is a *reduction* from $\mathrm{Gap}A(c,s)$ to $\mathrm{Gap}B(c',s')$ if for all instances I of A

- **Completeness:** if $A(I) \geq c$, then $B(f(I)) \geq c'$.
- **Soundness:** if $A(I) \leq s$, then $B(f(I)) \leq s'$.

It is easily seen that, if such a reduction exists and $\mathrm{Gap}A(c,s)$ is NP-hard, then so is $\mathrm{Gap}B(c',s')$.

2.2 Label Cover Games

Versions of *label cover* problems are ubiquitous in the study of hardness of approximation (see [13]). A particularly important case are the *unique games* of Khot [18], defined below. To arrive at the definition, we first introduce some terminology. For positive integers d and e, a relation $R \subseteq U \times V$ is said to be d-to-e if it relates each element of U to exactly d elements of V and each element of V to exactly e elements of U.

▶ **Definition 2.1** (d-to-d games)**.** *A* d-to-d game *is a tuple* (G, Σ, Φ)*, where* $G = (V, E)$ *is a multi-graph*[1]*,* Σ *is a finite alphabet and* $\Phi : E \to \mathcal{P}(\Sigma^2)$ *assigns to each edge* $e \in E$ *a* d*-to-*d *binary relation.*

A colouring $\chi : V \to \Sigma$ satisfies *an edge* (u, v) *if* $(\chi(u), \chi(v)) \in \Phi(u, v)$.

The value *of the game* (G, Σ, Φ) *is the maximum over all colourings of the proportion of edges in* E *that are satisfied.*

In this paper, we are particularly interested in 2-to-2 games and 1-to-1 games, the latter also being known as *Unique Games*. We write UG_q for the function problem of determining the value of an instance of unique games with an alphabet of size q. We can then state Khot's unique games conjecture.

▶ **Conjecture 2.2** (Unique Games Conjecture (UGC) [18])**.** *For any* $\delta, \epsilon > 0$*, there exists a positive integer* q *so that* $\mathrm{GapUG}_q(1-\epsilon, \delta)$ *is NP-Hard.*

The significance of the conjecture is that it has been shown that many optimal hardness of approximation results follow from it, including Max Cut and Vertex Cover [19, 23, 18].

The best known hardness result for unique games, towards proving Conjecture 2.2 is that $\mathrm{GapUG}_q(\frac{1}{2} - \epsilon, \delta)$ is NP-Hard for arbitrarily small δ and ϵ. This is obtained as a consequence of the hardness of 2-to-2 games established by Khot, Minzer and Safra, which we return to in Section 3.

▶ **Theorem 2.3** (Khot-Minzer-Safra)**.** *For any* $\delta, \epsilon > 0$*, there exists a positive integer* q *so that* $\mathrm{Gap2to2}_q(1-\epsilon, \delta)$ *is NP-Hard.*

It is conjectured that Theorem 2.3 can be strengthened to make the completeness parameter 1, but this remains unproved.

In this paper, we are particularly concerned with *weighted* 2-to-2 and 1-to-1 games, attaching a weight to each constraint.

[1] That is to say, there may be multiple edges between the same pair of vertices. In the sequel we refer simply to graphs to mean multi-graphs.

▶ **Definition 2.4** (Weighted d-to-d games). *A* weighted d-to-d game *is a tuple* (G, Σ, Φ, w), *where* (G, Σ, Φ) *is a* d*-to-*d *game and* $w : E(G) \to \mathbb{R}^+$ *is a function assigning a positive real weight to each constraint.*

Let $\mathrm{tot} = \sum_{e \in E(G)} w(e)$ *be the total weight. The* value *of the game* (G, Σ, Φ, w) *is the maximum over all colourings* $\chi : V \to \Sigma$ *of the fraction* $\sum_{e \in S_\chi} w(e)/\mathrm{tot}$, *where* S_χ *denotes the set of edges* $e = (u, v)$ *for which* $(\chi(u), \chi(v)) \in \Phi(e)$.

We write $\mathcal{WG}_{2:2;q}$ to denote the class of weighted 2-to-2 games with q labels and $\textsc{Weight2to2}_q$ to denote the function taking such a game to its value. Similarly, we write UG_q and $\textsc{WeightUG}_q$ for the functions giving the values of unique games and weighted unique games with q labels respectively.

2.3 Undefinability of Approximation

We assume the reader is familiar with first-order logic and the basics of finite model theory. A good introduction is to be found in [14]. Our structures are finite structures in a finite relational vocabulary. Our main inexpressibility results are stated for *fixed-point logic with counting* (FPC). We do not need a formal definition here but note that every property definable in FPC is decidable in polynomial-time and indeed FPC can be understood as a complexity class defined by *symmetric polynomial-time* computation. For full definitions, refer to [10] and references therein.

The two properties of FPC that we do need are that (1) every class of structures definable in FPC has *bounded counting width*; and (2) that the class of properties definable in FPC is closed under *first-order interpretations*. We elaborate on these below.

For a function problem M, and real numbers c and s with $0 \leq s < c \leq 1$, we say that $\mathrm{Gap}M(c, s)$ is undefinable in FPC if there is no FPC definable class of structures that separates the sets c-M and $\overline{s\text{-}M}$. Atserias and Dawar [6] initiated a study of the FPC undefinability of approximations, showing that many of the NP-hardness results for gap problems can be reproduced as *unconditional* undefinability results in FPC. In particular $\mathrm{Gap3SAT}(1, \frac{7}{8} + \epsilon)$ is not FPC definable. More significantly, they established the following

▶ **Theorem 2.5** (Atserias-Dawar [6]). $\mathrm{Gap}3XOR(1, \frac{1}{2} + \epsilon)$ *is not* FPC *definable.*

Note the completeness parameter of 1 in the statement, which contrasts with $1 - \epsilon$ in the case of Theorem 2.3. Perfect completeness cannot be established in the case of NP-hardness because satisfiability of XOR formulas *is* decidable in polynomial-time. However, it is not definable in FPC and this allows the stronger result in the context of undefinability. This is crucial to the application we make of Theorem 2.5 in Section 5.3

Following up on this work, Tucker-Foltz [26] studied the undefinability of gaps in unique games. In particular, he established the inapproximability of unique games in FPC by any constant factor and the FPC-undefinability of $\mathrm{GapUG}_q(\frac{1}{2}, \frac{1}{3} + \delta)$ for a suitable value of q.

Counting Width

For relational structures $\mathbb{A}$ and $\mathbb{B}$ in the same vocabulary, and a positive integer k, $\mathbb{A} \equiv^k \mathbb{B}$ denotes that the two structures cannot be distinguished by any sentence of first-order logic *with counting* using no more than k distinct variables. For a class $\mathcal{C}$ of structures, the *counting width* of $\mathcal{C}$ is the function $\nu : \mathbb{N} \to \mathbb{N}$ such that for any n, $\nu(n)$ is the least k such that $\mathcal{C}$, restricted to structures with at most n elements is a union of $\equiv^k$-equivalence classes. Any class that is definable by a sentence of FPC has counting width bounded by a constant. Almost all results showing that a class is not definable in FPC proceed by showing that it, in fact, does not have bounded counting width.

Interpretations

A *first-order interpretation* of a relational vocabulary τ in a vocabulary σ is a sequence of σ-formulas in first-order logic, which can be seen as mapping σ-structures to τ-structures. There are many variations of the precise definition in the literature. We use the version defined in [6] and refer the reader to that for the formal definition. Given a function problem A whose instances are σ-structures and a function problem B whose instances are τ-structures, an interpretation Θ of τ in σ is a Gap$A(c,s)$ to Gap$B(c',s')$ reduction if $A(\mathbb{A}) \geq c$ implies $B(\Theta(\mathbb{A})) \geq c'$ and $A(\mathbb{A}) \leq s$, then $B(\Theta(\mathbb{A})) \leq s'$. Definability in FPC and the property of having bounded counting width are both closed under first-order reductions. That is to say, if Gap$B(c',s')$ is FPC-definable and there is a first-order reduction of Gap$A(c,s)$ to Gap$B(c',s')$, then Gap$A(c,s)$ is FPC-definable as well.

3 The Reduction

The proof of Theorem 2.3 was completed in 2018 and remains to this day the most significant advance towards establishing the Unique Games Conjecture since the latter was formulated by Khot in [18]. The proof proceeds by a reduction from Gap3XOR$(1-\epsilon, \frac{1}{2}+\delta)$ and was presented in a series of papers [20, 25, 11]. The main difficulty lies in proving the combinatorial conditions that the soundness analysis relies on. The full reduction and proof of correctness can be found in [22, Chapter 3].

Our aim in the present paper is to show that the reduction constructed has two crucial properties. First, it preserves perfect completeness and thus can be seen as a reduction from Gap3XOR$(1, \frac{1}{2}+\delta)$. Secondly, with small modifications which do not affect the soundness or completeness analysis, it can be described as a first-order interpretation. Together these establish the main theorem.

▶ **Theorem 3.1.** *For every $\delta > 0$, there exists $q \in \mathbb{N}^+$ for which* Gap$\textsc{Weight2to2}_q(1,\delta)$ *is not* FPC *definable.*

In proving this, we do not need to reprise the difficult soundness analysis carried out by Khot et al. Rather we study the actual construction involved in the reduction. For this purpose, we describe the reduction in some detail in this section, and take up the two issues of perfect completeness and first-order definability in the next.

3.1 Regular 3XOR

An instance of 3XOR can be seen as a system of linear equations over the field $\mathbb{F}_2$ with exactly three variables appearing in each equation. We say that such an instance is *d-regular* if every variable appears in exactly d equations and no two equations share more than one variable. It is known that the NP-hardness of Gap3XOR$(1-\epsilon, \frac{1}{2}+\delta)$ holds even when restricted to *d-regular* instances for some fixed value of d (indeed, taking $d = 5$ suffices, see [22, Theorem 3.3.1]). In Section 4.3 we show that this is also true of the undefinability in FPC of Gap3XOR$(1, \frac{1}{2}+\delta)$ From now on, we restrict attention to d-regular instances for a suitable fixed value of d, and we call the resulting function problem Gap$\textsc{Regular}$3XOR.

3.2 Reducing to Transitive Games

In the first step of the reduction, we reduce regular 3XOR instances to label cover games with a mixture of 2-to-2 and 1-to-1 constraints, with an additional transitivity requirement. We formally define these below.

▶ **Definition 3.2** (Transitive 2-to-2 games). *A transitive 2-to-2 game is a tuple* (G, Σ, Φ) *where* $G = (V, E)$ *is a graph,* Σ *is a finite alphabet and* $\Phi : E \to \mathcal{P}(\Sigma^2)$ *assigns to each edge* e *either a* 2*-to-*2 *or a* 1*-to-*1 *relation and whenever* $\Phi(u, v)$ *is* 1*-to-*1*, then for any edge* (v, w)*,* $\Phi(u, w)$ *is the composition of* $\Phi(u, v)$ *and* $\Phi(v, w)$*.*

Note that the condition on composition only applies when $\Phi(u, v)$ is 1-to-1, but $\Phi(v, w)$ may be 1-to-1 or 2-to-2, and this determines whether $\Phi(u, w)$ is 1-to-1 or 2-to-2.

Now, fix an instance I of GapRegular3XOR, with X being the set of variables that appear in I and E the set of equations. Thus, each equation $e \in E$ is of the form $x+y+z = b$ for some $b \in \mathbb{F}_2$. We refer to x, y and z as the variables occurring in e and b as the *right-hand side* of e.

Fix a positive integer k and let $\mathcal{U} \subseteq E^k$ be the set of k-tuples U of equations, satisfying the following properties:

- no variable occurs in more than one equation of U; and
- if variables x and y appear in distinct equations of U, there is no equation in E (even outside U) in which both x and y occur.

For $U = (e_1, \ldots, e_k) \in \mathcal{U}$, let X_U denote the set of variables occuring in equations in U and for $i \in \{1, \ldots, k\}$ let $v_i \in \mathbb{F}_2^X$ denote the vector which has 1s in the three coordinates corresponding to the variables occurring in e_i and 0s everywhere else. We define the *space of side-conditions* corresponding to U to be $H_U = \mathrm{Span}(v_1, \ldots, v_k)$. We say that a linear function $f : \mathbb{F}_2^X \to \mathbb{F}_2$ satisfies the equations in U if $f(v_i) = b_i$ for all i, where b_i is the right-hand side of e_i.

Now, fix a parameter l with $l \leq |X|$, and we define $\mathcal{L}_U$ to be the collection of l-dimensional subspaces of $\mathbb{F}_2^X$ which are linearly independent of H_U. That is

$$\mathcal{L}_U = \left\{ L \subseteq \mathbb{F}_2^X \mid \dim(L) = l,\ L \cap H_U = \{\mathbf{0}\} \right\}.$$

The trivial intersection ensures that for any subspace $L \in \mathcal{L}_U$, any linear function $f : L \to \mathbb{F}_2$ can be uniquely extended to one on $L + H_U$[2] so that $f(v_i) = b_i$ for all i. Therefore, the number of linear functions on $L + H_U$ satisfying the equations in U is exactly 2^l.

We can now define the reduction Θ that takes the instance I to a 2-to-2 transitive game $\Theta(I)$. The reduction depends on the choice of parameters k and l. We omit the details on how to select the right parameters.

Vertices. The vertices of $\Theta(I)$ are pairs (U, L), where $U \in \mathcal{U}$ and $L \in \mathcal{L}_U$.

Alphabet. The alphabet is a set of labels of size 2^l. As noted above, for each vertex (U, L), there are exactly 2^l linear functions on $L + H_U$ satisfying the equations in U. We fix, for each (U, L), a bijection between the alphabet and this set of linear functions. Henceforth, we simply treat the functions themselves as labels.

Constraints. Given a pair of vertices $u = (U, L)$ and $v = (U', L')$, the constraint $\Phi(u, v)$ is a 1-to-1 relation if

$$\dim(L + H_U + H_{U'}) = \dim(L' + H_U + H_{U'}) = \dim(L + L' + H_U + H_{U'})$$

and a 2-to-2 relation if

$$\dim(L + H_U + H_{U'}) = \dim(L' + H_U + H_{U'}) = \dim(L + L' + H_U + H_{U'}) - 1.$$

[2] Here the sum is to be understood as vector space sum, i.e. $L + H_U$ is the space spanned by the union of L and H_U.

To define the relation, note that any function $f : L + H_U \to \mathbb{F}_2$ has a unique extension to $L+H_U+H_{U'}$ (by the conditions in the definition of $\mathcal{U}$). Then, we relate f to $f' : L'+H_{U'} \to \mathbb{F}_2$ if, and only if, f and f' agree on the shared space $(L + H_U + H_{U'}) \cap (L' + H_U + H_{U'})$.

It is the case for any pair, that $\dim(L+H_U+H_{U'}) = \dim(L'+H_U+H_{U'})$ [20, Lemma 4.3]. Let us call this dimension D. By [20, Lemma 4.4], any linear function $f : L + H_U \to \mathbb{F}_2$ satisfying the equations of U has a unique extension to $(L + H_U + H_{U'})$ that also satisfies the equations of U'. Then, it is easily seen that if $\dim(L + L' + H_U + H_{U'}) = D$, then f has exactly one label of (U', L') that it is consistent with, and if $\dim(L+L'+H_U+H_{U'}) = D+1$, there are exactly two such functions, thanks to the "free dimension". Hence, the constraints are 1-to-1 or 2-to-2 as required. The transitivity property of these constraints is established in [20, Appendix A].

3.3 The final (weighted) 2-to-2 game

The final step of the reduction is to transform the transitive game constructed in Section 3.2 into a *weighted* 2-to-2 game, getting rid of the 1-to-1 constraints. This weighted game is defined as follows.

Recall the transitive 2-to-2 game $\Theta(I)$ constructed in Section 3.2. The transitivity condition guarantees that the vertices of $\Theta(I)$ can be partitioned into cliques $C_1, \ldots, C_m$ so that edges in each clique are associated with 1-to-1 constraints. Moreover, these constraints are consistent in the sense that any colouring of a vertex V in a clique C can be extended in a unique way to a colouring of all vertices in C so that all edge constraints in C are satisfied. Also, by the transitivity condition, for distinct cliques C_i and C_j, either all pairs $(u, v) \in C_i \times C_j$ are connected by 2-to-2 constraints or none are. Furthermore, these 2-to-2 constraints are consistent in the sense that given a clique-consistent colouring for C_i and C_j, either all or none of these 2-to-2 constraints are satisfied.

The final (weighted) 2-to-2 instance $I^w_{2:2}$ we construct from $\Theta(I)$ has as vertices the vertices of $\Theta(I)$ and as edges all edges (u, v) of $\Theta(I)$ where u and v are in distinct cliques. For each such edge, with $u \in C_i$ and $v \in C_j$, we associate the constraint $\Phi(u, v)$ which is as in $\Theta(I)$. The weight $w(u, v)$ is the probability assigned to (u, v) by the following sampling process:

- Choose $U \in \mathcal{U}$, uniformly at random.
- Choose a random pair L, L' so that (U, L) and (U, L') are connected by a 2-to-2 edge. Let C_i be the clique containing (U, L) and C_j be the clique containing (U', L')
- Choose uniformly at random a pair of vertices $(u, v) \in C_i \times C_j$.

3.4 Irregular soundness case

For the result in Section 5.3, we need the FPC-undefinability of a different gap problem based on 2-to-2 games. Specifically, we define the value of a game to be, not the fraction of constraints that can be satisfied, but the fraction of the vertices formed by the largest set X so that all constraints between nodes in X are satisfied. Moreover, we relax the notion of colouring to allow vertices to be coloured by multiple colours.

▶ **Definition 3.3.** *For a 2-to-2 game $((V, E), \Sigma, \Phi)$, a colouring $c : V \to \binom{\Sigma}{j}$ satisfies a set $X \subseteq V$ if $\forall (u, v) \in E \cap X^2 . \exists a \in c(u), b \in c(v).(a, b) \in \Phi(u, v)$.*

That is to say, a j-colouring, i.e., one that assigns a set of j colours to each vertex satisfies a set X if each constraint between vertices in X is satisfied by some choice among the colours assigned to the vertices.

▶ **Definition 3.4** (Irregular Values). *For constants j and q define the function* $\textsc{Irreg2to2}_{j,q}$ *to take a 2-to-2 game $((V,E),\Sigma,\Phi)$ to the fraction $|X|/|V|$ where X is the largest subset of V that is satisfied by some j-colouring $c : V \to \binom{\Sigma}{j}$.*

We can now state the theorem below, which is a consequence of Theorem 3.1.

▶ **Theorem 3.5** (Definable 2-to-2 Games Theorem with irregular soundness). *For every δ with $0 < \delta < 1$ and $j \in \mathbb{N}^+$, there exists $q \in \mathbb{N}^+$ so that* $\text{Gap}\textsc{Irreg2to2}_{j,q}(1,\delta)$ *is not* FPC *definable.*

It is not hard to see that this is a consequence of Theorem 3.1, and the corresponding claim for NP-hardness appears in e.g. [20]. For completeness, we give a short proof.

▶ **Lemma 3.6.** *For a weighted 2-to-2 game $I = ((V,E),\Sigma,\Phi,w)$ with $q = |\Sigma|$, if $\textsc{Irreg2to2}_{j,q}((V,E),\Sigma,\Phi) = \delta$, then $\textsc{Weight2to2}(I) = \Omega(\frac{\delta^2}{j^2})$.*

Proof. Let c be a j-colouring of V that satisfies a set X with $|X|/|V| \geq \delta$. By [22, Remark 3.4.9], there is a $\Omega(\delta^2)$ (weighted) fraction of the edges E which are satisfied by c, in the sense that for each such edge (u,v) there are colours a and b in $c(u)$ and $c(v)$ respectively such that $(a,b) \in \Phi(u,v)$. We now construct a standard colouring by a random process. That is, for each vertex $v \in V$, independently choose a colour $\chi(v)$ from $c(v)$ uniformly at random. For an edge (u,v), let $\Xi(u,v)$ be the indicator variable indicating whether $(\chi(u),\chi(v)) \in \Phi(u,v)$ and let Ξ be the overall value of the colouring χ. If $(u,v) \in X^2$, the probability that χ satisfies the constraint $\Phi(u,v)$ is at least $\frac{1}{j^2}$, as by definition, among the j^2 pairs in $c(u) \times c(v)$, at least one satisfies the constraint. Then

$$\begin{aligned}\mathbb{E}[\Xi] = \mathbb{E}[\sum_{(u,v)\in E} w(u,v)\Xi(u,v)] &= \sum_{(u,v)\in E} w(u,v)\mathbb{E}[\Xi(u,v)] \\ \geq \sum_{(u,v)\in E\cap X^2} w(u,v)\mathbb{E}[\Xi(u,v)] &\geq \sum_{(u,v)\in E\cap X^2} w(u,v)\frac{1}{j^2} \geq \Omega(\delta^2)\frac{1}{j^2}\end{aligned}$$

Thus, there is a colouring that satisfies at least $\Omega(\frac{\delta^2}{j^2})$ (weighted) fraction of the constraints. ◀

From Lemma 3.6, we can conclude Theorem 3.5. For any fixed δ and j, the proof of Theorem 3.1 gives us a q and an FO reduction that takes satisfiable 3XOR instances to satisfiable 2-to-2 games and instances that are at most η-satisfiable to 2-to-2 games with value at most $\Omega(\frac{\delta^2}{j^2})$. Then, by Lemma 3.6, this same reduction also maps at most η-satisfiable 3XOR instances to 2-to-2 games I for which $\textsc{Irreg2to2}_{j,q}(I) < \delta$.

3.5 $2 \leftrightarrow 2$ games

The definition of 2-to-2 games, Definition 2.4 only requires each constraint $\Phi(u,.v)$ to be a 2-to-2 relation, meaning that each element on the left is related to exactly two elements on the right and vice versa. However, the reductions yield games of a more restricted kind and this will be useful in Section 5.3. Say that a binary relation $R \subseteq A \times B$ is $2 \leftrightarrow 2$ if it is the disjoint union of bipartite graphs $K_{2,2}$. That is to say A and B can be each partitioned into sets $A = \bigcup_i A_i$ and $B = \bigcup_i B_i$ so that each A_i and B_i has exactly two elements and $R = \bigcup_i A_i \times B_i$.

We claim that the reductions in the proof of Thereom 3.1 yield games in which all constraint relations are $2 \leftrightarrow 2$. Specifically, given linear functions $f \neq f' : L + H_U \to \mathbb{F}_2$ so that their unique extension to the domain $L + H_U + H_{U'}$ only differ in their "free dimension", i.e. they agree in values on $(L+H_U+H_{U'}) \cap (L'+H_U+H_{U'})$, f and f' are related to the same two linear functions on $L' + H_{U'}$ (uniquely extensible to $L' + H_U + H_{U'}$) in $\Phi((U, L), (U', L'))$. Thus, the constraint relations constructed are $2 \leftrightarrow 2$.

4 Definability

The aim in this section is to show that the reduction outlined in Section 3 can, with minor modifications, be implemented as a first-order interpretation, preserving perfect completeness. Thus, it gives a first-order definable reduction from $\mathrm{Gap3XOR}(1, \frac{1}{2} + \delta)$ to $\mathrm{GapWeight2to2}_q(1, \delta')$ for a suitable choice of parameters. This establishes Theorem 3.1.

4.1 Perfect completeness

To show that the reduction from Section 3 preserves perfect completeness, it suffices to verify that instances of 3XOR that are satisfiable (i.e. have value 1) are mapped by the reduction to instances of $\mathcal{WG}_{2:2}$ which also have value 1.

Assume I is an 3XOR instance on a set of variables X that is satisfiable, and let $s : X \to \mathbb{F}_2$ be an assignment of values to the variables that satisfies it. Let $I^w_{2:2}$ denote the weighted 2-to-2 game that I maps to under the reduction. Then, for each vertex (U, L) of $I^w_{2:2}$ the restriction of s to $L + H_U$ is a valid label since all equations are satisfied, and it is easily seen that this labelling satisfies all constraints.

4.2 Vocabularies

An instance of 3XOR is defined as a structure over the vocabulary $\tau_{3\mathrm{XOR}} = \langle \mathrm{Eq}_0, \mathrm{Eq}_1 \rangle$ with two ternary relations. We think of the universe of a $\tau_{3\mathrm{XOR}}$-structure $\mathbb{A}$ as a set of variables. For $b \in \{0, 1\}$, a triple $(x, y, z) \in \mathrm{Eq}_b$ is understood as representing the equation $x + y + z = b$, where addition is modulo 2.

For each positive integer q, we define a vocabulary $\tau_{\text{(T) 2-to-2}_q}$ such that structures in this vocabulary represent instances of transitive 2-to-2 games over a label alphabet of size q. Let S_q denote the collection of permutations of $[q] = \{1, \ldots, q\}$. Note that there is a natural bijective correspondence between S_q and the 1-to-1 relations on $[q]$. Now, let $S_q^{\#2}$ denote the set of pairs of permutations $(\pi_1, \pi_2) \in S_q \times S_q$ such that for all $i \in [q]$, $\pi_1(i) \neq \pi_2(i)$. Then, it is easily seen that each 2-to-2 relation on $[q]$ can be seen as the union of such a pair of permutations. Our vocabulary $\tau_{\text{(T) 2-to-2}_q}$ contains a binary relation for each element of S_q and one for each element of $S_q^{\#2}$:

$$\tau_{\text{(T) 2-to-2}_q} = \langle (C_\pi)_{\pi \in S_q}, (C_{\pi_1,\pi_2})_{(\pi_1,\pi_2) \in S_q^{\#2}} \rangle.$$

We write $\mathcal{C}_1$ for the collection of relation symbols $(C_\pi)_{\pi \in S_q}$ and $\mathcal{C}_2$ for the collection of relation symbols $(C_{\pi_1,\pi_2})_{(\pi_1,\pi_2) \in S_q^{\#2}}$. Note that the vocabulary itself does not enforce the transitivity property, only a subset of the structures with this vocabulary are transitive 2-to-2 games.

For *weighted* 2-to-2 games, we construct a vocabulary that allows us to code instances with positive integer weights. This is more limiting than allowing rational weights, but as we show below in Section 4.6, it suffices for our purpose. Specifically,

$$\tau_{\text{(w) 2-to-2}_q} = \langle C, (\Phi_{\pi_1,\pi_2})_{(\pi_1,\pi_2) \in S_q^{\#2}} \rangle,$$

where C is unary, and the relations Φ_{π_1,π_2} are all ternary. A $\tau_{\text{(w) 2-to-2}_q}$-structure $\mathbb{A}$ is to be understood as an instance $I^w_{2:2} = (G, \Sigma, \Phi)$ of $\mathcal{G}^w_{2:2}$ with integer weights. The universe of $\mathbb{A}$ is the disjoint union of the set V of vertices of $I^w_{2:2}$, and the set C of constraints, with the unary relation C picking out this set. For each $(\pi_1, \pi_2) \in S^{\#2}_q$, the relation $\Phi_{\pi_1,\pi_2} \subseteq V^2 \times C$ contains those triples (u, v, c) where $\Phi(u, v)$ is a pair (R, w) with R being the 2-to-2 relation associated with the pair (π_1, π_2). The integer weight w is given by the number of elements c for which (u, v, c) is in the relation. We assume our structures satisfy the (first-order) axiom that ensures that there is at most one relation Φ_{π_1,π_2} in which triples (u, v, c) appear, for each choice of u and v.

4.3 Undefinability of Regular 3XOR

The reduction in Section 3 starts from *regular* games. In contrast, the undefinability result in Theorem 2.5 is stated for general 3XOR. Thus, we begin by arguing that the pfoof of Theorem 2.5 can actually be used to show the undefinability of GapRegular3XOR$(1, \eta)$ for some η strictly smaller than 1.

We first note that the Gap3XOR$(1, \frac{1}{2} + \delta)$ is FPC undefinable even for "half-regular" 3XOR instances. That is, 3XOR instances where each variable appears in the same number of equations. To see this, note that Lemma 5 in [6] uses a bipartite unique-neighbour expander graph with $r|X|$ nodes on the left and $|X|$ nodes on the right. Thus the graph is 3-left-regular and is an $(\alpha|X|, \beta)$ expander. Such graph exists for every X by [27, Chapter 4]. By a variation shown for Theorem 4.4 in [27], we laim the existence of such a graph with the extra condition that the graph is right-regular. Using this extra assumption on the graph in Lemma 5 in [6] the proof establishes that Gap3XOR$(1, \frac{1}{2} + \delta)$ is FPC undefinable even for "half-regular" 3XOR instances.

A half-regular instance can be converted into a regular one by ensuring that any two equations share at most one variable.

First, by the unique-neighbour expander property of the graph in Lemma 5 in [6], we can assume that the half-regular 3XOR instance has no repeated equations or repeated variables within an equation. This half-regular instance (X, Eq) can be converted into a regular one (call it (X^*, Eq^*)) by replacing every equation $e : x + y + z = b$ with three equations (as done in [22]): $x + y_e + z_e = b, \quad x_e + y + z_e = b, \quad x_e + y_e + z = b$, where x_e, y_e, and z_e are new variables only used for these equations.

As shown in [22], if X is fully satisfiable then so is X^* and if X is no more than $\frac{1}{2} + \delta$-satisfiable, then X^* is at most η-satisfiable for some $\eta < 1$ (for example, taking $\eta = 0.9$ suffices).

The reduction can be easily defined by a first-order interpretation.

4.4 Shuffling variables

One issue that arises with the games constructed in the reduction from Section 3 is that we have a fixed alphabet of size $q = 2^l$ and we associate with each vertex (U, L) an arbitrary bijection between this and the 2^l distinct linear functions on the space $L + H_U$ that satisfy the equations in U. The consistency across different vertices is then enforced by the constraint relations. In order to turn this into a first-order reduction, we want to choose these bijections in a symmetry-preserving fashion.

Let I be our starting instance of 3XOR and $I^T_{2:2} = \Theta(I)$ the transitive 2-to-2 game obtained from the first step of the reduction of Section 3, and let X be the set of variables of I. Let $\rho \in \text{Sym}_X$ be a permutation of X. This permutation has a natural action on other

objects constructed from X. In particular, for an equation e of the form $x + y + z = b$, we write $\rho(e)$ for the equation $\rho(x) + \rho(y) + \rho(z) = b$. When U is a tuple of such equations, we write $\rho(U)$ for the tuple obtained by applying ρ componentwise to each element of the tuple. Similarly, for other objects obtained by set and tuple constructions from X, we apply the permutation ρ to denote the natural induced action without defining it formally.

Furthermore, we also use ρ to denote the invertible linear map on $\mathbb{F}_2^X$ obtained by applying ρ to the basis $(e_x)_{x\in X}$, and extending linearly to all of $\mathbb{F}_2^X$. Thus, in particular, for a subspace $L \subseteq \mathbb{F}_2^X$, $\rho(L)$ denotes the image of this space under this map.

The following is now straightforward.

▶ **Lemma 4.1** (Shuffling Variables 1). *For any permutation $\rho \in Sym_X$, if U and $\rho(U)$ are both in $\mathcal{U}$, and $(U, L) \in V(I^T_{2:2})$, then $\rho(U, L) \in V(I^T_{2:2})$.*

Proof. Since ρ maps the basis of H_U formed by the left-hand sides of the equations in U to the corresponding basis of $H_{\rho(U)}$, we have $\rho(H_U) = H_{\rho(U)}$. By invertibility of ρ, a space L is then linearly independent of H_U if, and only if, $\rho(L)$ is linearly independent of $H_{\rho(U)}$. ◀

Now, we want to choose the bijections between our set of 2^l labels and the linear functions associated with a vertex (U, L) in such a way that whenever (U, L) and $\rho(U, L)$ are both vertices in $I^T_{2:2}$, then they commute with ρ. For this, fix a *canonical* space $\mathbb{F}_2^{3k}$ of dimension $3k$. For each $U \in \mathcal{U}$, we write $X_U \subseteq X$ for the set of variables that appear in U. Since U is a sequence of k equations with pairwise disjoint sets of variables, we can fix a bijection between X_U and $[3k]$ which induces an isomorphism $\mu_U : \mathbb{F}_2^U \to \mathbb{F}_2^{3k}$. These isomorphisms are easily seen to be ρ-invariant (for all ρ), that is,

$$\forall S \in \mathbb{F}_2^{X_U}. \quad \mu_{\rho(X_U)}(\rho(S)) = \mu_U(S).$$

Under this map, there is a fixed subspace $H \subseteq \mathbb{F}_2^{3k}$ of dimension k such that $\mu_U(H_U) = H$ for all U. Similarly, there is a fixed collection $\mathcal{L}$ of l-dimensional spaces such that $\mu_U(\mathcal{L}_U) = \mathcal{L}$. Thus, we can identify the vertices of $I^T_{2:2}$ uniquely with pairs (U, L^*) where $U \in \mathcal{U}$ and $L^* \in \mathcal{L}$. This is to be understood as the representation of the vertex $(U, \mu_U^{-1}(L^*))$.

Similarly, for linear functions f over $L \in \mathcal{L}_\mathcal{U}$, we can define

$$(\rho(f))(x) = f(\rho^{-1}(x)) : \rho(L) \to \mathbb{F}_2 \quad \text{and}$$

$$(\mu_U(f))(x) = f(\mu_U^{-1}(x)) : \mu_U(L) \to \mathbb{F}_2.$$

Then, a linear function f on $L + H_U$ satisfies the equations in U if, and only if, $\mu_U(f)$ satisfies the equations in $\mu_U(U)$. Hence, we can interpret in a canonical way the label of a node $(U, \mu_U^{-1}(L^*))$ as a linear function with domain $H + L^*$ satisfying the equations in $\mu_U(U)$.

We now show that this can be consistently applied to the constraints of the game.

▶ **Lemma 4.2** (Shuffling Variables 2). *Suppose $(U, L), (U', L') \in E(I^T_{2:2})$ and $\rho(U), \rho(U')$ are both in $\mathcal{U}$. Then*

- $(\rho(U, L), \rho(U', L')) \in E(I^T_{2:2})$
- $\Phi((U, L), (U', L')) = \Phi(\rho(U, L), \rho(U', L'))$

Proof. By Lemma 4.1, $\rho(U, L), \rho(U', L') \in V(I^T_{2:2})$. Also

$$\dim(\rho(L) + H_{\rho(U)} + H_{\rho(U')}) = \dim(\rho(L + H_U + H_{U'})) = \dim(L + H_U + H_{U'})$$

The equalities hold because the mapping ρ is an automorphism of $\mathbb{F}_2^X$. The analogous dimensionality property holds with the mapping of subspaces $(L' + H_U + H_{U'})$ and $(L + L' + H_U + H_{U'})$. Therefore, the dimensionality constraint for drawing edges is invariant under the action of ρ. This proves the first bullet point.

Then if $(f, f') \in \Phi((U, L), (U', L'))$, it means $\mu_U^{-1}(f)$ and $\mu_{U'}^{-1}(f')$ are consistent on the intersection of their domains. Then $\mu_{\rho(U)}^{-1}(f) = \rho(\mu_U^{-1}(f))$ and $\mu_{\rho(U')}^{-1}(f') = \rho(\mu_{U'}^{-1}(f'))$ are consistent too, meaning $(f, f') \in \Phi(\rho(U, L), \rho(U', L'))$. Hence $\Phi((U, L), (U', L')) \subseteq \Phi(\rho(U, L), \rho(U', L'))$. Applying the same argument to ρ^{-1} yields the other direction. ◀

4.5 The reduction to the transitive game

We now describe how the reduction Θ from Section 3.2 can be given as a first-order interpretation. Fix positive integers k and l, which are the parameters to the reduction. Given a (regular) 3XOR instance $\mathbb{A} = (X, \mathrm{Eq}_0^{\mathbb{A}}, \mathrm{Eq}_1^{\mathbb{A}})$, our interpretation maps it to the following (transitive) 2-to-2 game (with alphabet size 2^l) $\mathbb{B}$.

Universe. The universe of $\mathbb{B}$ consists of tuples of elements of X of length $4k + 2^{3k}$. These tuples can be seen as broken up into three parts.

- The first $3k$ elements $(u_{1,1}, \ldots, u_{k,3})$ are the $3k$ variables in some $U \in \mathcal{U}$. To define this, we need to say that they are, in order, the collection of variables of a k-tuple of equations, that no variable appears more than once, and that when two variables appear in distinct equations, they do not occur together in some other equation in $\mathbb{A}$.
- The next k elements $r_1, \ldots, r_k$ define the right-hand sides of the k equations in U. To encode these as binary values, we use $r_i = u_{1,1}$ to encode the value 0 and $r_i = u_{1,2}$ to encode the value 1. Since $u_{1,1}$ and $u_{1,2}$ are distinct, this works and can be specified by a first-order formula.
- The next 2^{3k} elements also encode bits, using the values of $u_{1,1}$ and $u_{1,2}$ as 0 and 1. Think of these as specifying a subset of $\mathbb{F}_2^{3k}$. We can write a first-order formula that says that this subset is a subspace L^* of dimension l (since l and k are fixed, the formula is simply a big disjunction over all subspaces). Finally, we can also write a first-order formula that checks that L^* is in $\mathcal{L}$.

  For completeness, here is the first-order sentence checking all these conditions.

$$\begin{aligned}\pi^U = &\bigwedge_{i=1}^{k} [\mathrm{Eq}_0(u_{i,1}, u_{i,2}, u_{i,3}) \wedge r_i = 0] \vee [\mathrm{Eq}_1(u_{i,1}, u_{i,2}, u_{i,3}) \wedge r_i = 1] \\ &\wedge \bigwedge_{(a,i)\neq(b,j)} u_{a,i} \neq u_{b,j} \\ &\wedge \bigwedge_{a\neq b,i,j} \neg \left(\exists x \bigvee_{(\alpha,\beta,\gamma)\in \mathrm{Perm}(u_{a,i},u_{b,j},x)} \mathrm{Eq}_0(\alpha, \beta, \gamma) \vee \mathrm{Eq}_1(\alpha, \beta, \gamma) \right) \\ &\wedge \bigvee_{L^*\in\mathcal{L}} \left(\bigwedge_{i=0}^{2^{3k}-1} b_i = L_i^* \right)\end{aligned}$$

Where $\mathrm{Perm}(x, y, z)$ describes the set of permutations of x, y, z.

We can thus, as required, identify the elements of $\mathbb{B}$ with pairs (U, L) which are the vertices of $\Theta(\mathbb{A})$.

Relations. Given two vertices (U, L) and (U', L') of $\mathbb{B}$, the type of constraint between them (1-to-1, 2-to-2 or no constraint at all) only depends on $\mu_U(L), \mu_{U'}(L'), r, r'$ and $I(U, U')$, where r,r' are the vectors of the right-hand sides of the equations and

$$I(U, U') \triangleq \{((a, i), (b, j)) \in (\{1, \ldots, k\} \times \{1, 2, 3\})^2 \mid u_{a,i} = u'_{b,j}\}$$

If two pairs of vertices agree on all five of these values, there is a permutation ρ of the variables that will take one to the other and then by Lemma 4.2, they must have the same constraint between them.

Note that each of these five parameters can take only a constant number of different values, so for each constraint $C \in \mathcal{C}_1 \cup \mathcal{C}_2$, there is a (constant) finite set S_C containing such 5-tuples so that (U, L) and (U', L') are connected by a constraint C if, and only if, $(\mu_U(L), \mu_{U'}(L'), r, r', I(U, U')) \in S_C$. The formula π^C defining the relation C in $\mathbb{B}$ simply states that the 5-tuple corresponding to a pair of vertices is in S_C. This translates to a disjunction of a finite number of cases and is clearly FO-definable. This concludes the reduction to the transitive game.

4.6 Weight approximation

We now show how to get a weighted 2-to-2 game, that is an approximation of the instance $I^w_{2:2}$ constructed in Section 3.3. The vertices of the game are exactly those in the structure $\mathbb{B}$ above. The main task is to define the weights, by defining a suitable set C of constraints. Recall that the vertices of $I^w_{2:2}$ are partitioned into cliques $C_1, \ldots, C_m$ based on the 1-to-1 constraints. Suppose $(U_1, L_1) \in C_i$ and $(U_2, L_2) \in C_j$ are two vertices connected by a 2-to-2 constraint. Then, the weight of the constraint is

$$\sum_{\substack{U,L,L' \\ L,L' \in \mathcal{L}_U \\ \dim(L \cap L') = l-1}} 1_{(U,L) \in C_i \wedge (U,L') \in C_j} \frac{1}{|\mathcal{U}|} \frac{1}{|\{L, L' \in \mathcal{L}_U \mid \dim(L \cap L') = l-1\}|} \frac{1}{|C_i||C_j|}.$$

Each of the three factors (apart from the indicator variable) describes the probability of a certain choice in the steps of the random process which define the weights.

Of course, $\frac{1}{|\mathcal{U}|}$ is constant for all pairs $(U_1, L_1), (U_2, L_2)$. Similarly, $\frac{1}{|\{L,L' \in \mathcal{L}_U | \dim(L \cap L') = l-1\}|}$ is constant by the symmetry argument presented in Section 4.4. Thus, removing them from the expression does not change the relative weights of the constraints. Also, the clique size only depends on $(U_1, L_1), (U_2, L_2)$, so the weight expression (without the normalising factors) simplifies to

$$\frac{|\{(U, L, L') \mid (U, L) \in C_i, (U, L') \in C_j\}|}{|C_i||C_j|}. \tag{1}$$

These weights are rational, so we cannot express them directly in structures over $\tau_{\text{(w) 2-to-2}_q}$, which is our vocabulary for describing integer-weighted games. One potential way to handle rational weights would be to multiply all weights with a common denominator. This is not a viable option since the number of different-sized cliques grows with the size of the input, making the common denominator too large. However, we have a workaround: instead of these weights, we give an approximation that does not change the soundness parameter significantly but makes the common denominator of the weights small enough (polynomial as a function of the input size) to be definable.

▶ **Lemma 4.3.** *Given a weighted 2-to-2 game $G = (V, \Sigma, \Phi, w)$, whose value is at most δ, any game $G' = (V, \Sigma, \Phi, w')$ where $\forall \phi \in \Phi. \frac{1}{\gamma} < \frac{w(\phi)}{w'(\phi)} < \gamma$ has value at most $\delta\gamma^2$.*

Proof (sketch). The sum of weights drops at most by a factor γ, and the sum of the weights of the satisfied constraints increases by at most a factor of γ. ◀

So, the idea is to approximate clique sizes so that the number of possible denominators is constant and their product grows only polynomially with the input size, while bounding the change with a suitable multiplicative factor γ.

Fix a vertex (U, L) in a clique C_i. Recall that $(U', L') \in C_i$ if, and only if, there is a one-to-one constraint between (U, L) and (U', L') in $\mathbb{B}$. First, let us split the equations in U' into two groups: "useful" and "useless" ones. An equation in U' is useful (for U) if it shares at least one variable with U and useless otherwise. Note that the number of useful equations of (U', L') only depends on U', not on L'.

Next, we define an equivalence relation $\equiv_U$ on the vertices of the game as follows: $(U_1, L_1) \equiv_U (U_2, L_2)$ iff

- $\mu_{U_1}(L_1) = \mu_{U_2}(L_2)$.
- U_1 and U_2 have the same useful equations (for U), and these equations are in the same positions within the k-tuple.
- The right-hand sides of the equations in U_1 and U_2 are the same.

It is easily seen that this is, indeed, an equivalence relation.

Note that the clique C_i is invariant under the equivalence relation $\equiv_U$: each equivalence class is either contained in C_i or disjoint with it, by Lemma 4.2 (choosing ρ to be a permutation that fixes the variables of U and any useful equations).

Now, for any f with $0 \leq f \leq k$, we can establish an upper bound on the number of equivalence classes with f useful equations. Recall that any node (U', L') can be uniquely represented by U' and the subspace $\mu_{U'}(L') = L^* \in \mathcal{L}$:

- The number of possible subspaces $L^* \subseteq \mathbb{F}_2^{3k}$ is at most $2^{2^{3k}}$, as that is an upper bound for $|\mathcal{L}|$ (in fact, it is much smaller, but for our purposes, this upper bound suffices).
- The number of ways to choose the positions of the useful equations is $\binom{k}{f} \leq 2^k$.
- The number of choices for the right-hand sides of the equations is 2^k.
- Since the 3XOR instance is regular (each variable appears in at most d equations), the number of equations sharing a variable with U is at most $3kd$, so the number of ways of choosing the useful equations is bounded by $(3kd)^k$.

These bounds are all constants, so the number of equivalence classes within the clique, with f useful equations (call it $\nu^f_{U,L}$) is bounded by a constant Ψ for all f, U, L.

The number of elements in an equivalence class with f useful equations is simply the number of ways to set the remaining $k - f$ equations. This can be approximated by $|\mathrm{Eq}|^{k-f}$. Given f useful equations, the probability of a random set of $k - f$ equations having common variables with U, the set of useful equations or each other, or making the k-tuple invalid by having two variables from different equations which have a common equation in the 3XOR instance, converges to zero ($O\left(\frac{k^2}{|X|}\right)$) as the instance size grows, due to the regularity condition. By adding all the approximate sizes of the equivalence classes within C_i, we can conclude that the approximation

$$\chi(\boldsymbol{\nu_{U,L}}) \triangleq \chi(\nu^0_{U,L}, \nu^1_{U,L}, \ldots, \nu^k_{U,L}) \triangleq \sum_{f=0}^{k} \nu^f_{U,L} |\mathrm{Eq}|^{k-f} \approx |C_i|$$

is accurate within an arbitrarily small factor as the input size grows. Using this approximation in the weight expression (1), we see that $\prod_{\boldsymbol{v} \in \{0,\ldots,\Psi\}^{k+1}} \chi(\boldsymbol{v})^2$ is a common denominator of all weights. Multiplying all weights by this number, we get the expression

$$\begin{aligned} w((U_1,L_1),(U_2,L_2)) &= |\{(U, L, L') \mid (U, L) \in C_i, (U, L') \in C_j\}| \\ &\cdot \prod_{\boldsymbol{v} \in \{0,\ldots,\Psi\}^{k+1}} \begin{cases} \chi(\boldsymbol{v}) & \text{if } \boldsymbol{v} \neq \boldsymbol{\nu_{(U_1,L_1)}} \\ 1 & \text{if } \boldsymbol{v} = \boldsymbol{\nu_{(U_1,L_1)}} \end{cases} \cdot \prod_{\boldsymbol{v} \in \{0,\ldots,\Psi\}^{k+1}} \begin{cases} \chi(\boldsymbol{v}) & \text{if } \boldsymbol{v} \neq \boldsymbol{\nu_{(U_2,L_2)}} \\ 1 & \text{if } \boldsymbol{v} = \boldsymbol{\nu_{(U_2,L_2)}} \end{cases} \end{aligned} \tag{2}$$

As we see next, we can define a reduction in FO to weighted 2-to-2 games using these approximate weights.

4.7 Defining the weighted game

Finally, we are ready to show that the construction of a weighted 2-to-2 game with approximate weights as above can be given by an FO interpretation.

Universe. We need to define the set of vertices, and the set of constraints. The elements of the universe are tuples of elements of X (the set of variables of the 3XOR instance I) of length $8k + 1 + 2^{3k+1} + Q$, where Q is a parameter we define below.

A vertex (U, L) is coded by the first $4k + 2^{3k}$ elements of this tuple, as before, followed by a sequence of 0s. Recall that we code bits 0 and 1 by the first and second elements of the tuple. The first of these 0s is to be interpreted as an indicator that the tuple is a vertex (it will be 1 for a constraint), and the rest are padding to make the length of the tuples match.

A constraint c is coded by a tuple where the first $4k + 2^{3k}$ elements represent a vertex (U, L), this is followed by a 1 (i.e. a repeat of the second element of the tuple) and then the next $4k+2^{3k}$ represent a second vertex (U', L'). The rest of the tuple codes a unique identifier of the constraint, ID. We construct the interpretation so that for all fixed $(U, L), (U', L')$, there are $w((U, L), (U', L'))$ different identifiers where w is the approximate weight described above. We show that for this weight function, there is a formula W which defines a set of exactly $w((U, L), (U', L'))$ tuples extending the description of (U, L) and (U', L').

▶ **Lemma 4.4.** *There exists $Q \in \mathbb{N}^+$ and a first-order formula W which defines a set T of tuples coding pairs $(U, L), (U', L')$ together with a Q-element unique identifier and such that for each fixed $(U, L), (U', L')$, T contains exactly $w((U, L), (U', L'))$ many tuples extending $(U, L), (U', L')$.*

The proof of this lemma, constructing the formula W is in Section 4.8 below.

Thus, we can define the formulas defining the set of vertices and constraints. For simplicity, we use $U, L, U', L', \mathrm{ID}$ to describe the sub-tuple of variables in their corresponding parts of the N-tuple, where $N = 8k + 1 + 2^{3k+1} + Q$.

$$\mathrm{Node}(U, L, \mathrm{IsConstraint}, U', L', \mathrm{ID}) \equiv \mathrm{IsConstraint} = 0 \wedge \pi^U(U, L) \wedge \bigwedge_{x \in (U', L', \mathrm{ID})} x = 0$$

To check if it is a valid constraint, we need

$$\mathrm{Constraint}(U, L, \mathrm{IsConstraint}, U', L', \mathrm{ID}) \equiv \mathrm{IsConstraint} = 1 \wedge \pi^U(U, L) \wedge \pi^U(U', L') \wedge \bigvee_{C \in \mathcal{C}_2} \pi^C((U, L), (U', L')) \wedge W((U, L), (U', L'), \mathrm{ID})$$

Constraints. For each $C_{\pi_1,\pi_2} \in \mathcal{C}_2$, we can construct the formula that defines the set of triples (x, y, c) where $x = (U, L, 0, \ldots, 0)$ $y = (U', L', 0 \ldots, 0)$ and $c = (U, L, 1, U', L', \mathrm{ID})$, such that there is a constraint of type C between x and y and ID is a valid id of a constraint between them.

$$\Phi_{\pi_1,\pi_2}(x, y, c) \equiv \pi^{C_{\pi_1,\pi_2}}(x, y) \wedge (U, L) = (U_1, L_1) \wedge (U', L') = (U_2, L_2).$$

This completes the proof of Theorem 3.1.

4.8 Defining W

To prove Lemma 4.4 we define a first-order formula $W(x, y, z)$ in the vocabulary $\tau_{3\mathrm{XOR}}$, where x, y and z are tuples of free variables. The formula is such that if x and y are interepreted by the elements coding the nodes (U, L) and (U', L') respectively, then there are exactly $w((U, L)(U', L'))$ assignments of values to the tuple z that make W true. Here $w(U, L)(U', L')$ is the expression given in Equation 2.

To define W, we construct formulas defining various elements of Equation 2. More precisely, for various numerical expressions $e(x, y)$, which depend on the values assigned to x and y, we construct formulas we denote $w_{q,e}(x, y, z)$, where q is the length of the tuple of variables z. These formulas have the property that when x and y are interepreted by the elements coding the nodes (U, L) and (U', L') the number of q-tuples that can be assigned to z to make $\omega_{q,e}$ true is exactly $e(x, y)$. As before, we use 0 and 1 to denote the first and second elements of the tuple. Also, for a first-order formula $\phi(x, y)$, let 1_ϕ denote the indicator variable that ϕ is true (under an assignment of values to x and y).

$e = 1$: $\omega_{1,e}(x, y, z) \equiv (z = 0)$

$e = 1_\phi$: $\omega_{1,e}(x, y, z) \equiv (z = 0) \wedge \phi(x, y)$

$e = e_1 \times e_2$: Given ω_{q_1,e_1} and ω_{q_2,e_2}, we can define

$$\omega_{q_1+q_2,e}(x, y, z_1, \ldots, z_{q_1}, z_{q_1+1}, \ldots, z_{q_2}) \equiv \omega_{q_1,e_1}(z_1, \ldots, z_{q_1}) \wedge \omega_{q_2,e_2}(z_{q_1+1}, \ldots, z_{q_2})$$

$e = e_1 + e_2$: Given ω_{q_1,e_1} and ω_{q_2,e_2}, (assuming without loss of generality that $q_2 \geq q_1$, we can define

$$\omega_{1+q_2,e}(x, y, z_1, z_2, \ldots, z_{q_2+1}) \equiv \left[z_1 = 0 \bigwedge \omega_{q_1,e_1}(z_2, \ldots, z_{q_1+1}) \bigwedge \bigwedge_{i=q_1+2}^{q_2+1} z_i = 0 \right] \bigvee \left[z_1 = 1 \bigwedge \omega_{q_2,e_2}(z_2, \ldots, z_{q_2+1}) \right]$$

$e = |\mathrm{Eq}|$: It suffices to take a formula defining the disjoint union of the relations Eq_0 and Eq_1.

$$\omega_{4,e}(x, y, z_1, z_2, z_3, z_4) \equiv (z_1 = 0 \wedge \mathrm{Eq}_0(z_2, z_3, z_4)) \vee (z_1 = 1 \wedge \mathrm{Eq}_1(z_2, z_3, z_4))$$

$e = |\{(U_1, L_1, L_2) \mid (U_1, L_1) \in C_i, (U_1, L_2) \in C_j\}|$: The numerator in Equation 1 (and a term in Equation 2) is $e = |\{(U_1, L_1, L_2) \mid (U_1, L_1) \in C_i, (U_1, L_2) \in C_j\}|$. We can get a formula for this by defining exactly this set of tuples. Here z is a tuple of variables composed of three tuples z_1, z_2 and z_3 where z_1 has length $4k$ and each of z_2 and z_3 is of length 2^{3k}.

$$\omega_{4k+2*2^{3k},e}(x, y, z) = \pi^U(z_1, z_2) \wedge \pi^U(z_1, z_3) \wedge \bigvee_{C \in \mathcal{C}_2} C((z_1, z_2), (z_1, z_3) \wedge \bigvee_{C \in \mathcal{C}_1} C((z_1, z_2), x) \wedge \bigvee_{C \in \mathcal{C}_1} C((z_1, z_3), y)$$

Defining the size of the equivalence classes. Another element of Equation 2 are conditions of the form $\nu^f_{U,L} = r$ for various values of r. We now construct a formula $\nu^{f,\geq r}(x)$ with $4k + 2^{3k}$ free variables that expresses the condition $\nu^f_{U,L} \geq r$ when x is interpreted by the tuple coding (U, L). In the following, lower case letters u, l, possibly with subscript indices always denote tuples of variables of length $4k$ and 2^{3k} respectively. Recall that two elements in the clique are in the equivalence relation $\equiv_{(U,L)}$ if, and only if, their L values are the same and share the same useful equations with the same positions.

We begin with defining a couple of auxiliary formulas. For any $j \in \{1, \ldots, k\}$, the formula $\text{useful}_j(x, u)$ says of a tuple u that the jth equation it represents is useful and the formula $\text{diff}_j(u_1, u_2)$ asserts that the two tuples u_1 and u_2 differ in the jth equation:

$$\text{useful}_j(x,u) \equiv \bigvee_{i \in \{1,\ldots,3k\}} \left(u_{3(j-1)+1} = x_i \vee u_{3(j-1)+2} = x_i \vee u_{3(j-1)+3} = x_i\right); \text{ and}$$

$$\begin{aligned}\text{diff}_j(u_1,u_2) \equiv &(u_1)_{3(j-1)+1} \neq (u_2)_{3(j-1)+1} \vee (u_1)_{3(j-1)+2} \neq (u_2)_{3(j-1)+2} \vee \\ &(u_1)_{3(j-1)+3} \neq (u_2)_{3(j-1)+3} \vee (u_1)_{3k+j} \neq (u_2)_{3k+j}.\end{aligned}$$

With these, we can define $\nu^{f,\geq r}(x)$ as a formula whiuch asserts the existence of r nodes

$$\exists u_1, l_1, \ldots, u_r, l_r \bigwedge_i \pi^U(u_i, l_i);$$

which are are in the same clique as the node coded by x

$$\bigwedge_i \bigvee_{C \in \mathcal{C}_1} C(x, u_i, l_i);$$

all have f useful equations

$$\bigwedge_{i \in \{1,\ldots,r\}} \left\{ \bigvee_{S \subseteq \{1,\ldots,k\}, |S|=f} \left[\bigwedge_{j \in \{1,\ldots,k\}} \text{useful}_j(x, u_i) \leftrightarrow j \in S \right] \right\};$$

and such that no two nodes are $\equiv_{(U,L)}$ equivalent when x is interpreted as (U, L)

$$\bigwedge_{i \neq j \in \{1,\ldots,r\}} l_i \neq l_j \vee \bigvee_{o \in \{1,\ldots,k\}} \left(\text{useful}_o(u_i) \wedge \text{diff}_o(u_i, u_j)\right).$$

Then, as usual, $\nu^{f,r}(x) \equiv \nu^{f,\geq r}(x) \wedge \neg\nu^{f,\geq(r+1)}(x)$. To give an expression for $\omega_q(\nu^f_{U,L})$ for some q, we can rewrite it as $\sum_{r=1}^{\Psi} 1_{\nu^{f,r}_{U,L}} \cdot r$ and construct the expression using the composition rules (constants can be constructed via repeated addition of ones, addition, multiplication and indicator variables are defined above)

Putting it all together. For each term in Equation 2, we have described how to define a corresponding formula. Case splits can be handled via indicator variables and constants by repeatedly adding 1s. By a repeated application of the addition and multiplication rules, W can be constructed.

5 Consequences

5.1 Unique Games

An immediate corollary of the definable 2-to-2 games theorem is the inapproximability of unique games by any constant factors:

Given a (weighted) 2-to-2 game I, we can map it to a Unique Game I' by splitting every constraint into two: given a constraint of type C_{π_1,π_2}, we can replace them with two 1-to-1 constraints of type C_{π_1} and C_{π_2}. A colouring of the nodes then satisfies the constraint C_{π_1,π_2} in I if, and only if, exactly one of thee two constraints is satisfied in I'. Note that a colouring can only satisfy at most one of the two constraints. This gives a reduction from $\text{GapWeight2-to-2}_q(1, \delta)$ to $\text{GapWeightUG}_q(\frac{1}{2}, \frac{\delta}{2})$ for any $\delta > 0$.

This reduction is clearly FO-definable: the universe remains the same; then, for a 1-to-1 constraint C_π, we can determine if (x, y, c) represents a constraint of this type with the sentence $\Phi_\pi(x, y, c) \equiv \bigvee_{\pi_2} \Phi_{\pi,\pi_2}(x, y, c)$.

▶ **Theorem 5.1.** *For every $\delta > 0$, there exists $q \in \mathbb{N}^+$ so that* Gap$\textsc{WeightUG}_q\frac{1}{2}, \delta)$ *is* FPC *undefinable.*

This undefinability gap is stronger, at least in terms of the completeness and soundness parameters, than the gaps proved by Tucker-Foltz [26], the only previously known undefinability gaps for Unique Games. However, our construction uses weighted instances, so we can only conclude the undefinability gap over the domain of weighted unique games. Since the gaps in [26] are proved for unweighted games, they are incomparable to Theorem 5.1.

5.2 Vertex Cover

Another consequence of Theorem 2.3 is the NP-Hardness of approximating the Vertex Cover problem by a factor better than $\sqrt{2}$. The Unique Games Conjecture implies that nothing better than a factor 2 approximation is possible. This is tight, since polynomial-time algorithms achieving a 2-approximation are known. Before the results of Khot et al. establishing Theorem 2.3 the best known inapproximability result, conditional only on P $\neq$ NP, was ≈ 1.36. Atserias and Dawar [6] showed a corresponding unconditional FPC undefinabiity result. We improve on this with the following.

▶ **Theorem 5.2** (FPC-IS). *For every $\epsilon, \delta > 0$,* GapIS$(1 - \frac{1}{\sqrt{2}} - \delta, \epsilon)$ *is not definable in FPC.*

Here IS is the function problem giving the size of a maximal independent set in a graph as a proportion of the total number of vertices. This is equivalent to the FPC undefinability of GapVertexCover$(\frac{1}{1-\epsilon,\sqrt{2}} + \delta)$, implying the FPC-inapproximability of vertex cover by a factor smaller than $\sqrt{2}$. The theorem follows from the reduction presented in [22, Chapter 5] which can be defined in First-Order Logic using standard methods.

5.3 Graph Colouring

Perhaps the most striking consequence of our result is the following.

▶ **Theorem 5.3.** *For every $t \geq 3$, the class of 3-colourable graphs are not* FPC *separable from those that are not t-colourable.*

Theorem 5.3 should be contrasted with what is known about the NP-hardness of promise graph colouring. It is known that it is NP-hard to separate the 3-colourable graphs from those that are not 5-colourable [7]. It is conjectured that it is NP-hard to separate the 3-colourable graphs from those that are not t-colourable for all $t \geq 3$, but this is open even for $t = 6$. Thus, Theorem 5.3 provides the first significant example of an FPC hardness of approximation result that is open in the classical setting of NP-hardness.

Guruswami and Sandeep [16] show a reduction from Gap$\textsc{Irreg2to2}_{j,q}(1, \delta)$ to the problem of separating 3-colourable graphs from non-t-colourable ones [12]. The reduction is easily definable in first-order logic, proving Theorem 5.3.

6 Conclusion

We have shown that the reductions involved in the proof of the celebrated proof by Khot, Minzer and Safra of the 2-to-2 games theorem can all be implemented as interpretations in first-order logic. This means that the NP-hardness they establish of separating nearly satisfiable instances from highly unsatisfiable ones can be turned into an unconditional inseparability result in FPC. Moreover, the result is achieved with *perfect completeness*: it is impossible to separate with an FPC sentence the fully satisfiable 2-to-2 games from those that are highly unsatisfiable.

From this result we are able to derive a number of consequences, the most striking of which is that it is impossible to separate with an FPC sentence the graphs that are 3-colourable from those that are not t-colourable for any constant t. The NP-hardness of such a separation is only conjectured for values t larger than 5. We also obtain strong FPC undefinability results for approximation of unique games. In terms of approximation ratios these are an improvement over those of Tucker-Foltz [26]. However, the latter results were obtained for *unwieghted* games while ours are for weighted games.

This work suggests a number of further directions to pursue. One is an investigation of the FPC definability of promise constraint satisfaction problems (PCSP). The t-colouring of 3-colourble graphs is one such example, but PCSP are a very active current area of investigation. Our results could also be tightened by showing them for unweighted instances rather than with weights. Indeed, we believe that Theorem 5.1 could be improved to apply to unweighted games as well, making it a direct improvement of the results of [26]. For this improvement, it would be sufficient to prove the FPC analogue for the result of Crescenzi et al. [9] showing a gap reduction from weighted CSP instances to unweighted ones. The proof of Khot, Minzer and Safra applies this reduction to establish Theorem 2.3 on unweighted games. This merits further study.

References

1 Matthew Anderson and Anuj Dawar. On symmetric circuits and fixed-point logics. *Theory of Computing Systems*, 60(3):521–551, July 2017. doi:10.1007/s00224-016-9692-2.

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

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

4 Albert Atserias, Andrei Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009. Automata, Languages and Programming (ICALP 2007). doi:10.1016/j.tcs.2008.12.049.

5 Albert Atserias and Víctor Dalmau. Promise constraint satisfaction and width. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022*, pages 1129–1153. SIAM, 2022. doi:10.1137/1.9781611977073.48.

6 Albert Atserias and Anuj Dawar. Definable inapproximability: New challenges for duplicator, 2019. arXiv:1806.11307.

7 Libor Barto, Jakub Bulín, Andrei Krokhin, and Jakub Opršal. Algebraic approach to promise constraint satisfaction. *Journal of the ACM*, 68(4):1–66, July 2021. doi:10.1145/3457606.

8 Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61(1), January 2014. doi:10.1145/2556646.

9 Pierluigi Crescenzi, Riccardo Silvestri, and Luca Trevisan. On weighted vs unweighted versions of combinatorial optimization problems. *Inf. Comput.*, 167(1):10–26, May 2001. doi:10.1006/inco.2000.3011.

10 Anuj Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, 2(1):8–21, January 2015. doi:10.1145/2728816.2728820.

11 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*, STOC 2018, pages 376–389, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188804.

12 Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. *SIAM Journal on Computing*, 39(3):843–873, January 2009. doi:10.1137/07068062x.

13 Irit Dinur and Shmuel Safra. On the hardness of approximating label-cover. *Information Processing Letters*, 89(5):247–254, March 2004. doi:10.1016/j.ipl.2003.11.007.

14 Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite Model Theory*. Springer, 2nd edition, 1999.

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

16 Venkatesan Guruswami and Sai Sandeep. d-to-1 hardness of coloring 3-colorable graphs with o (1) colors. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

17 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001. doi:10.1145/502090.502098.

18 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 767–775, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.510017.

19 Subhash Khot. On the unique games conjecture (invited survey). In *2010 IEEE 25th Annual Conference on Computational Complexity*, pages 99–121, 2010. doi:10.1109/CCC.2010.19.

20 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, pages 576–589, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055432.

21 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in Grassmann graph have near-perfect expansion. *Annals of Mathematics*, 198(1):1–92, 2023. doi:10.4007/annals.2023.198.1.1.

22 Dor Minzer. *On Monotonicity Testing and the 2-to-2 Games Conjecture*, volume 49. Association for Computing Machinery, New York, NY, USA, 1 edition, 2022.

23 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 245–254, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1374376.1374414.

24 Benjamin Rossman. Equi-rank homomorphism preservation theorem on finite structures. In *33rd EACSL Annual Conference on Computer Science Logic, CSL*, 2025.

25 Khot Subhash, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 592–601, 2018. doi:10.1109/FOCS.2018.00062.

26 Jamie Tucker-Foltz. Inapproximability of Unique Games in Fixed-Point Logic with Counting. *Logical Methods in Computer Science*, Volume 20, Issue 2, April 2024. doi:10.46298/lmcs-20(2:3)2024.

27 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer FhScience*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

Description Complexity of Unary Structures in First-Order Logic with Links to Entropy

Reijo Jaakkola
Mathematics Research Centre, Tampere University, Finland

Antti Kuusisto
Mathematics Research Centre, Tampere University, Finland

Miikka Vilander
Mathematics Research Centre, Tampere University, Finland

Abstract

The description complexity of a model is the length of the shortest formula that defines the model. We study the description complexity of unary structures in first-order logic FO, also drawing links to semantic complexity in the form of entropy. The class of unary structures provides, e.g., a simple way to represent tabular Boolean data sets as relational structures. We define structures with FO-formulas that are strictly linear in the size of the model as opposed to using the naive quadratic ones, and we use arguments based on formula size games to obtain related lower bounds for description complexity. For a typical structure the upper and lower bounds in fact match up to a sublinear term, leading to a precise asymptotic result on the expected description complexity of a randomly selected structure. We then give bounds on the relationship between Shannon entropy and description complexity. We extend this relationship also to Boltzmann entropy by establishing an asymptotic match between the two entropies. Despite the simplicity of unary structures, our arguments require the use of formula size games, Stirling's approximation and Chernoff bounds.

2012 ACM Subject Classification Theory of computation → Finite Model Theory; Mathematics of computing → Information theory

Keywords and phrases formula size, finite model theory, formula size games, entropy, randomness

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.17

Related Version *Full Version*: https://arxiv.org/abs/2406.02108

Funding Antti Kuusisto and Miikka Vilander were supported by the Academy of Finland projects *Explaining AI via Logic* (XAILOG), grant number 345612 and *Theory of computational logics*, grant numbers 352419, 352420, 353027, 324435 and 328987.

1 Introduction

This paper investigates the resources needed to define finite models with a unary relational vocabulary. While unary models are very simple, it turns out that proving limits on the formula sizes for defining them is non-trivial. Furthermore, unary models are important as they give a direct relational representation of Boolean data sets, consisting simply of data points and their properties – thereby providing one of the simplest data representation schemes available. In practice all tabular data can be discretized and modeled via a Boolean data set. This relates to applications in, e.g., explainability and compression.

Given a logic $\mathcal{L}$ and a class $\mathcal{M}$ of models, the *description complexity* $C(\mathfrak{M})$ of a model $\mathfrak{M}$ is the minimum length of a formula $\varphi \in \mathcal{L}$ that defines $\mathfrak{M}$ with respect to $\mathcal{M}$. In the main scenario of this paper, $\mathcal{M}$ is the class of models with the same domain of a finite size n and with the same unary vocabulary τ. We mostly study the setting via first-order logic FO. However, as description complexity links to the themes of *compressibility* and *compression*,

we also investigate the restricted languages FO_d where the quantifier rank of every formula is limited to a positive integer d. This will lead to dramatically shorter description lengths (cf. Section 3) via a natural lossy compression phenomenon.

We also investigate how the *Shannon entropies* of unary structures are linked to their description complexities, the general trend being that higher entropy relates to higher description complexity. Shannon entropy is a well-known measure of intrinsic complexity, or randomness, from information theory. The Shannon entropy of a probability distribution $\mathbb{P} : X \to [0, 1]$ over a finite set X is given by $-\sum_{x \in X} \mathbb{P}(x) \log_2 \mathbb{P}(x)$. A relational structure $\mathfrak{M}$ of size n over a unary vocabulary τ naturally defines a probability distribution over its domain. Indeed, let T be the set of unary quantifier-free *types* over τ, i.e., subsets of τ. A point a of a model $\mathfrak{M}$ *realizes* a type $\pi \subseteq \tau$ if π is the set of relation symbols corresponding to exactly those unary relations that contain the point a. Now a τ-model $\mathfrak{M}$ of size n naturally defines the probability distribution $\mathbb{P} : T \to [0, 1]$ such that $\mathbb{P}(\pi) = \frac{|\pi|}{n}$, where $|\pi|$ is the number of points of $\mathfrak{M}$ realizing the type π. The Shannon entropy of $\mathfrak{M}$ is then naturally defined to be equal to the Shannon entropy of the distribution $\mathbb{P} : T \to [0, 1]$.

While the Shannon entropy of $\mathfrak{M}$ gives an intrinsic measure of complexity (or randomness) of $\mathfrak{M}$, another entropy measure may perhaps be easier to grasp intuitively. *Boltzmann entropy* has its origins in statistical mechanics, and it was originally defined as $k \ln \Omega$, where k is the Boltzmann constant and Ω the number of *microstates* of a system. In our setting, we follow [14] and define Boltzmann entropy of a *model class* $\mathcal{A}$ as $\log_2 |\mathcal{A}|$, thus dropping the Boltzmann constant k, using binary logarithms and associating models with microstates. Now, it is natural to then define the Boltzmann entropy of a model $\mathfrak{M}$ as $\log_2 |\mathcal{M}|$, where $\mathcal{M}$ is the isomorphism class of $\mathfrak{M}$ (recall here that in our setting, all models have the same domain of size n, so $\mathcal{M}$ is finite). The reason why the Boltzmann entropy of $\mathfrak{M}$ is a reasonable measure of intrinsic complexity of $\mathfrak{M}$ is now easy to motivate. Firstly, consider a τ-model $\mathfrak{M}_0$ of size n where each $P \in \tau$ is interpreted as the empty relation. This is a very simple model whose isomorphism class has size 1 and the Boltzmann entropy of $\mathfrak{M}_0$ is thus very low: $\log_2 1 = 0$. On the other hand, models with the predicates in τ distributed in more disordered ways have larger isomorphism classes and thus greater Boltzmann entropies.

1.1 Contributions

Concerning upper bounds on description complexity, we show how to define unary structures via FO-formulas that are linear in model size. This contrasts the standard quadratic formulas that use equalities for counting cardinalities in a naive way. We also give analogous formulas for FO_d with quantifier rank at most d. Concerning lower bounds, we use formula size games to provide bounds with a worst case gap of a constant factor of 2 in relation to the upper bounds. This is done both for full FO and FO_d.

For a random structure the upper and lower bounds in fact match up to a sublinear additive term. Using this, we show that – asymptotically – the expected description complexity of a random unary structure of size n and over the vocabulary τ is exactly $3n/2^{|\tau|}$.

We then turn our attention to entropy. We show a close relationship between the Shannon entropy and Boltzmann entropy of a unary structure. We obtain related upper and lower bounds and thereby also establish the following asymptotic equivalence for *every* sequence $\mathfrak{M}_n$ of models of increasing size n: $H_S(\mathfrak{M}_n) \sim \frac{1}{n} H_B(\mathfrak{M}_n)$. We note that a result bearing a resemblance to this one has been obtained in a slightly different framework in [15].

Finally, we relate the description complexity of a model to its entropy. We investigate the general picture of the relationship by giving upper and lower bounds on the description complexity of a model in terms of its entropy. See Figure 1a for the case of FO and Figure 1b

for FO_d. The bounds allow us to *exclude* a large portion of the (a priori) possible combinations of description complexity and entropy. In particular, we see that models with very high entropy have higher description complexity than models with very low entropy. Moreover, *models with a very low entropy are guaranteed to have a reasonably low description complexity, while models with very high entropies must have a notable description complexity.*

1.2 Related work, techniques and applications

Description complexity is conceptually related to Kolmogorov complexity, and it is also well known that entropy and Kolmogorov complexity are linked. Indeed, for computable distributions, Shannon entropy links to Kolmogorov complexity to within a constant. This is discussed, e.g., in [17, 9, 16]. However, [23] shows that the general link fails for Rényi and Tsallis entropies. See, e.g., [9, 16, 23] for discussions on Rényi and Tsallis entropies.

Concerning work in the intersection of logic and entropy, the recent article [14] by Jaakkola et al. provides related results for a graded modal logic GMLU over Kripke-models with the universal accessibility relation.

They show that the expected Boltzmann entropy of the equivalence classes of GMLU is asymptotically equivalent to the expected description complexity times the vocabulary size. While [14] concerns GMLU, the current paper studies (monadic) FO. Because of the multi-variable nature of FO, this leads to some major differences in the techniques required. The upper bound formulas of the current paper use some clever tricks that are not possible in the modal logic GMLU. Indeed, together with the results of [14], our upper bound formulas show that FO is more succinct than GMLU. Furthermore, the techniques used for the lower bounds for GMLU do not suffice for FO, necessitating new arguments.

Surprisingly, the relationship to entropy also turns out to be different. Indeed, in the case of GMLU, models with maximal entropy have maximal description complexity, while in the case of FO this is no longer the case.

For proving bounds on formula sizes, we use *formula size games* for FO. Indeed, variants of standard Ehrenfeucht-Fraïssé games would not suffice, as we need to deal with formula length, and thereby with all logical operators, including connectives. The formula size game that we use for FO is a slight modification of the game of Hella and Väänänen [10]. The first formula size game, developed by Razborov in [20], dealt with propositional logic. A later variant of the game was defined by Adler and Immerman for CTL in [1]. In [11] the formula size game for modal logic ML was used by Hella and Vilander to establish that bisimulation invariant FO is non-elementarily more succinct than ML. For a further example, we also mention the frame validity games of Balbiani et al. [2]. Recently, Fagin et al. in [7, 8] and Carmosino et al. in [4, 5, 6] have developed and used *multi-structural games* to prove lower bounds on the *number of quantifiers* that are needed for separating two structures in a given logic. In [8] they have also pointed out that strong lower bounds on the number of quantifiers would imply new lower bounds in circuit complexity.

Description complexity is relevant in many applications, one interesting link being data compression. It is natural to consider unary models $\mathfrak{M}$ as data sets to be compressed into corresponding FO-sentences. To give a *simplified* example, let $\mathcal{M}$ be the class of models over the unary alphabet $\tau = \{P, Q\}$ and with domain $M = \{1, \ldots, 10\}$. Let $\mathfrak{M}_1$ be the model where $P^{\mathfrak{M}_1} = Q^{\mathfrak{M}_1} = M$ and $\mathfrak{M}_2$ be the model where $P^{\mathfrak{M}_2} = \{1, 2, 3\}$ and $Q^{\mathfrak{M}_2}$ is, say, $\{3, 4, 5, 6, 7\}$. Now, the simple formula $\forall x(P(x) \wedge Q(x))$ fully defines the model $\mathfrak{M}_1$ with respect to $\mathcal{M}$, while the model $\mathfrak{M}_2$ clearly requires a more complex formula. Suppose then that our models are represented as tabular Boolean data, meaning that each model corresponds to a 0-1-matrix with ten rows (one row for each domain element $m \in M$) and

two columns, one column for P and another one for Q. In this framework, when using FO as a compression language, the Boolean matrix for $\mathfrak{M}_1$ then compresses nicely into the formula $\forall x(P(x) \wedge Q(x))$, while the matrix for $\mathfrak{M}_2$ compresses to a notably more complex formula.

Many of the technical goals in *explainable artificial intelligence* (XAI) relate to compression [22], often revolving around issues of compressing information given by probability distributions. It is natural to expect representations of distributions with very high values of Shannon entropy to be more difficult to compress than ones with very low values. Concerning formula length, recent articles on XAI using minimum length formulas of logics as explanations of longer specifications include, e.g.,[3, 18, 12, 13], and numerous others. For work on using short Boolean formulas as general explanations of real-life data given in the form of unary relational structures (i.e., tabular Boolean data sets), see [13]. In that paper, surprisingly short Boolean formulas are shown to give similar error rates to ones obtained by more sophisticated classifiers, e.g., neural networks and naive Bayesian classifiers.

Concerning further directions in explainability, minimum size descriptions ψ of unary relational models $\mathfrak{M}$ can be useful for finding explanations in the context of the *special explainability problem* [12]. The positive case of this problem amounts to finding formulas χ with a given bound k on length such that $\mathfrak{M} \models \chi \models \varphi$, where φ acts as a classifier. In this context, it often suffices to find a short interpolant χ such that $\psi \models \chi \models \varphi$, where ψ is a minimum description of $\mathfrak{M}$. In applications, this latter task can often be more efficient than the first one, especially when ψ is significantly smaller than $\mathfrak{M}$. One way to ensure ψ is short enough is to describe $\mathfrak{M}$ in a sufficiently incomplete way, such as with FO_d with small d.

Finally, in applications, it is typically easy to compute the Shannon entropy of structures, while description complexity and thereby issues relating to compressibility and explainability are *much more difficult to determine*. Therefore, even a rough picture of the links between entropy and description complexity can be useful.

The plan of the paper is as follows. After the preliminaries in Section 2, we provide upper bounds for the description complexity of unary structures in Section 3. In Section 4 we establish related lower bounds using games. In Section 5 we determine asymptotically the expected description complexity of a random unary structure. In Section 6 we give bounds on the relationship between entropy and description complexity. In Section 7 we conclude.

2 Preliminaries

Let $\tau = \{P_1, \dots, P_k\}$ be a monadic vocabulary and let $Var = \{x_1, x_2, \dots\}$ be a countably infinite set of variables. The syntax of first-order logic $\mathrm{FO}[\tau]$ is generated by the grammar: $\varphi ::= x = y \mid P(x) \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\varphi \mid \forall x\varphi$, where $x, y \in Var$ and $P \in \tau$. The **quantifier rank** of a formula $\varphi \in \mathrm{FO}[\tau]$ is the maximum number of nested quantifiers in the formula. We denote by $\mathrm{FO}_d[\tau]$ the fragment of $\mathrm{FO}[\tau]$ that only includes the formulas with quantifier rank at most d. A formula $\varphi \in \mathrm{FO}[\tau]$ is in **negation normal form** if negations are only applied to **atomic formulas** $x = y$ or $P(x)$. We assume all formulas are in negation normal form and treat the notation $\neg\varphi$ as shorthand for the negation normal form formula obtained from φ by pushing the negation to the level of atomic formulas.

The **size** of a formula $\varphi \in \mathrm{FO}[\tau]$ is defined as the number of atomic formulas, conjunctions, disjunctions and quantifiers in φ. Note that negations do not contribute to the size of φ. This choice together with using negation normal form means that positive and negative atomic information is treated as equal in terms of formula size. In line with this thinking, we will refer also to $x \neq y$ and $\neg P(x)$ as atomic formulas in the sequel.

A formula $\varphi \in \mathrm{FO}[\tau]$ is in **prenex normal form** if it is of the form $Q_1 x_1 \dots Q_m x_m \psi$, where $Q_i \in \{\exists, \forall\}$ for $i \in \{1, \dots, m\}$ and $\psi \in \mathrm{FO}[\tau]$ has no quantifiers. It is well-known that every FO-formula can be transformed into an equivalent formula in prenex normal form which has the same size as the original formula.

A **τ-model** is a tuple $\mathfrak{M} = (M, P_1^{\mathfrak{M}}, \dots, P_k^{\mathfrak{M}})$, where $M = \{1, \dots, n\}$ and $P_i^{\mathfrak{M}} \subseteq M$ for $i \in \{1, \dots, k\}$. A model $\mathfrak{M}$ is a **model of size n** if $|M| = n$. A partial function $s : \mathit{Var} \rightharpoonup M$ is called an **interpretation**. We also call pairs $(\mathfrak{M}, s)$ models and identify the pair $(\mathfrak{M}, \emptyset)$ with the model $\mathfrak{M}$. The truth relation $(\mathfrak{M}, s) \models \varphi$ is defined in the usual way for $\mathrm{FO}[\tau]$.

Let $\mathfrak{M} = (M, P_1^{\mathfrak{M}}, \dots, P_k^{\mathfrak{M}})$ be a τ-model of size n. We say that a formula $\varphi \in \mathrm{FO}[\tau]$ **defines** $\mathfrak{M}$ if for all τ-models $\mathfrak{M}'$ of size n we have $(\mathfrak{M}', \emptyset) \models \varphi$ iff $\mathfrak{M}'$ is isomorphic to $\mathfrak{M}$. As first-order logic cannot distinguish between isomorphic structures, we can in some sense identify the model $\mathfrak{M}$ with the class of models isomorphic to $\mathfrak{M}$. The **description complexity** $C(\mathfrak{M})$ of $\mathfrak{M}$ is the size of the smallest formula in $\mathrm{FO}[\tau]$ that defines $\mathfrak{M}$.

Note that our definition of description complexity concerns separating $\mathfrak{M}$ only from other models of the same size n. Requiring separation from all other models would unduly emphasize the size of the model, making even very simple models have a high description complexity. For example, the model $\mathfrak{M} = (M, P^{\mathfrak{M}})$ of size n, where $P^{\mathfrak{M}} = M$, would already require a formula with size in the order of n. In our setting, $C(\mathfrak{M}) = 2$, because $\mathfrak{M}$ is defined by the formula $\forall x P(x)$.

A **τ-type** π is a subset of τ. A point $a \in M$ **realizes** a τ-type π if for all $P \in \tau$ we have $a \in P^{\mathfrak{M}}$ iff $P \in \pi$. We let $|\pi|_{\mathfrak{M}}$ denote the number of points in $\mathfrak{M}$ realizing π. We often omit the subscript when the model is clear from the context. Note that two τ-models $\mathfrak{M}$ and $\mathfrak{M}'$ are isomorphic iff each type is realized in the same number of points in both models.

We also consider more coarse ways to divide models into classes than isomorphism. For each positive integer d we can define an equivalence relation $\equiv_d$ over τ-models of size n as follows. Given two τ-models $\mathfrak{M}$ and $\mathfrak{M}'$ of size n, we define that $\mathfrak{M} \equiv_d \mathfrak{M}'$ iff for each τ-type π with $|\pi|_{\mathfrak{M}} < d$, we have that $|\pi|_{\mathfrak{M}} = |\pi|_{\mathfrak{M}'}$. In other words, $\mathfrak{M} \equiv_d \mathfrak{M}'$ iff each type that is realized in less than d points in $\mathfrak{M}$ is realized in the same number of points in both models. It is easy to show that $\mathfrak{M} \equiv_d \mathfrak{M}'$ iff they satisfy the same sentences of $\mathrm{FO}_d[\tau]$. The **d-description complexity** $C_d(\mathfrak{M})$ of a τ-model $\mathfrak{M}$ is the size of the smallest $\mathrm{FO}_d[\tau]$-formula that defines the equivalence class of $\mathfrak{M}$ in $\equiv_d$.

To characterize model classes, we use tuples with $t = 2^{|\tau|}$ numbers. For an isomorphism class, the tuple is simply $(|\pi_1|, \dots, |\pi_t|)$. For an equivalence class $\mathcal{M}$ of $\equiv_d$, we only use numbers up to d. For a tuple $\overline{m} = (m_1, \dots, m_t)$, if $m_i = d$, then there are at least d realizing points of type π_i in models of the class $\mathcal{M}$. If $m_i < d$, then each model has exactly m_i points realizing the type π_i. The notation $\mathcal{M}_{\overline{m}}$ refers to classes of $\equiv_d$ via these tuples. The tuples that correspond to some class of $\equiv_d$ are characterized by the conditions $m_i \leq d$ for $i \in \{1, \dots, t\}$, $\sum_{i=1}^{t} m_i \leq n$ and if $\sum_{i=1}^{t} m_i < n$, then $m_j = d$ for some $j \in \{1, \dots, t\}$. If $\sum_{i=1}^{t} m_i = n$, then $\mathcal{M}_{\overline{m}}$ is an isomorphism class.

Since τ-types partition the points of a τ-model $\mathfrak{M}$, we may consider a natural probability distribution over the types in $\mathfrak{M}$. The probability p_π of a type π is simply $|\pi|/n$, that is, the probability of hitting a point of type π when selecting a point from $\mathfrak{M}$ randomly. The **Shannon entropy** of $\mathfrak{M}$ is the quantity $H_S(\mathfrak{M}) := \sum_{i=1}^{t} -p_{\pi_i} \log(p_{\pi_i}) = \sum_{i=1}^{t} -\frac{|\pi_i|}{n} \log\left(\frac{|\pi_i|}{n}\right)$. Here we follow the convention $0 \log(0) = 0$. Shannon entropy is an information theoretic way of measuring randomness of probability distributions. Uniform distributions have maximal Shannon entropy, as the uncertainty of the outcome of choosing a random point is maximized. Conversely, for a distribution that places all of the probability mass on a single event, Shannon

entropy is zero. Hence, a model realizing each type the same number of times (or as close as possible) has maximal Shannon entropy, while for a model that realizes only a single type Shannon entropy is zero.

Another way to define entropy of a model $\mathfrak{M}$ uses the model class $\mathfrak{M}$ belongs to. Given an equivalence relation $\equiv$ over models of size n (and thus domain $\{1,\dots,n\}$), the **Boltzmann entropy** of $\mathfrak{M}$ with respect to $\equiv$ is $H_B(\mathfrak{M}) := \log(|\mathcal{M}|)$, where $\mathcal{M}$ is the equivalence class of $\mathfrak{M}$. In this paper the equivalence relation $\equiv$ is either isomorphism in the case of full FO or $\equiv_d$ for FO_d. For isomorphism, we write $H_B(\mathfrak{M})$ and for $\equiv_d$ we write $H_B^d(\mathfrak{M})$.

Boltzmann entropy originates from statistical mechanics, where it measures the randomness of a macrostate (= a model class) via the number of microstates (= models) that correspond to it. The idea is that a larger macrostate is "more random" (or "less specific") since it is more likely to be hit by a random selection. We show in Section 6 that $H_S(\mathfrak{M}) \sim \frac{1}{n} H_B(\mathfrak{M})$, where n is the size of the domain of $\mathfrak{M}$. Thus the two notions of entropy are asymptotically equivalent up to normalization. This shows that both entropies indeed measure the randomness of a model from different points of view.

3 Upper bound formulas

In this section we define arbitrary τ-models via formulas of size linear in the size of the model. Recall that defining a model means separating it from all non-isomorphic models with the same domain size. To see why linear size formulas are quite succinct, note that the following naive formula $\bigwedge_{\ell=1}^{2^{|\tau|}} \exists x_1 \dots \exists x_{|\pi_\ell|} \left(\bigwedge_{i=1}^{|\pi_\ell|} \pi_\ell(x_i) \wedge \bigwedge_{j=i+1}^{|\pi_\ell|} x_i \neq x_j \right)$, which expresses that for each $1 \leq \ell \leq 2^{|\tau|}$ the type π_ℓ is realized by at least $|\pi_\ell|$ distinct points, is of quadratic size in the size n of the model.

For clean results on formula size, we define a constant $c_\tau := 15|\tau|2^{|\tau|}$. Note that we consider c_τ to be constant as it only depends on the size of the alphabet τ, which in our context is constant.

▶ **Theorem 1.** *Let $\mathfrak{M}$ be a model of size n. Let $T = \{\pi_1, \dots, \pi_\ell\}$ be the types realized in $\mathfrak{M}$, enumerated in ascending order of numbers of realizing points. Now we have the bound $C(\mathfrak{M}) \leq \min(3|\pi_\ell| + c_\tau, 6|\pi_{\ell-1}| + c_\tau)$.*

Proof. We obtain two different upper bound formulas. Due to lack of space, we only give one of them in full here; see A.1 for details on the second formula.

We begin with an easy formula we use extensively below. For a type π and $x \in \mathit{Var}$, let

$$\pi(x) := \bigwedge_{P \in \pi} P(x) \wedge \bigwedge_{P \notin \pi} \neg P(x).$$

The formula $\pi(x)$ states that the point x realizes the type π.

Let $T = \{\pi_1, \dots, \pi_\ell\}$ be a set of τ-types and let $\overline{m}$ be a sequence of $r \leq \ell$ positive integers with $0 < m_1 \leq \dots \leq m_r$. Let $\mathfrak{M}$ be a model of size n, where exactly the types in T are realized. We will make sure of this with a separate formula later. The formula $\varphi(T, \overline{m})$ below is satisfied by such a model $\mathfrak{M}$ if and only if for every $i \in \{1, \dots, r\}$, the model $\mathfrak{M}$ has *at least* m_i points that realize the type π_i. Note that we do not assert anything about the types $\pi_{r+1}, \dots, \pi_\ell$, but we still need to mention them in the formula. We define

$$\psi_{m_r} := y \neq x_{m_r - 1} \wedge \bigvee_{\substack{j \in \{1,\dots,r\} \\ m_j = m_r}} (\pi_j(x_1) \wedge \pi_j(y))$$

$\psi_i := y \neq x_{i-1} \wedge \psi_{i+1}$, if $m_j \neq i$ for all $j \in \{1, \dots, r\}$, and

$\psi_i := y \neq x_{i-1} \wedge (\bigvee_{\substack{j \in \{1,\dots,r\} \\ m_j = i}} (\pi_j(x_1) \wedge \pi_j(y)) \vee \psi_{i+1})$, otherwise.

$\psi_1 := \psi_2$, if $m_j \neq 1$ for all $j \in \{1, \dots, r\}$, and

$\psi_1 := \bigvee_{\substack{j \in \{1,\dots,r\} \\ m_j = 1}} \pi_j(x_1) \vee \psi_2$, otherwise.

$\varphi(T, \overline{m}) := \forall x_1 \dots \forall x_{m_r - 1} \exists y (\bigvee_{j \in \{r+1,\dots,\ell\}} \pi_j(x_1) \vee \psi_1)$

We proceed with an explanation of how the formula $\varphi(T, \overline{m})$ works. We assume that precisely the types in T are realized in the model $\mathfrak{M}$ to be evaluated, so we know that the first universal variable x_1 is always attached to a point that realizes one of the types in T. The formula first checks if x_1 realizes one of the types $\pi_{r+1}, \dots, \pi_\ell$ that we wish to ignore. The recursion then handles the rest of the types, starting with the smallest ones. If the type π_j of x_1 has $m_j = 1$, nothing further is stated as we already know the type is realized in $\mathfrak{M}$ by our assumption.

Now, consider a type π_j with, say, $m_j = 5$. Up to the subformula ψ_5, the recursion of our formula has insisted that $y \neq x_i$ for $i \in \{1, 2, 3, 4\}$. Note that the formula does not contain any atomic formulas $x_{i_1} \neq x_{i_2}$. The crucial point is that since the variables $x_1, \dots, x_4$ are universally quantified, the existence of y must hold also in the case, where $x_1, \dots x_4$ happen to all be different points of the same type π_j. If the evaluated model $\mathfrak{M}$ has at least 5 points that realize π_j, then the formula holds as another point y that realizes π_j can be found. If, however, $\mathfrak{M}$ has only 4 points that realize π_j, then one of the universally quantified tuples includes precisely those 4 points and another y of the same type cannot be found.

We adopt the notation $k = |\tau|$ and compute the size of $\varphi(T, \overline{m})$. The formula has m_r quantifiers. For each type $\pi \in T$, there are at most two occurrences of the subformula $\pi(x)$ (with different variables x). Each subformula $\pi(x)$ contains k atomic formulas. Thus there are at most $2k|T|$ atomic formulas of the form $P(x)$ or $\neg P(x)$. Each inequality $y \neq x_i$ for $1 \leq i \leq m_r - 1$ occurs exactly once, so there are $m_r - 1$ atomic formulas that are equalities or inequalities. Finally we multiply the number of atomic formulas by two and subtract one to also account for the binary connectives. The size of $\varphi(T, \overline{m})$ is thus at most

$$m_r + 2(m_r - 1 + 2k|T|) - 1 = 3m_r + 4k|T| - 3.$$

We proceed to define our first complete upper bound formula that defines an isomorphism class of models. Let $\mathfrak{M}$ be a τ-model with domain $M = \{1, \dots, n\}$. Let $T = \{\pi_1, \dots, \pi_\ell\}$ be the set of τ-types realized in $\mathfrak{M}$ and let $\overline{m} = (|\pi_1|, \dots, |\pi_\ell|)$. Assume further that $\overline{m}$ is increasing. The full formula $\varphi(\mathfrak{M})$ is based on bounding the size of every type in T from below, thus separating it from all non-isomorphic models with the same domain size.

$$\varphi(\mathfrak{M}) := \bigwedge_{i=1}^{\ell} \exists x \, \pi_i(x) \wedge \forall x \bigvee_{i=1}^{\ell} \pi_i(x) \wedge \varphi(T, \overline{m})$$

In addition to the size of $\varphi(T, \overline{m})$ computed above, $\varphi(\mathfrak{M})$ includes $|T| + 1$ quantifiers and two occurrences of $\pi(x)$ for each type $\pi \in T$, resulting in $2k|T|$ atomic formulas. Accounting for the added binary connectives, the size of $\varphi(\mathfrak{M})$ is thus at most

$$|T| + 1 + 2 \cdot 2k|T| + 3|\pi_\ell| + 4k|T| - 3 = 3|\pi_\ell| + 8k|T| + |T| - 2 \leq 3|\pi_\ell| + c_\tau.$$

The second formula $\psi(\mathfrak{M})$ of size at most $6|\pi_{\ell-1}| + c_\tau$ states that each type π_i with $i \neq \ell$ has exactly $|\pi_i|$ points. See A.1 for details. Both formulas define any model $\mathfrak{M}$ so we can always use whichever is smaller, thus proving the claim. ◀

▶ **Corollary 2.** *Let $\mathfrak{M}$ be a model of size n. Now $C(\mathfrak{M}) \leq 2n + c_\tau$.*

Proof. A model $\mathfrak{M}$ corresponding to the tuple $(0, \ldots, 0, n/3, 2n/3)$ maximises the value of the expression $\min(3|\pi_\ell| + c_\tau, 6|\pi_{\ell-1}| + c_\tau)$, getting the value $2n + c_\tau$. ◀

We now consider defining equivalence classes of $\equiv_d$. Recall that an equivalence class of $\equiv_d$ corresponds to a tuple $\overline{m} = (m_1, \ldots, m_t)$, where $t = 2^{|\tau|}$, $m_i \leq d$ for all $i \in \{1, \ldots, t\}$, $\sum_{i=1}^{t} m_i \leq n$ and if $\sum_{i=1}^{t} m_i < n$, then $m_j = d$ for some $j \in \{1, \ldots, t\}$.

▶ **Theorem 3.** *Let $\mathfrak{M}$ be a τ-model of size n. Let $\mathcal{M}_{\overline{m}}$ be the equivalence class of $\mathfrak{M}$ in $\equiv_d$, where $\overline{m} = (m_1, \ldots, m_t)$ is the corresponding tuple with the numbers in ascending order. Let m_r be the highest number in $\overline{m}$ below d. Now $C_d(\mathfrak{M}) \leq 3d + 3m_r + c_\tau$. Additionally, if $m_{t-1} < d$, then $C_d(\mathfrak{M}) \leq 6m_{t-1} + c_\tau$.*

Proof. We use the same subformulas from Theorem 1 to obtain two linear size formulas. See A.2 for details. The first formula of size $3d + 3m_r + c_\tau$ works for any tuple $\overline{m}$ and states that each type π_i has exactly m_i points if $m_i < d$ and at least d points if $m_i = d$. The second formula of size $6m_{t-1} + c_\tau$ states that each type π_i with $i \neq t$ has exactly m_i points and works only if all types except possibly π_t have less than d points. ◀

Note that since $m_r < d$, we have $6m_r < 3d + 3m_r$ so the bound for the special case is tighter than the general one. While we must use the more general bound for any $\overline{m}$ with at least two instances of d, the tighter bound is significantly better for small classes with only one instance of d in their tuple. For example, the class with the tuple $(0, \ldots, 0, 1, d)$ gets an upper bound of $6 + c_\tau$ regardless of the number d. At the other extreme, the class with the tuple $(0, \ldots, 0, d-1, d, d)$ gets an upper bound of $3d + 3(d-1) + c_\tau = 6d - 3 + c_\tau$.

We again directly obtain a global upper bound on description complexity.

▶ **Corollary 4.** *Let $\mathfrak{M}$ be a τ-model of size n. Now $C_d(\mathfrak{M}) \leq 6d - 3 + c_\tau$.*

4 Lower bounds via formula size games

In this section, we show lower bounds that match the upper bounds of Section 3 up to a factor of 2. We use the formula size game for first-order logic defined in [10]. We modify the game slightly to correspond to formulas in prenex normal form as this form does not affect the size of the formula. In addition, we introduce a second resource parameter q that corresponds to the number of quantifiers in the separating formula. The game consists of two phases: a quantifier phase, where only $\exists$-moves and $\forall$-moves can be made by S, and an atomic phase, where only $\vee$-moves, $\wedge$-moves and atomic moves can be made. Before the definition of the game, we define some notation.

Let $\mathcal{A}$ be a set of τ-models and let $\varphi \in \mathrm{FO}[\tau]$. We denote $\mathcal{A} \models \varphi$ to mean $(\mathfrak{M}, s) \models \varphi$ for all $(\mathfrak{M}, s) \in \mathcal{A}$. Similarly, we denote $\mathcal{A} \models \neg\varphi$ to mean $(\mathfrak{M}, s) \not\models \varphi$ for all $(\mathfrak{M}, s) \in \mathcal{A}$.

For an interpretation s, a point $a \in M$ and a variable $x \in \mathit{Var}$, we denote by $s[a/x]$ the interpretation s' such that $s'(x) = a$ and $s'(y) = s(y)$ for all $y \in \mathrm{dom}(s)$, $y \neq x$. Let $\mathcal{A}$ be a set of τ-models with the same domain M and let $f : \mathcal{A} \to M$ be a function. We denote by $\mathcal{A}[f/x]$ the set $\{(\mathfrak{M}, s[f(\mathfrak{M}, s)/x]) \mid (\mathfrak{M}, s) \in \mathcal{A}\}$. Intuitively, the function f gives the

new interpretation of the variable x for each model $(\mathfrak{M}, s) \in \mathcal{A}$. Additionally, we denote $\mathcal{A}[M/x] := \{(\mathfrak{M}, s[a/x]) \mid (\mathfrak{M}, s) \in \mathcal{A},\ a \in M\}$. Here the variable x is given all possible interpretations, usually leading to a larger set of models. We next define the game.

Let $\mathcal{A}_0$ and $\mathcal{B}_0$ be sets of τ-models and let $r_0, q_0 \in \mathbb{N}$ with $r_0 > q_0$. The FO **prenex formula size game** $\mathrm{FS}^\tau(r_0, q_0, \mathcal{A}_0, \mathcal{B}_0)$ has two players: Samson (S) and Delilah (D). Positions of the game are of the form $(r, q, \mathcal{A}, \mathcal{B})$, where $r, q \in \mathbb{N}$ and $\mathcal{A}$ and $\mathcal{B}$ are sets of τ-models. The starting position is $(r_0, q_0, \mathcal{A}_0, \mathcal{B}_0)$. In a position $(r, q, \mathcal{A}, \mathcal{B})$, if $r = 0$, then the game ends and D wins. Otherwise, if $q > 0$, the game is said to be in the **quantifier phase** and S can choose from the following three moves:

- **$\exists$-move:** S chooses $f : \mathcal{A} \to M$ and $x_i \in Var$. The new position is $(r-1, q-1, \mathcal{A}[f/x_i], \mathcal{B}[M/x_i])$.
- **$\forall$-move:** The same as the $\exists$-move with the roles of $\mathcal{A}$ and $\mathcal{B}$ switched.
- **Phase change:** S moves on to the atomic phase and the new position is $(r, 0, \mathcal{A}, \mathcal{B})$.

In a position $(r, q, \mathcal{A}, \mathcal{B})$, if $q = 0$, the game is said to be in the **atomic phase** and S can choose from the following three moves:

- **$\wedge$-move:** S chooses $r_1, r_2 \in \mathbb{N}$ and $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathcal{B}$ such that $r_1 + r_2 + 1 = r$ and $\mathcal{B}_1 \cup \mathcal{B}_2 = \mathcal{B}$. Then D chooses the next position from the options $(r_1, 0, \mathcal{A}, \mathcal{B}_1)$ and $(r_2, 0, \mathcal{A}, \mathcal{B}_2)$.
- **$\vee$-move:** The same as the $\wedge$-move with the roles of $\mathcal{A}$ and $\mathcal{B}$ switched.
- **Atomic move:** S chooses an atomic formula α. The game ends. If $\mathcal{A} \models \alpha$ and $\mathcal{B} \models \neg\alpha$, then S wins. Otherwise, D wins.

The prenex formula size game characterizes separation of model classes with formulas of limited size in the following way.

▶ **Theorem 5.** *Let $\mathcal{A}_0$ and $\mathcal{B}_0$ be sets of τ-models and let $r_0, q_0 \in \mathbb{N}$ with $r_0 > q_0$. The following are equivalent*

1. *S has a winning strategy in the game $\mathrm{FS}^\tau(r_0, q_0, \mathcal{A}_0, \mathcal{B}_0)$,*
2. *there is an $\mathrm{FO}[\tau]$-formula φ in prenex normal form with size at most r_0 and at most q_0 quantifiers such that $\mathcal{A}_0 \models \varphi$ and $\mathcal{B}_0 \models \neg\varphi$,*
3. *there is an $\mathrm{FO}[\tau]$-formula φ with size at most r_0 and at most q_0 quantifiers such that $\mathcal{A}_0 \models \varphi$ and $\mathcal{B}_0 \models \neg\varphi$.*

Proof. For the simple inductive proof on how the game works, see [10]. The slight modifications of the separate parameter q for quantifiers and prenex normal form do not change the proof in any meaningful way so we omit it. For the equivalence between the second and third item, note that transforming a formula into prenex form and renaming variables as needed, does not increase its size in full FO with no restrictions on, say, the number of variables. ◀

We take a moment to build some intuition on the formula size game. The role of player S is to show that the model sets $\mathcal{A}_0$ and $\mathcal{B}_0$ can be separated by some FO formula with restrictions on size and number of quantifiers. To achieve this, S starts building the supposedly separating formula, starting from the quantifiers.

Each move of the game corresponds to an operator or atomic formula. When making a move, S makes choices for each model that reflect how that particular model is going to satisfy the formula, in the case of models in $\mathcal{A}$, or not satisfy it, in the case of models in $\mathcal{B}$. For example, for an $\exists$-move, S must choose for each model in $\mathcal{A}$ the point to quantify. This is done via the function f. For a $\wedge$-move, S chooses for each model in $\mathcal{B}$ one of the conjuncts, asserting that the model will not satisfy that conjunct.

The resources r_0 and q_0 restrict the moves of S. He can only make at most q_0 quantifier moves in the quantifier phase of the game. The resource r_0 limits the size of the entire separating formula, including the quantifiers. In the atomic phase, for $\wedge$-moves, S must

divide the remaining resource r between the two conjuncts. It is then the role of D to choose the conjunct she thinks cannot be completed in such a way that the models present are separated. Once D has chosen a conjunct, the other conjunct not chosen is discarded for the rest of the game. Thus, the entire separating formula need not be constructed.

We move on to our lower bounds. Let $\mathfrak{M}$ be a τ-model with domain $M = \{1, \dots, n\}$ and let $T = \{\pi_1, \dots, \pi_\ell\}$ be the types realized in $\mathfrak{M}$, enumerated in ascending order of numbers of realizing points, like in the previous section. We assume that $\ell \geq 2$ as a model, where all points are of the same type, is easily defined by a constant-sized formula. We use the formula size game to show a lower bound of the order $3|\pi_{\ell-1}|$ for the description complexity of $\mathfrak{M}$.

Let $\mathfrak{M}'$ be the model obtained from $\mathfrak{M}$ by changing the type of one point from $\pi_{\ell-1}$ to π_ℓ. We define $\mathcal{A}_0 = \{(\mathfrak{M}, \emptyset)\}$ and $\mathcal{B}_0 = \{(\mathfrak{M}', \emptyset)\}$. We will show that separating the sets $\mathcal{A}_0$ and $\mathcal{B}_0$ requires a formula of size at least $3|\pi_{\ell-1}| - 3$. We begin with an easy lemma on the number of quantifiers required to separate $\mathcal{A}_0$ from $\mathcal{B}_0$.

▶ **Lemma 6.** *If φ separates $\mathcal{A}_0$ from $\mathcal{B}_0$, then φ has at least $|\pi_{\ell-1}|$ quantifiers.*

Proof. Let $r_0 > |\pi_{\ell-1}| - 1$. We show that D has a winning strategy for the formula size game $\mathrm{FS}^\tau(r_0, |\pi_{\ell-1}| - 1, \mathcal{A}_0, \mathcal{B}_0)$. By Theorem 5, this proves the claim.

We show that in any position of such a game, there is a pair $(\mathfrak{M}, s) \in \mathcal{A}$ and $(\mathfrak{M}', s') \in \mathcal{B}$ of models that cannot be separated by any atomic formula. At the starting position, the single models in $\mathcal{A}_0$ and $\mathcal{B}_0$ are such a pair as no variables have been quantified. We proceed to show that D can maintain this pair of models through any move of S. We only treat one of each pair of dual moves as the other is handled the same way.

∃-move: S chooses a function $f : \mathcal{A} \to M$. We focus on the point $a = f(\mathfrak{M}, s)$ chosen for the model $(\mathfrak{M}, s) \in \mathcal{A}$. On the other side, copies of $(\mathfrak{M}', s') \in \mathcal{B}$ are generated for each point $b \in M$, but we restrict attention to only one as follows. If there is a previously quantified variable x with $s(x) = a$, then we choose $b = s'(x)$. Otherwise we choose a new point b of the same type as a. If the type of a is π_i with $i < \ell - 1$, then $\mathfrak{M}$ and $\mathfrak{M}'$ have the same points of type π_i so we may choose $b = a$. If $i \in \{\ell - 1, \ell\}$, then both $\mathfrak{M}$ and $\mathfrak{M}'$ have at least $|\pi_{\ell-1}| - 1$ points of the type π_i so we may choose a fresh b of the same type. The new pair of models found in this manner is clearly atomic-equivalent.

Phase change: With no changes to the sets of models $\mathcal{A}$ and $\mathcal{B}$, the important pair of models is still clearly present in the next position.

∧-move: S chooses splits $r_1 + r_2 + 1 = r$ and $\mathcal{B}_1 \cup \mathcal{B}_2 = \mathcal{B}$. Now the model $(\mathfrak{M}', s') \in \mathcal{B}$ is in $\mathcal{B}_1$ or $\mathcal{B}_2$ and $\mathcal{A}$ remains unchanged. Thus our model pair is present in one of the positions $(r_1, 0, \mathcal{A}, \mathcal{B}_1)$ and $(r_2, 0, \mathcal{A}, \mathcal{B}_2)$. By choosing such a position, D maintains the pair of models.

Atomic move: The model pair is atomic-equivalent, so D wins after any atomic move. ◀

The next lemma concerns the atomic phase. We show that if the number of different atomic formulas required to separate the model sets $\mathcal{A}$ and $\mathcal{B}$ is too large, D wins the game.

▶ **Lemma 7.** *In a game $\mathrm{FS}^\tau(r_0, q_0, \mathcal{A}_0, \mathcal{B}_0)$, let $(r, 0, \mathcal{A}, \mathcal{B})$ be the first position of the atomic phase and let Γ be a minimum size set of atomic formulas such that for every $(\mathfrak{M}, s) \in \mathcal{A}$ and $(\mathfrak{M}', s') \in \mathcal{B}$, there is $\alpha \in \Gamma$ with $(\mathfrak{M}, s) \models \alpha$ and $(\mathfrak{M}', s') \not\models \alpha$. If $r < 2|\Gamma| - 1$, then D has a winning strategy from the position $(r, 0, \mathcal{A}, \mathcal{B})$.*

Proof. We show that every move of S either ends the game in a win for D, or maintains the condition $r < 2|\Gamma| - 1$. Assume this condition holds in position $(r, 0, \mathcal{A}, \mathcal{B})$.

Atomic move: S chooses an atomic formula α. Since $1 \leq r < 2|\Gamma| - 1$, we have $|\Gamma| \geq 2$ so the single atomic formula α does not separate $\mathcal{A}$ from $\mathcal{B}$ and D wins.

∧-move: S chooses splits $r_1 + r_2 + 1 = r$ and $\mathcal{B}_1 \cup \mathcal{B}_2 = \mathcal{B}$. Assume for contradiction that there are sets Γ_1 and Γ_2 of atomic formulas such that Γ_i separates $\mathcal{A}$ from $\mathcal{B}_i$ and $r_i \geq 2|\Gamma_i| - 1$. Now for every pair of models $(\mathfrak{M}, s) \in \mathcal{A}$ and $(\mathfrak{M}', s') \in \mathcal{B}$ we have $(\mathfrak{M}', s') \in \mathcal{B}_1$ or $(\mathfrak{M}', s') \in \mathcal{B}_2$ so the set $\Gamma_1 \cup \Gamma_2$ separates $\mathcal{A}$ from $\mathcal{B}$. Recalling that Γ is a separating set of minimum size and $r < 2|\Gamma| - 1$, we also have $r < 2|\Gamma_1 \cup \Gamma_2| - 1 \leq 2(|\Gamma_1| + |\Gamma_2|) - 1 \leq r_1 + r_2 + 1 = r$, which is a contradiction. Thus we have $r_1 < 2|\Gamma_1| - 1$ or $r_2 < 2|\Gamma_2| - 1$. By choosing the correct position D can maintain the required condition.

∨-move: Identical to the ∧-move with the roles of $\mathcal{A}$ and $\mathcal{B}$ switched. ◀

We are now ready for the main theorem of this section.

▶ **Theorem 8.** *Let $\mathfrak{M}$ be a model of size n. Let $T = \{\pi_1, \ldots, \pi_\ell\}$ be the types realized in $\mathfrak{M}$, enumerated in ascending order of numbers of realizing points, where $\ell \geq 2$. Now $C(\mathfrak{M}) \geq 3|\pi_{\ell-1}| - 3$.*

Proof. We begin with a definition. Let Γ be a set of atomic FO-formulas. We denote the set of variables occurring in formulas of Γ by $V(\Gamma)$. We define the **variable graph of Γ** as $G(\Gamma) = (V(\Gamma), E(\Gamma))$, where $(x, y) \in E(\Gamma)$ iff $x = y \in \Gamma$ or $x \neq y \in \Gamma$. We say that $\Delta \subseteq \Gamma$ is a **connected component of Γ** if $G(\Delta)$ is a maximal connected subgraph of $G(\Gamma)$.

For convenience, we denote here $m := |\pi_{\ell-1}|$. Consider a formula size game $\mathrm{FS}^\tau(3m - 4, q_0, \mathcal{A}_0, \mathcal{B}_0)$. We show that D has a winning strategy for this game, thus proving the claim by Theorem 5. By Lemma 6 we see that to have a chance of winning, S must begin the game with at least m quantifiers. We then move on to the first position $(r, 0, \mathcal{A}, \mathcal{B})$ of the atomic phase, where $r \leq 2m - 4$. Let Γ be a set of atomic formulas such that for every $(\mathfrak{M}, s) \in \mathcal{A}$ and $(\mathfrak{M}', s') \in \mathcal{B}$, there is $\alpha \in \Gamma$ such that $(\mathfrak{M}, s) \models \alpha$ and $(\mathfrak{M}', s') \not\models \alpha$. If $|\Gamma| \geq m - 1$ for every such Γ, then $r \leq 2m - 4 = 2(m - 1) - 2 < 2|\Gamma| - 1$ so D has a winning strategy by Lemma 7. We now assume for contradiction that there exists such a Γ with $|\Gamma| \leq m - 2$.

Consider the connected components Δ of Γ. Since a connected graph with k edges has at most $k + 1$ vertices, for every Δ at most $m - 1$ variables occur in the formulas of Δ.

We now explain why there is a *single* pair of models $(\mathfrak{M}, s) \in \mathcal{A}$ and $(\mathfrak{M}', s') \in \mathcal{B}$ such that they are atomic equivalent with respect to the variables in $V(\Delta)$ for *every* connected component Δ of Γ. We consider the quantifier moves S made in the quantifier phase in the order the moves were made. For every variable x used in a $\exists$-move, we consider Δ such that $x \in V(\Delta)$. We proceed as in the proof of Lemma 6, with respect to only the variables in $V(\Delta)$. That is, if there is a previously quantified variable $y \in V(\Delta)$ such that $s(y) = s(x)$, we choose the opposing model where $s'(x) = s'(y)$. Otherwise, we choose a point with no variables of $V(\Delta)$ attached. Each Δ uses at most $m - 1$ variables so we do not run out of fresh points of any type. The same protocol works for $\forall$-moves as well.

Note that the choices of models are made based on the connected component Δ of x, completely independently of other components. Since every variable x is in exactly one component Δ, this means that the resulting pair of models is simultaneously atomic equivalent with regards to each component separately. Thus this model pair cannot be separated by any atomic formula in Γ. This contradiction with the definition of Γ proves the claim. ◀

We now consider lower bounds in the setting of FO_d. Recall that an equivalence class of $\equiv_d$ is characterized by a tuple $(m_1, \ldots, m_t)$, where $t = 2^{|\tau|}$, $m_i \leq d$, $\sum_{i=1}^t m_i \leq n$ and if $\sum_{i=1}^t m_i < n$, then $m_j = d$ for some j. Let $\overline{m} = (m_1, \ldots, m_t)$ be such a tuple in ascending order of the numbers m_i. If $\sum_{i=1}^t m_i = n$, then $\overline{m}$ corresponds to an isomorphism class and the lower bounds above work as is. Thus we assume that $\sum_{i=1}^t m_i < n$ and consequently $m_t = d$. By taking a model $\mathfrak{M}$ in the equivalence class $\mathcal{M}_{\overline{m}}$ with a maximal number of

points of the type π_t, we can directly obtain the model $\mathfrak{M}'$ as above and get a lower bound on defining the class $\mathcal{M}_{\overline{m}}$ in full FO. This bound directly extends also to FO_d, as limiting quantifier rank gives no advantage in terms of formula size.

▶ **Corollary 9.** *Let $\mathcal{M}_{\overline{m}}$ be an equivalence class of $\equiv_d$, where $\overline{m} = (m_1, \ldots, m_t)$ is the corresponding tuple with the numbers in ascending order. Now $C(\mathcal{M}_{\overline{m}}) \geq 3m_{t-1} - 3$.*

5 Expected description complexity

Using Theorems 1 and 8, we can determine asymptotically the expected description complexity of a *random* τ-model. Here by random we mean that the model is sampled uniformly at random from the set of all τ-models of size n. That is, we determine the asymptotic behavior of the quantity $\mathbb{E}_n[C] := \frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M}} C(\mathfrak{M})$ as $n \to \infty$, where the sum is taken over all the τ-models $\mathfrak{M}$ of size n.

We say that a τ-model $\mathfrak{M}$ is **balanced**, if for every τ-type π, we have $||\pi|_{\mathfrak{M}} - \frac{n}{2^{|\tau|}}| = o(n)$. In other words, a model is balanced if every type is realized roughly the same number of times, allowing for a sublinear discrepancy. We use the well-known Chernoff bounds to establish that a random model is very likely balanced.

▶ **Proposition 10** (Multiplicative Chernoff bound)**.** *Let $X := \sum_{i=1}^{n} X_i$ be a sum of independent 0-1-valued random variables, where $X_i = 1$ with probability p and $X_i = 0$ with probability $1 - p$. Let $\mu := \mathbb{E}[X]$. Now, for every $0 \leq \delta < 1$ we have that $\Pr[|X - \mu| \geq \delta\mu] \leq 2e^{-\delta^2\mu/3}$*

Proof. See for example Corollary 4.6 in [19]. ◀

▶ **Lemma 11.** *The probability that a random τ-model of size n is balanced is at least $1 - 2^{|\tau|+1}/n$.*

Proof. A routine calculation using Proposition 10. See A.3 for details. ◀

The previous lemma gives a rough characterization of random τ-models. Using this characterization together with Theorem 8 we can determine asymptotically the expected description complexity of a random τ-model.

▶ **Theorem 12.** $\mathbb{E}_n[C] \sim \frac{3n}{2^{|\tau|}}$

Proof. To give an upper bound on $\mathbb{E}_n[C]$ we first rewrite it as follows:

$$\mathbb{E}_n[C] = \frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ balanced}} C(\mathfrak{M}) + \frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ not balanced}} C(\mathfrak{M}) \tag{1}$$

Using Corollary 2 and Lemma 11 we see that

$$\frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ not balanced}} C(\mathfrak{M}) \leq \frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ not balanced}} 2n + c_\tau = \Pr[\mathfrak{M} \text{ is not balanced}] \cdot (2n + c_\tau)$$
$$\leq \frac{2^{|\tau|+1}}{n} \cdot (2n + c_\tau) = 2^{|\tau|+2} + \frac{c_\tau 2^{|\tau|+1}}{n} = \mathcal{O}(1).$$

Since we are interested in the asymptotic behavior of $\mathbb{E}_n[C]$, the above shows that we can safely concentrate on the first sum in Equation (1). Using Theorems 1 and 8 we see that if $\mathfrak{M}$ is balanced, then $\frac{3n}{2^{|\tau|}} - o(n) \leq C(\mathfrak{M}) \leq \frac{3n}{2^{|\tau|}} + o(n)$. Hence

$$\Pr[\mathfrak{M} \text{ is balanced}] \cdot \left(\frac{3n}{2^{|\tau|}} - o(n)\right) \leq \frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ balanced}} C(\mathfrak{M}) \leq \Pr[\mathfrak{M} \text{ is balanced}] \cdot \left(\frac{3n}{2^{|\tau|}} + o(n)\right).$$

Since $\Pr[\mathfrak{M} \text{ is balanced}]$ goes to one as $n \to \infty$, we see that $\frac{1}{2^{|\tau|n}} \sum_{\mathfrak{M} \text{ balanced}} C(\mathfrak{M}) \sim \frac{3n}{2^{|\tau|}}$, which is what we wanted to show. ◀

6 Entropy and description complexity

In this section we establish results that illustrate how entropy and description complexity relate to each other. As one can already imagine after seeing our results on description complexity, there can be models with very close entropies and quite different description complexities. We can nevertheless use our results to *exclude* many a priori possible combinations of description complexity and entropy. For notational simplicity, we adopt the notation $t := 2^{|\tau|}$.

We begin by showing that the Boltzmann and Shannon entropies of a single model are essentially the same up to normalization. This underlines the fact that both entropies measure the same thing: the randomness of a model.

▶ **Theorem 13.** *Let $\mathfrak{M}$ be a τ-model of size n. Now*

$$H_S(\mathfrak{M}) - \frac{1}{n} H_B(\mathfrak{M}) < \frac{(t-1)\log(\sqrt{2\pi n})}{n} - \frac{\log(e)}{12n^2} + \frac{t\log(e)}{12n^2+n}.$$

Proof. Using the quantitative version of Stirling's approximation given in [21], we obtain

$$\begin{aligned}
H_B(\mathfrak{M}) &= \log \binom{n}{n_1 \dots n_t} = \log \frac{n!}{n_1! \dots n_t!} = \log(n!) - \sum_{i=1}^{t} \log(n_i!) \\
&< \log\left(\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}\right) - \sum_{i=1}^{t} \log\left(\sqrt{2\pi n_i}\left(\frac{n_i}{e}\right)^{n_i} e^{\frac{1}{12n+1}}\right) \\
&= \log(\sqrt{2\pi n}) + n\log(n) - n\log(e) + \frac{\log(e)}{12n} \\
&\quad - \sum_{i=1}^{t}\left(\log(\sqrt{2\pi n_i}) + n_i\log(n_i) - n_i\log(e) + \frac{\log(e)}{12n+1}\right) \\
&\leq n\log(n) - \sum_{i=1}^{t} n_i\log(n_i) - (t-1)\log(\sqrt{2\pi n}) + \frac{\log(e)}{12n} - \frac{t\log(e)}{12n+1}.
\end{aligned}$$

Note that the term $n\log(e)$ is cancelled out above because $n_1 + \cdots + n_t = n$. Using this same fact we also easily see that

$$H_S(\mathfrak{M}) = \sum_{i=1}^{t} -\frac{n_i}{n}\log\frac{n_i}{n} = \sum_{i=1}^{t}\frac{n_i}{n}\log(n) - \sum_{i=1}^{t}\frac{n_i}{n}\log(n_i) = \log(n) - \sum_{i=1}^{t}\frac{n_i}{n}\log(n_i).$$

Finally, by dividing $H_B(\mathfrak{M})$ with n we obtain

$$H_S(\mathfrak{M}) - \frac{1}{n} H_B(\mathfrak{M}) < \frac{(t-1)\log(\sqrt{2\pi n})}{n} - \frac{\log(e)}{12n^2} + \frac{t\log(e)}{12n^2+n}.$$ ◀

The above quantitative result readily implies that the Boltzmann and Shannon entropies of a single model are asymptotically the same up to normalization. A connection that bears a similarity to the one pointed out here has also been noted briefly in [15].

▶ **Corollary 14.** *Let $(\mathfrak{M}_n)_{n\in\mathbb{Z}_+}$ be a sequence of τ-models where each $\mathfrak{M}_n$ has size n. Now $H_S(\mathfrak{M}_n) \sim \frac{1}{n} H_B(\mathfrak{M}_n)$ as $n \to \infty$.*

The above results show that for the connections to description complexity, we could use either of the two notions of entropy. We opt for Shannon entropy here.

We will next use results from Sections 3 and 4 to prove two theorems that give bounds on description complexity in terms of Shannon entropy. Recall from Section 3 the constant $c_\tau := 15|\tau|2^{|\tau|}$. The first of our two theorems gives global upper and lower bounds on description complexity based on the same edge case distributions.

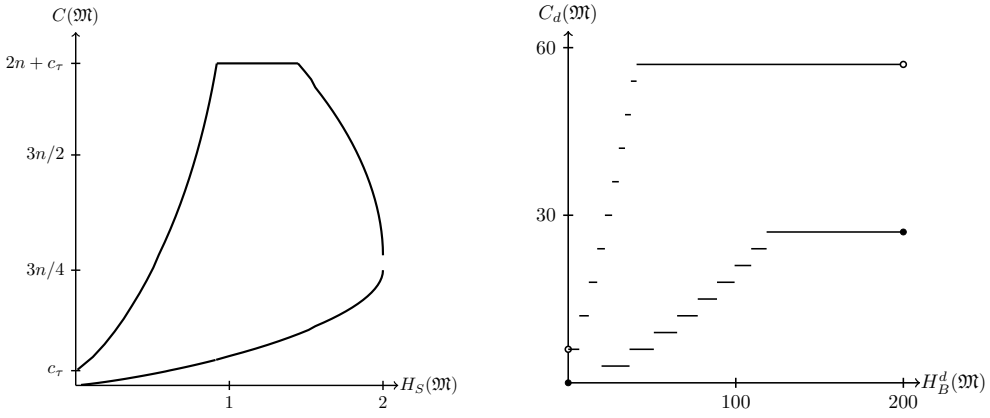
▶ **Theorem 15.** *Let* $p \in [0, \frac{1}{t}[$. *If* $H_S(\mathfrak{M}) > ((t-1)p-1)\log(1-(t-1)p) - (t-1)p\log(p)$, *then* $3np - 3 < C(\mathfrak{M}) < 3n(1-(t-1)p) + c_\tau$.

Proof. Let $f(p) := ((t-1)p-1)\log(1-(t-1)p) - (t-1)p\log(p)$. The function $f(p)$ gives the entropy of a τ-model $\mathfrak{M}'$ corresponding to the tuple $(np, \dots, np, n(1-(t-1)p))$, where $n(1-(t-1)p) > np$ for the given values of p. Since all types but the largest are evenly distributed, any model, where the largest type has at least $n(1-(t-1)p)$ realizing points has entropy at most $H_S(\mathfrak{M}') = f(p)$. Therefore if $H_S(\mathfrak{M}) > f(p)$, then the largest type of $\mathfrak{M}$ has less than $n(1-(t-1)p)$ realizing points. By Theorem 1, we obtain $C(\mathfrak{M}) < 3n(1-(t-1)p)+c_\tau$. On the other hand, since the largest type of $\mathfrak{M}$ has less realizing points than in $\mathfrak{M}'$, those points realize some other type. Therefore the second largest type of $\mathfrak{M}$ has more than np realizing points. By Theorem 8, we obtain $C(\mathfrak{M}) > 3np - 3$. ◀

The next theorem uses low entropy models with only two realized types to show a better upper bound on description complexity for low entropy models than the above global one.

▶ **Theorem 16.** *Let* $p \in [0, \frac{1}{2}]$. *If* $H_S(\mathfrak{M}) < (p-1)\log(1-p) - p\log(p)$, *then* $C(\mathfrak{M}) < 6np + c_\tau$.

Proof. Let $h(p) := (p-1)\log(1-p) - p\log(p)$. The function $h(p)$ gives the entropy of a τ-model $\mathfrak{M}$ corresponding to the tuple $(0, \dots, 0, np, n(1-p))$. If $H_S(\mathfrak{M}) < h(p)$, then the second largest type of $\mathfrak{M}$ must be smaller than np. Thus, by Theorem 1, $C(\mathfrak{M}) < 6np + c_\tau$. ◀



(a) **(b)**

Figure 1 Figure 1a on the left shows an area that encapsulates all combinations of Shannon entropy and FO-description complexity for the values $|\tau| = 2$ and $n = 1000$. Figure 1b on the right concerns the case of FO_d and shows bounds on description complexity in terms of Boltzmann entropy for values $|\tau| = 2$, $n = 100$ and $d = 10$ with the constants -3 and c_τ omitted.

Figure 1a incorporates both of the above theorems as well as Corollary 2 to show an area, where all possible combinations of Shannon entropy and description complexity must fall. First, comparing the left side of the plot to the right, we can see that models with very high entropy have significantly higher description complexity than models with very low entropy.

We can also see from Figure 1a that the gap between our upper bounds and lower bounds is only constant at both extremes of entropy. For models with middling entropy, the gap is at its largest. This is because middling values of entropy can be realized by models with very different distributions of types, leading to different description complexity.

We conjecture that the upper bound given by Theorem 1 is in reality tight up to the constant c_τ. Now, recall that for any single model, our upper and lower bounds have a worst case gap of a factor of 2. Therefore, assuming that our conjecture is true, the lower bound would only rise to at most double its current height. In other words, the general picture illustrated by Figure 1a would not be significantly different under our conjecture.

We proceed to show that similar relationships between description complexity and entropy hold also in the case of limited quantifier rank. As the classes of $\equiv_d$ contain multiple different isomorphism types of models, it is not clear how to define Shannon entropy. Boltzmann entropy, however, is still straightforward so we use Boltzmann entropy here. We formulate similar theorems to those above for full FO.

▶ **Theorem 17.** *Let $h \in \{1, ..., d-1\}$. If $H_B^d(\mathfrak{M}) > \log \binom{n}{h \dots h \; n-(t-1)h}$, then $C_d(\mathfrak{M}) > 3h-3$.*

Proof. Let $f(n,h) = \log \binom{n}{h \dots h \; n-(t-1)h}$. The function $f(n,h)$ gives the Boltzmann entropy of the class of models $\mathcal{M}_{\overline{m}}$, where $\overline{m} = (h, \dots, h, d)$. Any class of models obtained from this one by lowering any of the numbers in the tuple is clearly smaller than $\mathcal{M}_{\overline{m}}$ and thus has lower Boltzmann entropy. Thus, for any larger class of models the second largest number in its tuple must be greater than h. By Corollary 9, we obtain $C_d(\mathfrak{M}) > 3h - 3$. ◀

▶ **Theorem 18.** *Let $h \in \{1, \dots, d-1\}$. If $H_B^d(\mathfrak{M}) < \log \binom{n}{h}$, then $C_d(\mathfrak{M}) < 6h + c_\tau$.*

Proof. The function $g(n,h) = \log \binom{n}{h}$ gives the Boltzmann entropy of a class $\mathcal{M}_{\overline{m}}$ of models, where $\overline{m} = (0, \dots, 0, h, d)$. Now every class of models, where the second largest number in the tuple is at least h, is larger than or equal to $\mathcal{M}_{\overline{m}}$. Thus if $H_B^d(\mathfrak{M}) < g(n,h)$, then the class of $\mathfrak{M}$ is smaller and the second largest number in its tuple is smaller than h. By Theorem 3 we obtain $C_d(\mathfrak{M}) < 6h + c_\tau$. ◀

We again have a plot in Figure 1b, where the possible combinations of entropy and description complexity lie between the two chopped lines. This time, we plotted from the above theorems $3h$ for the lower bound and $6h$ for the upper bound, omitting the constants -3 and c_τ. For these low values of n and d, the constants would have warped the picture in a significant way. With high enough n and d, the constants are clearly negligible, but for such values, the Boltzmann entropy quickly becomes impractical to calculate as the model class sizes explode. We provide a plot of the leading terms for the values $n = 100$ and $d = 10$ without the constants to illustrate the trends one would see for higher values of n and d.

We see that the first observation we made for full FO still holds. The models with very high entropy have significantly higher description complexity than those with very low entropy. Concerning the gap between the upper and lower bounds, it is again constant at the extremes. The largest gap can now be found significantly before the halfway point of entropy, unlike for full FO. This is because the limit d of quantifier rank quite quickly cuts short the growth of the upper bound while the lower bound grows slower.

7 Conclusion

We have studied the description complexity of unary models, obtaining bounds for FO and FO_d. We have found the asymptotic description complexity of a random unary structure and studied the relation between Shannon entropy and description complexity – also observing a connection between Boltzmann and Shannon entropy. Links to entropy can be useful as computing entropy is *significantly easier than determining description complexity.*

An obvious future goal would be to close the gaps between the upper and lower bounds. Generalizing to full relational vocabularies is also interesting, although this seems to require highly involved arguments. The part on entropy would there relate to Boltzmann entropy, as there is no obvious unique definition for Shannon entropy in the k-ary scenario.

References

1 Micah Adler and Neil Immerman. An *n!* lower bound on formula size. *ACM Trans. Comput. Log.*, 4(3):296–314, 2003. doi:10.1145/772062.772064.

2 Philippe Balbiani, David Fernández-Duque, Andreas Herzig, and Petar Iliev. Frame-validity games and lower bounds on the complexity of modal axioms. *Log. J. IGPL*, 30(1):155–185, 2022. doi:10.1093/jigpal/jzaa068.

3 Pablo Barceló, Mikaël Monet, Jorge Pérez, and Bernardo Subercaseaux. Model interpretability through the lens of computational complexity. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/b1adda14824f50ef24ff1c05bb66faf3-Abstract.html.

4 Marco Carmosino, Ronald Fagin, Neil Immerman, Phokion Kolaitis, Jonathan Lenchner, and Rik Sengupta. On the number of quantifiers needed to define boolean functions, 2024.

5 Marco Carmosino, Ronald Fagin, Neil Immerman, Phokion G. Kolaitis, Jonathan Lenchner, and Rik Sengupta. A finer analysis of multi-structural games and beyond. *CoRR*, abs/2301.13329, 2023. doi:10.48550/arXiv.2301.13329.

6 Marco Leandro Carmosino, Ronald Fagin, Neil Immerman, Ph. G. Kolaitis, Jonathan Lenchner, Rik Sengupta, and Ryan Williams. Parallel play saves quantifiers. *ArXiv*, 2024.

7 Ronald Fagin, Jonathan Lenchner, Kenneth W. Regan, and Nikhil Vyas. Multi-structural games and number of quantifiers. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, 2021. doi:10.1109/LICS52264.2021.9470756.

8 Ronald Fagin, Jonathan Lenchner, Nikhil Vyas, and Ryan Williams. On the Number of Quantifiers as a Complexity Measure. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 48:1–48:14, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2022.48.

9 Peter Grünwald and Paul M. B. Vitányi. Shannon information and Kolmogorov complexity. *CoRR*, cs.IT/0410002, 2004. URL: http://arxiv.org/abs/cs.IT/0410002, doi:10.48550/arXiv.cs/0410002.

10 Lauri Hella and Jouko Väänänen. The size of a formula as a measure of complexity. In Åsa Hirvonen, Juha Kontinen, Roman Kossak, and Andrés Villaveces, editors, *Logic Without Borders - Essays on Set Theory, Model Theory, Philosophical Logic and Philosophy of Mathematics*, volume 5 of *Ontos Mathematical Logic*, pages 193–214. De Gruyter, 2015. doi:10.1515/9781614516873.193.

11 Lauri Hella and Miikka Vilander. Formula size games for modal logic and μ-calculus. *J. Log. Comput.*, 29(8):1311–1344, 2019. doi:10.1093/logcom/exz025.

12 Reijo Jaakkola, Tomi Janhunen, Antti Kuusisto, Masood Feyzbakhsh Rankooh, and Miikka Vilander. Explainability via short formulas: the case of propositional logic with implementation. In *Joint Proceedings of (HYDRA 2022) and the RCRA Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion*, volume 3281 of *CEUR Workshop Proceedings*, pages 64–77, 2022. URL: https://ceur-ws.org/Vol-3281/paper6.pdf.

13 Reijo Jaakkola, Tomi Janhunen, Antti Kuusisto, Masood Feyzbakhsh Rankooh, and Miikka Vilander. Short boolean formulas as explanations in practice. In Sarah Alice Gaggl, Maria Vanina Martinez, and Magdalena Ortiz, editors, *Logics in Artificial Intelligence - 18th European Conference, JELIA 2023, Dresden, Germany, September 20-22, 2023, Proceedings*, volume 14281 of *Lecture Notes in Computer Science*, pages 90–105. Springer, 2023. doi:10.1007/978-3-031-43619-2_7.

14 Reijo Jaakkola, Antti Kuusisto, and Miikka Vilander. Relating description complexity to entropy. In Petra Berenbrink, Patricia Bouyer, Anuj Dawar, and Mamadou Moustapha Kanté, editors, *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*, volume 254 of *LIPIcs*, pages 38:1–38:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.STACS.2023.38.

15 Andrey Kolmogorov. The theory of transmission of information. In *Selected Works of A. N. Kolmogorov: Volume III: Information Theory and the Theory of Algorithms*, pages 6–32. Springer Netherlands, 1993. doi:10.1007/978-94-017-2973-4_3.

16 Sik K. Leung-Yan-Cheong and Thomas M. Cover. Some equivalences between Shannon entropy and Kolmogorov complexity. *IEEE Trans. Inf. Theory*, 24(3):331–338, 1978. doi:10.1109/TIT.1978.1055891.

17 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.

18 João Marques-Silva, Thomas Gerspacher, Martin C. Cooper, Alexey Ignatiev, and Nina Narodytska. Explanations for monotonic classifiers. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 7469–7479. PMLR, 2021. URL: http://proceedings.mlr.press/v139/marques-silva21a.html.

19 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. doi:10.1017/CBO9780511813603.

20 Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Comb.*, 10(1):81–93, 1990. doi:10.1007/BF02122698.

21 Herbert Robbins. A remark on stirling's formula. *The American Mathematical Monthly*, 62(1):26–29, 1955. doi:10.2307/2308012.

22 Advait Sarkar. Is explainable AI a race against model complexity? In *Workshop on Transparency and Explanations in Smart Systems (TeXSS), in conjunction with ACM Intelligent User Interfaces (IUI 2022)*, volume 3124 of *CEUR Workshop Proceedings*, pages 192–199, 2022.

23 Andreia Teixeira, Armando Matos, Andre Souto, and Luis Filipe Coelho Antunes. Entropy measures vs. Kolmogorov complexity. *Entropy*, 13(3):595–611, 2011. doi:10.3390/e13030595.

A Appendix

A.1 Proof of Theorem 1 continued

We define here the second upper bound formula $\psi(\mathfrak{M})$ of size at most $6|\pi_{\ell-1}| + c_\tau$, along with required subformulas.

Let T, $\overline{m}$ and $\mathfrak{M}$ be as in the proof so far. We define another formula $\chi(T, \overline{m})$ below. Now the model $\mathfrak{M}$ satisfies $\chi(T, \overline{m})$ if and only if for every $i \in \{1, \ldots, r\}$, the model $\mathfrak{M}$ has *at most* m_i points that realize the type π_i. We again do not assert anything about the types π_j with no corresponding m_j.

$$\theta_{m_r} := y = x_{m_r} \vee \bigvee_{\substack{j\in\{1,\dots,r\}\\ m_j=m_r}} (\pi_j(x_1) \wedge \neg\pi_j(y))$$

$$\theta_i := y = x_i \vee \theta_{i+1}, \text{ if } m_j \neq i \text{ for all } j \in \{1,\dots,r\}, \text{ and}$$

$$\theta_i := y = x_i \vee \Big(\bigvee_{\substack{j\in\{1,\dots,r\}\\ m_j=i}} (\pi_j(x_1) \wedge \neg\pi_j(y))\Big) \vee \Big(\bigwedge_{\substack{j\in\{1,\dots,r\}\\ m_j=i}} \neg\pi_j(x_1) \wedge \theta_{i+1}\Big), \text{ otherwise.}$$

$$\chi(T,\overline{m}) := \forall x_1 \exists x_2 \dots \exists x_{m_r} \forall y \Big(\bigvee_{j\in\{r+1,\dots,\ell\}} \pi_j(x_1) \vee \theta_1\Big)$$

We again explain how the above formula works. Note that directly taking the negation of the formula $\varphi(T,\overline{m})$ would not work as we are dealing with all types at once. We instead again start with a universally quantified variable x_1 that is attached to a point realizing a type $\pi_j \in T$. We first check if π_j is one of the types we can safely ignore. Assume then that $m_j = 5$. The existentially quantified variables $x_2, \dots, x_5$ are then chosen to be of the same type π_j as x_1 in such a way that every point of the type π_j has at least one x_i attached to it. Since $m_j = 5$, the first step of the recursion insists that either y is the same as x_1 or the recursion continues. When the recursion arrives at θ_5, we cannot go any further, as to continue, we would need $m_j \neq 5$. We are instead left with the two options of either $y = x_5$ or y realizes a different type than x_1. This amounts to saying that there are no more than 5 points that realize the type π_j.

The crucial point of the formula $\chi(T,\overline{m})$ is that the first universally quantified variable x_1 allows us to use the same existential quantifiers to count all types at once. To ensure that we do not require all of the types to be the same size, we restrict the type realized by x_1 before continuing with the recursion.

We compute the size of $\chi(T,\overline{m})$. The formula has $m_r + 1$ quantifiers. For each type π, the subformula $\pi(x)$ occurs at most three times and for at least one type with $|\pi| = m_r$, only two times. This results in $3k|T| - k$ atomic formulas of the form $P(x)$ or $\neg P(x)$. For the equalities and inequalities, each equality $y = x_i$ for $1 \leq i \leq m_r$ occurs exactly once, for a total of m_r such atomic formulas. Accounting for the binary connectives, the size of $\chi(T,\overline{m})$ is thus at most

$$m_r + 1 + 2(m_r + 3k|T| - k) - 1 = \ 3m_r + 6k|T| - 2k.$$

Our second complete upper bound formula $\psi(\mathfrak{M})$ avoids counting the type π_ℓ with the most realizing points by bounding the size of all other types from above and from below. For this formula we denote by $\overline{m} \setminus |\pi_\ell|$ the sequence $(|\pi_1|, \dots, |\pi_{\ell-1}|)$. We define

$$\psi(\mathfrak{M}) := \bigwedge_{i=1}^{\ell} \exists x\, \pi_i(x) \wedge \forall x \bigvee_{i=1}^{\ell} \pi_i(x) \wedge \varphi(T,\overline{m} \setminus |\pi_\ell|) \wedge \chi(T,\overline{m} \setminus |\pi_\ell|).$$

The numbers of new quantifiers and atomic formulas are the same as for $\varphi(\mathfrak{M})$. Accounting for the binary connectives, including the one connecting $\varphi(T,\overline{m} \setminus |\pi_\ell|)$ and $\chi(T,\overline{m} \setminus |\pi_\ell|)$, the size of $\psi(\mathfrak{M})$ is now at most

$$\begin{aligned} &|T| + 1 + 2(k|T| + k|T|) + 3|\pi_{\ell-1}| + 4k|T| - 3 + 3|\pi_{\ell-1}| + 6k|T| - 2k + 1 \\ &= 6|\pi_{\ell-1}| + 14k|T| + |T| - 2k - 1 \leq 6|\pi_{\ell-1}| + c_\tau. \end{aligned}$$

A.2 Proof of Theorem 3

Let $\overline{m} = (m_1, \ldots, m_t)$ be a tuple corresponding to a class of $\equiv_d$, ordered in the following way. The first numbers $m_1, \ldots, m_r$ are the ones greater than 0 and smaller than d in ascending order. The numbers $m_{r+1}, \ldots, m_\ell$ are all equal to d, and finally the numbers $m_{\ell+1}, \ldots, m_t$ are all equal to 0.

Using this order for the types, the set $T = \{\pi_1, \ldots, \pi_\ell\}$ is now the set of types realized in models of the class and the first r types are each realized exactly $m_i < d$ times. This is in line with the notation of the formulas for full FO above.

Our first formula works for any $\overline{m}$. The formula states that each type π_j is realized at least m_j times and furthermore, the ones with $m_j < d$ are realized at most m_j times.

$$\varphi_d(\overline{m}) := \bigwedge_{i=1}^{\ell} \exists x\, \pi_i(x) \wedge \forall x \bigvee_{i=1}^{\ell} \pi_i(x) \wedge \varphi(T, (m_1, \ldots, m_\ell)) \wedge \chi(T, (m_1, \ldots, m_r))$$

In the same way as for $\psi(\mathfrak{M})$ in the proof of Theorem 1, the size of $\varphi_d(\overline{m})$ is at most

$$\begin{aligned} &|T| + 1 + 2(k|T| + k|T|) + 3d + 4k|T| - 3 + 3m_r + 6k|T| - 2k + 1 \\ &= 3d + 3m_r + 14k|T| + |T| - 2k - 1 \leq 3d + 3m_r + c_\tau. \end{aligned}$$

Our second formula is only for the special case, where there is exactly one m_j equal to d. In this case, as with full FO, we can avoid counting the type with the most realizing points. The rest of the types π_j have $m_j < d$ and the formula states that each π_j is realized at least and at most m_j times.

$$\psi_d(\overline{m}) := \bigwedge_{i=1}^{\ell} \exists x\, \pi_i(x) \wedge \forall x \bigvee_{i=1}^{\ell} \pi_i(x) \wedge \varphi(T, (m_1, \ldots, m_r)) \wedge \chi(T, (m_1, \ldots, m_r))$$

Again in the same way as for $\psi(\mathfrak{M})$ in the proof of Theorem 1, the size of $\psi_d(\overline{m})$ is at most

$$\begin{aligned} &|T| + 1 + 2(k|T| + k|T|) + 3m_r + 4k|T| - 3 + 3m_r + 6k|T| - 2k + 1 \\ &= 6m_r + 14k|T| + |T| - 2k - 1 \leq 6m_r + c_\tau. \end{aligned}$$

The upper bounds of the claim follow.

A.3 Proof of Lemma 11

We will use Proposition 10. For every type π and $1 \leq i \leq n$ we associate a 0-1-valued random variable $X_{\pi,i}$ such that $X_{\pi,i} = 1$ with probability $2^{-|\tau|}$ and $X_{\pi,i} = 0$ with probability $1 - 2^{-|\tau|}$. Intuitively this is an indicator random variable for the event "the ith element received the type π". Now $X_\pi = \sum_{i=1}^{n} X_{\pi,i}$ is a random variable that counts the number of times π is realized. Clearly $\mathbb{E}[X_\pi] = n/2^{|\tau|}$, which also holds for every type π. Set $\mu := n/2^{|\tau|}$ and $\delta(n) := \sqrt{\frac{3}{2^{|\tau|}} \frac{\ln(n)}{n}}$. Now

$$2e^{-\delta(n)^2 \mu/3} = 2n^{-1}$$

and

$$\delta(n)\mu = \frac{\sqrt{3}}{2^{|\tau|}\sqrt{2^{|\tau|}}} \sqrt{\ln(n)n}.$$

Thus, by Proposition 10, we know that

$$\Pr\left[|X_\pi - \mu| \geq \frac{\sqrt{3}}{2^{|\tau|}\sqrt{2^{|\tau|}}}\sqrt{\ln(n)n}\right] \leq 2n^{-1}$$

Applying the union bound, we also see that

$$\begin{aligned}
&\Pr\left(\exists \pi \ : \ |X_\pi - \mu| \geq \frac{\sqrt{3}}{2^{|\tau|}\sqrt{2^{|\tau|}}}\sqrt{\ln(n)n}\right) \\
&\leq \sum_\pi \Pr\left[|X_\pi - \mu| \geq \frac{\sqrt{3}}{2^{|\tau|}\sqrt{2^{|\tau|}}}\sqrt{\ln(n)n}\right] \\
&\leq 2^{|\tau|+1} n^{-1}
\end{aligned}$$

Thus, with probability at least $1 - 2^{|\tau|+1}/n$ in a random model $\mathfrak{M}$ of size n we have for every type π that

$$\left||\pi|_{\mathfrak{M}} - \frac{n}{2^{|\tau|}}\right| \leq \frac{\sqrt{3}}{2^{|\tau|}\sqrt{2^{|\tau|}}}\sqrt{\ln(n)n}.$$

Hence, with probability at least $1 - 2^{|\tau|+1}/n$ a random model of size n is balanced.

Reachability for Multi-Priced Timed Automata with Positive and Negative Rates

Andrew Scoones
Department of Computer Science, University of Oxford, UK

Mahsa Shirmohammadi
CNRS, IRIF, Université of Paris Cité, France

James Worrell
Department of Computer Science, University of Oxford, UK

Abstract

Multi-priced timed automata (MPTA) are timed automata with observer variables whose derivatives can change from one location to another. Observers are read-once variables: they do not affect the control flow of the automaton and their value is output only at the end of a run. Thus MPTA lie between timed and hybrid automata in expressiveness. Previous work considered observers with non-negative slope in every location. In this paper we treat observers that have both positive and negative rates. Our main result is an algorithm to decide a gap version of the reachability problem for this variant of MPTA. We translate the gap reachability problem into a gap satisfiability problem for mixed integer-real systems of nonlinear constraints. Our main technical contribution – a result of independent interest – is a procedure to solve such contraints via a combination of branch-and-bound and relaxation-and-rounding.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Quantitative automata; Theory of computation → Timed and hybrid models; Theory of computation → Verification by model checking

Keywords and phrases Bilinear constraints, Existential theory of real closed fields, Diophantine approximation, Pareto curve

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.18

Funding *Andrew Scoones*: Supported by UKRI Frontier Research Grant EP/X033813/1.
Mahsa Shirmohammadi: Supported by VeSyAM (ANR-22- CE48-0005).
James Worrell: Supported by UKRI Frontier Research Grant EP/X033813/1.

1 Introduction

Timed automata [1] are a widely studied model of real-time systems that extend classical finite state-automata with real-valued variables, called *clocks*, that evolve with derivative one and which can be queried and reset along transitions. *Multi-Priced Timed Automata* (MPTA) [7, 10, 13, 25] further extend timed automata with variables, called *observers*, that have a non-negative slope that can change from one location to another. Such variables can model the accumulation of costs or the use of resources along a computation, such as energy and memory consumption in embedded systems, or bandwidth in communication networks. For this reason MPTA are widely used to model multi-objective real-time optimisation problems [9].

While observers exhibit richer dynamics than clocks, they may not be queried while taking edges. Thus MPTA lie between timed automata (for which reachability is decidable) and linear hybrid automata (for which reachability is undecidable [17]). A natural class of verification problems for MPTA concerns reachability subject to constraints on the observers. A simple variant is the *Domination Problem*, which asks to reach a location subject to upper

bounds on each observer. Here one can think of the constraints as representing upper bounds on accumulated costs or resources. The Domination Problem was shown decidable in [21] using well-quasi-orders and was later shown to be PSPACE-complete in [12, Theorem 4].

A more expressive version of the Domination Problem partitions the set of observers into *cost variables* and *reward variables* and asks to reach a location subject to upper bounds on costs and lower bounds on rewards. This variant is, unfortunately, undecidable. However it is shown in [12, Theorem 6] that a gap version of the problem – called the *Gap Domination Problem* – is decidable. In the Gap Domination Problem the input additionally contains a slack $\varepsilon > 0$. The objective is to distinguish the case that the constraints on the observers can be satisfied with slack ε from the case in which they cannot be satisfied at all. In general, gap problems are decision versions of approximation problems [3, Chapter 18.2]. Decidability of the Gap Domination Problem implies that the Pareto curve of undominated reachable cost vectors can be computed to arbitrary precision (cf. [11]).

The objective of this paper is to address a more expressive variant of MPTA than hitherto considered: namely those in which observers can have both positive and negative rates. Alternatively, and equivalently, one can consider MPTA with nonnegative rates, but in which one allows reachability specifications to contain constraints on the *difference* between two observers rather than just threshold constraints that compare observers to constants. Indeed, this extension is motivated by the desire to measure net resource use along computations. In this more general setting, the Domination Problem, of course, remains undecidable; one moreover loses monotonicity properties on which previous positive decidability results rely, including the decision procedure for the Gap Domination Problem given in [12, Theorem 15]. The main result of this paper is to establish decidability (in nondeterministic exponential time) of the Gap Domination Problem in the presence of positive and negative rates via a new decision procedure.

We start by recalling a result of [12] that characterises the set of all reachable observer values for a given MPTA via a system of mixed integer-real nonlinear constraints. Our main technical contribution, which is of independent interest, shows how to solve a gap version of the satisfiability problem for such systems of constraints. Our method involves a combination of relaxation-and-rounding and branch-and-bound that relies on Khinchine's Flatness Theorem from Diophantine approximation. We formulate a relaxation of the system of constraints such that a solution to the relaxed version can be rounded to a solution of the original problem, while unsolvability of the relaxed version permits a branch-and-bound step that eliminates a variable from the original system of constraints.

Systems of non-linear constraints over integer and real variables appear in many different domains and are widely studied, although typically not from the point of view of decidability since most classes of problems with unbounded integer variables are undecidable [16]. Other than [12], we are not aware of previous work on the gap problem considered here. Kachiyan and Porkolab [19] showed that it is decidable whether a convex semialgebraic set contains an integer point; however we work with non-convex sets.

In this paper we consider MPTA with arbitrarily many observers. There is a significant literature and mature tool support concerning the special case of MPTA with a single observer, which are variously called Priced Timed Automata or Weighted Timed Automata. In this case, the optimal cost to reach a given location is computable [2, 6, 20]. In the case of one cost and one reward observer, one can also compute the optimal reward-to-cost ratio in reaching a given location [7]. The preceding results use the so-called *corner-point abstraction*, which is insufficient for multi-objective model checking. Instead, the present paper implicitly relies on the *simplex-automaton abstraction*, introduced in [12], which underlies the non-linear

constraint problems that are the subject of our main results. All previously mentioned works involve observers that evolve linearly with time. Observer variables that vary non-linearly with time are considered in [4]. In the non-linear setting the optimal cost reachability problem is undecidable in general. Another variant, this time towards greater simplicity, is to consider observers that are only updated through discrete transitions [26].

2 Automata and Decision Problems

2.1 Multi-Priced Timed Automata

Let $\mathbb{R}_{\geq 0}$ denote the set of non-negative real numbers. Given a set $\mathcal{X} = \{x_1, \ldots, x_n\}$ of *clocks*, the set $\Phi(\mathcal{X})$ of *clock constraints* is generated by the grammar

$$\varphi ::= \texttt{true} \mid x \leq k \mid x \geq k \mid \varphi \wedge \varphi \,,$$

where $k \in \mathbb{N}$ is a natural number and $x \in \mathcal{X}$. A *clock valuation* is a mapping $\nu : \mathcal{X} \to \mathbb{R}_{\geq 0}$ that assigns to each clock a non-negative real number. We denote by $\mathbf{0}$ the valuation such that $\mathbf{0}(x) = 0$ for all clocks $x \in \mathcal{X}$. We write $\nu \models \varphi$ to denote that ν satisfies the constraint φ. Given $t \in \mathbb{R}_{\geq 0}$, we let $\nu + t$ be the clock valuation such that $(\nu + t)(x) = \nu(x) + t$ for all clocks $x \in \mathcal{X}$. Given $\lambda \subseteq \mathcal{X}$, let $\nu[\lambda \leftarrow 0]$ be the clock valuation such that $\nu[\lambda \leftarrow 0](x) = 0$ if $x \in \lambda$, and $\nu[\lambda \leftarrow 0](x) = \nu(x)$ otherwise.

A *multi-priced timed automaton* (MPTA) $\mathcal{A} = \langle L, \ell_0, L_f, \mathcal{X}, \mathcal{Y}, E, R \rangle$ comprises a finite set L of *locations*, an *initial location* $\ell_0 \in L$, a set $L_f \subseteq L$ of *accepting locations*, a finite set $\mathcal{X}$ of *clock variables*, a finite set $\mathcal{Y}$ of *observers*, a set $E \subseteq L \times \Phi(\mathcal{X}) \times 2^{\mathcal{X}} \times L$ of *edges*, and a *rate function* $R : L \to \mathbb{Z}^{\mathcal{Y}}$. Here $R(\ell)(y)$ is the derivative of the observer $y \in \mathcal{Y}$ in location ℓ. Denote by $\|\mathcal{A}\|$ the length of the description of $\mathcal{A}$, where all integers are written in binary.

A *state* of $\mathcal{A}$ is a triple (ℓ, ν, t) where ℓ is a location, ν a clock valuation, and $t \in \mathbb{R}_{\geq 0}$ is a *time stamp*. A *run* of $\mathcal{A}$ is an alternating sequence of states and edges

$$\rho = (\ell_0, \nu_0, t_0) \xrightarrow{e_1} (\ell_1, \nu_1, t_1) \xrightarrow{e_2} \ldots \xrightarrow{e_m} (\ell_m, \nu_m, t_m) \,,$$

where $t_0 = 0$, $\nu_0 = \mathbf{0}$, $t_{i-1} \leq t_i$ for all $i \in \{1, \ldots, m\}$, and $e_i = \langle \ell_{i-1}, \varphi, \lambda, \ell_i \rangle \in E$ is such that $\nu_{i-1} + (t_i - t_{i-1}) \models \varphi$ and $\nu_i = (\nu_{i-1} + (t_i - t_{i-1}))[\lambda \leftarrow 0]$ for $i = 1, \ldots, m$. The run is *accepting* if $\ell_m \in L_f$. The *value* of such a run is a vector $\mathrm{val}(\rho) \in \mathbb{R}^{\mathcal{Y}}$, defined by $\mathrm{val}(\rho) = \sum_{i=0}^{m-1} (t_{i+1} - t_i) R(\ell_i)$. We refer to Figure 1 for an example of an MPTA and its operational semantics.

2.2 The Gap Domination Problem

The *Domination Problem* is as follows. Given an MPTA $\mathcal{A}$ with set $\mathcal{Y}$ of observers and a target $\gamma \in \mathbb{R}^{\mathcal{Y}}$, decide whether there is an accepting run ρ of $\mathcal{A}$ such that $\mathrm{val}(\rho) \leq \gamma$ pointwise.

Our formulation of the Domination Problem involves a conjunction of constraints of the form $y \leq c$, where $y \in \mathcal{Y}$ and $c \in \mathbb{Q}$. However such inequalities can encode more general linear constraints of the form $a_1 y_1 + \cdots + a_k y_k \sim c$, where $y_1, \ldots, y_k \in \mathcal{Y}$, $a_1, \ldots, a_k, c \in \mathbb{Z}$ and $\sim \in \{\leq, \geq, =\}$. To do this one introduces a fresh observer to denote each linear term $a_1 y_1 + \cdots + a_k y_k$ (two fresh observers are needed for an equality constraint). For this reduction it is crucial that we allow observers with negative rates.

The Domination Problem is PSPACE-complete for MPTA with positive rates only [12, Theorem 11], but is undecidable if negative rates are allowed [12, Theorem 3]. This motivates us to consider the *Gap Domination Problem* – a variant of the above problem in which

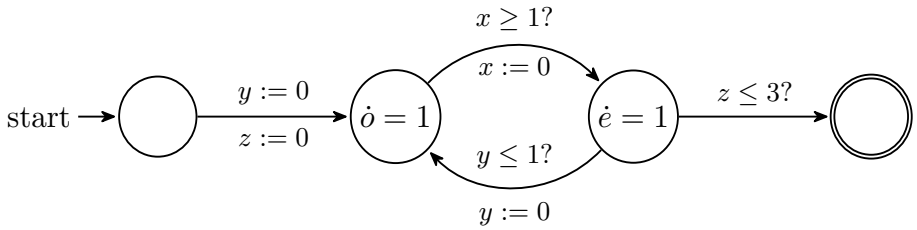


Figure 1 The figure shows an MTPA with three clocks x, y, z and two observer variables o, e, respectively standing for *odd* and *even*. The observer variables have slope 0 unless otherwise indicated; thus o aggregates the total dwell time in the *odd* state and e aggregates the total dwell time in the *even* state. An accepting run is completely determined by a sequence of nonnegative real numbers $d_0, \ldots, d_{2k}$, giving the respective delays between successive transitions. Suppose we wish to reach the accepting state subject to the two objectives $e \geq 2$ and $o \geq 1$. This is achieved, among others, by the run with sequence of time delays $\frac{2}{3}, \frac{1}{3}, \frac{2}{3}, \frac{1}{3}, \frac{2}{3}, \frac{1}{3}, \frac{2}{3}$ and the run with integer sequence of delays $1, 0, 1, 0, 1, 1, 0$ (and any convex combination of the two runs). If the inequalities in the guards on x and y are replaced by equalities then the first run is the unique one realising the two given objectives. In the case of so-called *pure* reachability objectives, i.e., exclusively upper bound constraints or exclusively lower bound constraints on the observers, there is an explicit upper bound on the granularity of the delays in a run witnessing that the objective is realisable ($\frac{1}{3}$ in the present example) [12, Section 6]. This no longer holds in the case of reachability objectives that contain both upper and lower bounds on observers.

the input additionally includes a *slack parameter* $\varepsilon > 0$. If there is some run ρ such that $\mathrm{val}(\rho) \leq \gamma - \varepsilon$ then the output should be "dominated" and if there is no run ρ such that $\mathrm{val}(\rho) \leq \gamma$ then the output should be "not dominated". In case neither of these alternatives hold (i.e., γ is dominated but not with slack ε) then there is no requirement on the output. The Gap Domination Problem is the decision version of the task of computing ε-approximate Pareto curve in the sense of [11].

The following proposition and (a generalisation of [12, Propositions 6 and 7]), concerning the structure of the set of reachable vectors of observer values, allows us to reduce the Gap Domination Problem to a Diophantine problem. Geometrically the proposition says that the set of reachable observer vectors consists of a countable union of simplexes, where each simplex is specified by its vertices – a tuple of integer vectors – and the set of such tuples is semilinear. The proposition is based on the fact that if there are d observers then any reachable observer valuation is a convex combination of $d+1$ valuations that are respectively reached along $d+1$ runs, all taking the same sequence of edges, in which all transitions occur at integer time points (see [12] for details).

▶ **Proposition 1.** *Let $\mathcal{A}$ be an MPTA with set of observers $\mathcal{Y}$ having cardinality d. Then there is a semilinear set $\mathcal{S}_\mathcal{A} \subseteq (\mathbb{Z}^\mathcal{Y})^{d+1}$ such that for every accepting run ρ of $\mathcal{A}$ there exists $(\gamma_1, \ldots, \gamma_{d+1}) \in \mathcal{S}_\mathcal{A}$ for which $\mathrm{cost}(\rho)$ lies in the convex hull of $\{\gamma_1, \ldots, \gamma_{d+1}\}$. Moreover $\mathcal{S}_\mathcal{A}$ can be written as a union of a collection of linear sets that can be computed in time exponential in $\|\mathcal{A}\|$ and each of which has a description length polynomial in $\|\mathcal{A}\|$.*

Proof. The proposition was proved in [12] under the assumption that observers have nonnegative slope. The general case follows easily. Indeed, given an arbitrary MPTA $\mathcal{A} = \langle L, \ell_0, L_f, \mathcal{X}, \mathcal{Y}, E, R \rangle$, we define a new MPTA $\mathcal{A}'$, differing from $\mathcal{A}$ only in its set of observers and rate function, such that all observers in $\mathcal{A}'$ have non-negative rates. The set of observers of $\mathcal{A}'$ is $\mathcal{Y}' := \{y_+, y_- : y \in \mathcal{Y}\}$ and the rate function R' is given by

$$R'(y_+)(\ell) := \max(R(y)(\ell), 0) \quad \text{and} \quad R'(y_-)(\ell) := \max(-R(y)(\ell), 0)$$

for all $y \in \mathcal{Y}$ and all $\ell \in L$.

Define $\Phi : \mathbb{Z}^{\mathcal{Y}'} \to \mathbb{Z}^{\mathcal{Y}}$ by $\Phi(\gamma)(y) = \gamma(y_+) - \gamma(y_-)$. If a run ρ of $\mathcal{A}'$ has cost vector γ then ρ has cost vector $\Phi(\gamma)$ considered as a run of $\mathcal{A}$. Thus if we define $\mathcal{S}_{\mathcal{A}} := \Phi(\mathcal{S}_{\mathcal{A}'})$, where Φ has been lifted pointwise to a linear map $\Phi : (\mathbb{Z}^{\mathcal{Y}'})^{d+1} \to (\mathbb{Z}^{\mathcal{Y}})^{d+1}$, then $\mathcal{S}_{\mathcal{A}}$ satisfies the requirements of the proposition. ◀

The following is immediate from Proposition 1.

▶ **Corollary 2.** *Given $\gamma \in \mathbb{R}^{\mathcal{Y}}$, there exists a run ρ with $\mathrm{val}(\rho) \leq \gamma$ if and only if the following mixed integer-real system of non-linear inequalities has a solution.*

$$\begin{array}{lll} \lambda_1\gamma_1 + \cdots + \lambda_{d+1}\gamma_{d+1} & \leq & \gamma \\ (\gamma_1, \ldots, \gamma_{d+1}) \in \mathcal{S}_{\mathcal{A}} & & \\ \gamma_1, \ldots, \gamma_{d+1} \in \mathbb{Z}^{\mathcal{Y}} & & \end{array} \qquad \begin{array}{lll} 1 & = & \lambda_1 + \cdots + \lambda_{d+1} \\ 0 & \leq & \lambda_1, \ldots, \lambda_{d+1} \\ \lambda_1, \ldots, \lambda_{d+1} & \in & \mathbb{R} \end{array} \tag{1}$$

In the following two sections we analyse systems of constraints of the above form, obtaining a general result that allows us to solve the Gap Domination Problem.

3 Mixed Integer Bilinear Systems

3.1 The Satisfiability Problem

A *mixed-integer bilinear (MIB) system* is a collection of constraints in integer variables $\boldsymbol{x}$ and real variables $\boldsymbol{y}$ of the form:

$$\begin{array}{l} \boldsymbol{x}^\top A_i \boldsymbol{y} \leq b_i \qquad (i = 1, \ldots, \ell) \\ C\boldsymbol{x} \leq \boldsymbol{d} \\ E\boldsymbol{y} \leq \boldsymbol{f} \\ \boldsymbol{x} \in \mathbb{Z}^m, \boldsymbol{y} \in \mathbb{R}^n \,. \end{array} \tag{2}$$

We assume that all constants in (2) are integer; thus if the system is satisfiable then there is a satisfying assignment in which $\boldsymbol{y}$ is a rational vector. We say that a satisfying assignment has *slack* $\varepsilon > 0$ if $\boldsymbol{x}^\top A_i \boldsymbol{y} \leq b_i - \varepsilon$, for $i = 1, \ldots, \ell$. Note that the slack requirement refers only to the nonlinear constraints.

We say that the system (2) is *bounded* if the polyhedron $\{\boldsymbol{y} \in \mathbb{R}^n : E\boldsymbol{y} \leq \boldsymbol{f}\}$ is bounded, i.e., is a polytope. Crucially, the MIB systems arising from multi-priced timed automata in Corollary 2 are bounded. Unfortunately, however, the satisfiability problem for MIB systems is undecidable, even in the bounded case.

▶ **Proposition 3.** *The satisfiability problem for bounded mixed-integer bilinear systems is undecidable.*

Proof. We reduce from the following version of Hilbert's 10th Problem (see [12, Proposition 1]): given a finite system $\mathcal{S}$ of equations in variables $x_1, \ldots, x_n$, with each equation either having the form $x_i = x_j + x_k$ or $x_i = x_j x_k$, determine whether $\mathcal{S}$ has a solution in the set of strictly positive integers.

The reduction involves transforming the system $\mathcal{S}$ into an equisatisfiable MIB system $\mathcal{S}'$ over a set of integer variables $x_0, \ldots, x_n \geq 0$ (i.e, the variables of $\mathcal{S}$ plus a new variable x_0) and real variables $y_1, \ldots, y_n \geq 0$. The construction is such that every solution of $\mathcal{S}$ extends to a solution of $\mathcal{S}'$ and, conversely, every solution of $\mathcal{S}'$ restricts to a solution of $\mathcal{S}$.

The system $\mathcal{S}'$ includes equations $x_0 = 1$ and $x_i y_i = 1$ for $i = 1, \ldots, n$. The linear equations $x_i = x_j + x_k$ from $\mathcal{S}$ are carried over to $\mathcal{S}'$ and, for each equation $x_i = x_j x_k$ in $\mathcal{S}$, we include an equation $(x_j + x_k)y_i = x_0(y_j + y_k)$ in $\mathcal{S}$. The latter is equivalent to

$\frac{x_j+x_k}{x_i} = \frac{1}{x_j} + \frac{1}{x_k}$ in the presence of the equations $x_i y_i = x_j y_j = x_k y_k = 1$ and $x_0 = 1$, which in turn is clearly equivalent to $x_i = x_j x_k$. By adding constraints $0 \leq y_i \leq 1$ for $i = 1, \ldots, n$ we furthermore make $\mathcal{S}'$ bounded without affecting the integrity of the reduction. ◀

3.2 The Gap Satisfiability Problem

In light of Proposition 3, we introduce the following gap version of the satisfiability problem for MIB systems. In this variant we seek a procedure that inputs $\varepsilon > 0$ and a MIB system $\mathcal{S}$ in the form (2) and returns either "UNSAT" or "SAT" subject to the following requirements:

1. If $\mathcal{S}$ has a satisfying assignment with slack ε then the output must be "SAT".
2. If $\mathcal{S}$ is not satisfiable then the output must be "UNSAT".

Note that we place no restriction on the output in the case that $\mathcal{S}$ is satisfiable but with no satisfying assignment having slack ε.

In Section 4 we will show that the Gap Satisfiability Problem is decidable for bounded MIB systems. The following proposition shows the necessity of the boundedness hypothesis.

▶ **Proposition 4.** *The Gap Satisfiability Problem is undecidable for (unbounded) MIB systems.*

Proof. The proof is by reduction from the same variant of Hilbert's Tenth Problem as in the proof of Proposition 3. Recall that an instance of this problem comprises a system $\mathcal{S}$ of equations in positive-integer variables $x_1, \ldots, x_n$, with each equation having the form either $x_i = x_j + x_k$ or $x_i = x_j x_k$, where $i, j, k \in \{1, \ldots, n\}$. Given such a system, we construct an MIB system $\mathcal{S}'$ over integer variables $x_0, \ldots, x_{n+1}$ and real variables $y_0, \ldots, y_{n+1}$ such that every satisfying assignment of $\mathcal{S}$ extends to a satisfying assignment of $\mathcal{S}'$ with slack $\frac{1}{2}$ and every satisfying assignment of $\mathcal{S}'$ restricts to a satisfying assignment of $\mathcal{S}$.

We include the equations $x_0 = 1$ and $y_0 = 1$ in $\mathcal{S}'$. Each linear equation $x_i = x_j + x_k$ in $\mathcal{S}$ is carried over to $\mathcal{S}'$. For each equation $x_i = x_j x_k$ in $\mathcal{S}$ we include the inequality $|x_i y_0 - x_j y_k| \leq \frac{1}{2}$ in $\mathcal{S}'$. We then add the following collection of constraints to $\mathcal{S}'$ for all $i \in \{1, \ldots, n+1\}$ that intuitively force x_i and y_i to be very close together:

1. $|x_i y_0 - x_0 y_i| \leq 1$;
2. $|x_{n+1} y_i - x_i y_{n+1}| \leq 1$;
3. $x_{n+1} y_0 \geq 4(x_0 + x_i)(y_0 + y_i) + 1$.

A satisfying valuation of $\mathcal{S}$ can be extended to a valuation that satisfies $\mathcal{S}'$ with slack $\frac{1}{2}$ by setting $x_0 := 1$, $x_{n+1} := 4 \max_{i \in \{1,\ldots,n\}} (1 + x_i)^2 + 1$, and $y_i := x_i$ for $i = 0, \ldots, n+1$.

Conversely, we claim that every satisfying valuation of $\mathcal{S}'$ (with no assumption on the slack) restricts to a satisfying valuation of $\mathcal{S}$. Indeed, by Item 2, above, for all $k \in \{1, \ldots, n\}$ we have

$$|x_{n+1}(x_k - y_k) - x_k(x_{n+1} - y_{n+1})| = |x_{n+1} y_k - x_k y_{n+1}| \overset{(2)}{\leq} 1.$$

By Items 1 and 3, this entails that for all $j \in \{1, \ldots, n\}$,

$$|x_k - y_k| \leq \frac{x_k |x_{n+1} - y_{n+1}| + 1}{x_{n+1}} \overset{(1)}{\leq} \frac{x_k + 1}{x_{n+1}} \overset{(3)}{\leq} \frac{1}{4(y_j + 1)} \overset{(1)}{\leq} \frac{1}{4x_j}$$

and hence $|x_j x_k - x_j y_k| \leq \frac{1}{4}$. Combined with $|x_i - x_j y_k| \leq \frac{1}{2}$ we conclude that $|x_i - x_j x_k| \leq \frac{3}{4}$ and hence $x_i = x_j x_k$. ◀

It is shown in [12, Theorem 6] how to solve the Gap Satisfiabililty Problem for a subclass of MIB systems, which we here call *positive*. A positive MIB system has the form

$$\begin{aligned}
&\boldsymbol{x}^\top A_i \boldsymbol{y} \leq b_i \qquad (i = 1, \ldots, \ell_1)\\
&\boldsymbol{x}^\top A_i \boldsymbol{y} \geq b_i \qquad (i = \ell_1 + 1, \ldots, \ell_2)\\
&C\boldsymbol{x} \leq \boldsymbol{f}, \boldsymbol{x} \geq \boldsymbol{0}\\
&E\boldsymbol{y} \leq \boldsymbol{f}, \boldsymbol{y} \geq \boldsymbol{0}\\
&\boldsymbol{x} \in \mathbb{Z}^m, \boldsymbol{y} \in \mathbb{R}^n\,.
\end{aligned}$$

with all coefficients of A_i being non-negative rational for $i = 1, \ldots, \ell_2$. This variant can be solved by a naive relaxing and rounding procedure, which does not require the boundedness assumption. However, while suffiicent to handle MPTA with non-negative rates, positive MIB appear insufficient for the case of MPTA with both positive and negative rates.

4 Decidability in the Bounded Case

4.1 Preliminaries

The following proposition on semilinear sets of integers [23, Corollary 1] will be used on several occasions below:

▶ **Proposition 5.** *Consider a set $S := \{\boldsymbol{x} \in \mathbb{Z}^m : A\boldsymbol{x} \leq \boldsymbol{b}\}$, where the entries of A and $\boldsymbol{b}$ are integers of absolute value at most H and the affine hull of S has dimension d. Then there exists a finite set $B \subseteq \mathbb{Z}^m$ and a matrix $P \in \mathbb{Z}^{m\times d}$ such that*

$$S = L(B, P) := \{\boldsymbol{w} + P\boldsymbol{z} : \boldsymbol{w} \in B,\, \boldsymbol{z} \in \mathbb{Z}^d,\, \boldsymbol{z} \geq \boldsymbol{0}\}$$

and the entries of P and $\boldsymbol{w}$ have absolute value at most $(2 + (m + 1)H)^m$.

We will also need the following result [24, Corollary 3.1] on semialgebraic sets of real numbers. We assume that polynomials are written as lists of monomials with all integers, including exponents, written in binary.

▶ **Proposition 6.** *Let $\{f_i\}_{i\in I}$ be a family of polynomials in n variables whose representation has total bit length at most L. Then the set $S := \{\boldsymbol{x} \in \mathbb{R}^n : \bigwedge_{i\in I} f_i \sim_i 0\}$, where $\sim_i \in \{<, =\}$, is either empty or contains a point of distance at most $2^{L^{8n}}$ to the origin.*

For further analysis it will be useful to transform the MIB problem to a *standard form*, shown in (3) below. In standard form the only linear constraints on the integer variables are that they be nonnegative. Correspondingly we enrich the nonlinear constraints, allowing them to contain an extra linear term in $\boldsymbol{y}$.

$$\begin{aligned}
&\boldsymbol{x}^\top A_i \boldsymbol{y} + \boldsymbol{b}_i^\top \boldsymbol{y} \leq c_i \quad (i = 1, \ldots, \ell)\\
&D\boldsymbol{y} \leq \boldsymbol{e}\\
&\boldsymbol{x} \geq \boldsymbol{0}\\
&\boldsymbol{x} \in \mathbb{Z}^m, \boldsymbol{y} \in \mathbb{R}^n\,.
\end{aligned} \tag{3}$$

The transformation of (2) to standard form is based on writing $S := \{\boldsymbol{x} \in \mathbb{Z}^m : C\boldsymbol{x} \leq \boldsymbol{d}\}$ as a semi-linear set $L(B, P)$, following Proposition 5, where $B \subseteq \mathbb{Z}^m$ and $P \in \mathbb{Z}^{m\times d}$ with d the dimension of the affine hull of S. For each vector $\boldsymbol{w} \in B$ we can apply the change of variables $\boldsymbol{x} = P\boldsymbol{z} + \boldsymbol{w}$ to (2) to obtain a problem in standard form: Thus we obtain a finite collection of problems in standard form, whose solutions are in one-one correspondence with the solutions of the original system (2).

4.2 Relaxation and Rounding

In this section we introduce a relaxed version of a bounded MIB system, in which all variables range over the reals. The relaxation is such that a satisfying assignment to the relaxed problem can be rounded to an integer solution of the original system, while unsatisfiability of the relaxed version permits a branch-and-bound step which leads to an equisatisfiable finite collection of MIB instances in one fewer integer variable.

The rounding is based on an application of the Flatness Theorem in Diophantine approximation – Theorem 7, below. To state this result we first recall some standard terminology related to this. Let $K \subseteq \mathbb{R}^n$ be a convex set and let $\boldsymbol{u} \in \mathbb{Z}^n$. Define the *width of K with respect to $\boldsymbol{u}$* to be

$$\mathrm{width}_{\boldsymbol{u}}(K) := \sup\{\boldsymbol{u}^\top(\boldsymbol{x}-\boldsymbol{y}) : \boldsymbol{x},\boldsymbol{y} \in K\}\,.$$

The *lattice width* of K is the minimum width in all directions:

$$\mathrm{width}(K) := \min\{\mathrm{width}_{\boldsymbol{u}}(K) : \boldsymbol{u} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}\}\,.$$

▶ **Theorem 7** (Flatness Theorem)**.** *There exists a constant $\omega(n)$, depending only on n, such that every convex polyhedron $K \subseteq \mathbb{R}^n$ with $\mathrm{width}(K) > \omega(n)$ contains an integer point.*

The constant $\omega(n)$ in Theorem 7 is called the *flatness constant*. The best-known upper bound on $\omega(n) = O(n^{3/2})$ [5], although a linear upper bound was conjectured in [18].

We will need the following proposition about definability of lattice width for classes of polyhedral sets.

▶ **Proposition 8.** *There is a quantifier-free formula in the theory of real closed fields, whose free variables respectively represent a matrix $A \in \mathbb{R}^{n\times m}$, vector $\boldsymbol{b} \in \mathbb{R}^n$, and scalar $c > 0$, that expresses the property $\mathrm{width}_{\boldsymbol{u}}(P) \geq c$ where $P := \{\boldsymbol{x} \in \mathbb{R}^m : A\boldsymbol{x} \geq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$.*

Proof. A necessary condition that $\mathrm{width}_{\boldsymbol{u}}(P) \geq c$ is that P be non-empty and hence, since it lies in the positive orthant, contain a vertex. Now each vertex of P, being the intersection of n linearly independent bounding hyperplanes, has the form $B^{-1}\boldsymbol{b}'$, where B is a non-singular $n \times n$ sub-matrix of $\begin{pmatrix} A \\ I_n \end{pmatrix}$, where I_n denotes the identity matrix of dimension n, and $\boldsymbol{b}'$ is a corresponding sub-vector of $\begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{pmatrix}$. Hence the vertices of P are definable by quantifier-free formulas.

Assume that P contains a vertex. Then $\mathrm{width}_{\boldsymbol{u}}(P)$ is infinite if and only if either $\boldsymbol{u}$ or $-\boldsymbol{u}$ lie in the recession cone of P, for which a sufficient and necessary condition is that $A\boldsymbol{u} \geq \boldsymbol{0}$ or $A\boldsymbol{u} \leq \boldsymbol{0}$. If $\mathrm{width}_{\boldsymbol{u}}(P)$ is finite then there exist two vertices $\boldsymbol{x}_0, \boldsymbol{x}_1$ of P such that $\mathrm{width}_{\boldsymbol{u}}(P) = \boldsymbol{u}^\top(\boldsymbol{x}_0 - \boldsymbol{x}_1)$. The proposition follows by combining the above observations. ◀

We now commence the detailed description of the relaxation construction. The input is a bounded MIB program $\mathcal{S}$ in standard form (3) and a slack $\varepsilon > 0$. Assume that $\mathcal{S}$ has at least one non-linear constraint. We start with the observation that for a given $\boldsymbol{y} \in \mathbb{R}^n$ the system (3) admits a solution $\boldsymbol{x} \in \mathbb{Z}^m$ if and only if the polyhedral set

$$P(\boldsymbol{y}) := \{\boldsymbol{x} \in \mathbb{R}^m : \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{x}^\top A_i \boldsymbol{y} + \boldsymbol{b}_i^\top \boldsymbol{y} \leq c_i,\ i = 1,\ldots,\ell\}\,, \tag{4}$$

contains an integer point.

Let H be an upper bound of the absolute value of the integer constants in the system (3). Since $\mathcal{S}$ is bounded, by [14, Lemma 3.1.25] the set $\{\boldsymbol{y} \in \mathbb{R}^n : D\boldsymbol{y} \leq \boldsymbol{e}\}$ is contained in the ball of radius $\kappa_1 := m^{1/2}H^{(m^2+m)}$ centred at the origin.

For a matrix A, let $\|A\|$ denote the spectral norm. Recall that if A has entries of absolute value at most H and has m columns then $\|A\| \leq \sqrt{m}H$. Now write

$$\delta := \min(\delta_0, 1), \quad \text{where } \delta_0 := \min\left\{\frac{\varepsilon}{\|A_i\|\kappa_1} : i = 1, \ldots, \ell\right\} \geq \frac{\varepsilon}{m^{1/2}H\kappa_1} \tag{5}$$

and define $U := \{\boldsymbol{u} \in \mathbb{Z}^m \setminus \{\boldsymbol{0}\} : 2\delta\|\boldsymbol{u}\| < \omega(m)\}$, where $\omega(m)$ is as in Theorem 7. Write $U = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s\}$ and consider the following *relaxed system* $\mathcal{S}'$ of linear and bilinear constraints in exclusively real variables (where the notation $P(\boldsymbol{y})$ is as in (4) and we use Proposition 8 to formulate the constraint $\text{width}_{\boldsymbol{u}_j}(P(\boldsymbol{y})) \geq \omega(m)$):

$$\begin{aligned}
&\boldsymbol{x}^\top A_i \boldsymbol{y} + \boldsymbol{b}_i^\top \boldsymbol{y} \leq c_i - \varepsilon \qquad (i = 1, \ldots, \ell)\\
&\text{width}_{\boldsymbol{u}_j}(P(\boldsymbol{y})) \geq \omega(m) \qquad (j = 1, \ldots, s)\\
&D\boldsymbol{y} \leq \boldsymbol{e},\ \boldsymbol{x} \geq \boldsymbol{1}\\
&\boldsymbol{x} \in \mathbb{R}^m,\ \boldsymbol{y} \in \mathbb{R}^n
\end{aligned} \tag{6}$$

▶ **Proposition 9.** *If the relaxed system $\mathcal{S}'$ is satisfiable, then so is the original system $\mathcal{S}$.*

Proof. Let $\boldsymbol{x}^*, \boldsymbol{y}^*$ be a solution of the system $\mathcal{S}'$, as shown in (6). Consider the set $P(\boldsymbol{y}^*)$ as defined in (4). By construction we have

$$\min_{\boldsymbol{u} \in U} \text{width}_{\boldsymbol{u}}(P(\boldsymbol{y}^*)) \geq \omega(m)\,. \tag{7}$$

But from the fact $\boldsymbol{x}^*$ satisfies each constraint $\boldsymbol{x}^\top A_i \boldsymbol{y}^* + \boldsymbol{b}_i^\top \boldsymbol{y}^* \leq c_i$ with slack ε and that $\boldsymbol{x}^* \geq \boldsymbol{1}$, we see that the ball $B_\delta(\boldsymbol{x}^*)$ is contained in $P(\boldsymbol{y}^*)$, for δ as defined in (5). It follows that

$$\begin{aligned}
\text{width}_{\boldsymbol{u}}(P(\boldsymbol{y}^*)) &\geq 2\delta\|u\|\\
&\geq \omega(m)
\end{aligned}$$

for all $\boldsymbol{u} \notin U$. Together with (7), we have that $\text{width}(P(\boldsymbol{y}^*)) \geq \omega(m)$ and hence, by Theorem 7, $P(\boldsymbol{y}^*)$ contains an integer point. This entails that the original system $\mathcal{S}$ is satisfiable. ◀

▶ **Proposition 10.** *If the relaxed system $\mathcal{S}'$ has no solution then every solution $\boldsymbol{x}^* \in \mathbb{Z}^m$ of the original system $\mathcal{S}$ that has slack ε either has some component equal to zero or satisfies $|\boldsymbol{u}^\top \boldsymbol{x}^*| \leq \kappa_2$ for some $\boldsymbol{u} \in U$, where κ_2 is an explicit constant depending only on $\mathcal{S}$ and ε.*

Proof. Assume that $\mathcal{S}'$ has no solution. Let $\boldsymbol{x}^* \in \mathbb{Z}^m$ and $\boldsymbol{y}^* \in \mathbb{R}^n$ be a solution of $\mathcal{S}$ with slack ε. If some component of $\boldsymbol{x}^*$ is zero then we are done, so we may suppose that $\boldsymbol{x}^* \geq \boldsymbol{1}$. By assumption, $\boldsymbol{x}^*, \boldsymbol{y}^*$ is not a solution of $\mathcal{S}'$ and so it must hold that

$$\min_{\boldsymbol{u} \in U} \text{width}_{\boldsymbol{u}}(P(\boldsymbol{y}^*)) < \omega(m)\,, \tag{8}$$

where $P(\boldsymbol{y}^*)$ is as defined in (4).

Let $\boldsymbol{u} \in U$ be the vector achieving the minimum on the left-hand side of (8). We will exhibit an upper bound on $|\boldsymbol{u}^\top \boldsymbol{x}^*|$ that does not depend on $\boldsymbol{y}^*$.

Assume first that $P(\boldsymbol{y}^*)$ contains the origin. Then by (8),

$$|\boldsymbol{u}^\top \boldsymbol{x}^*| = |\boldsymbol{u}^\top (\boldsymbol{x}^* - \boldsymbol{0})| \leq \omega(m)\,.$$

Assume now that $P(\boldsymbol{y}^*)$ does not contain the origin. Let L be the line segment connecting the origin to $\boldsymbol{x}^*$, and denote by $\boldsymbol{x}$ the point at which L intersects the boundary of $P(\boldsymbol{y}^*)$. Then we have $\boldsymbol{x}^* - \boldsymbol{x} = \lambda \boldsymbol{x}$ for some $\lambda > 0$. Moreover, since $\boldsymbol{x}$ lies on the boundary of $P(\boldsymbol{y}^*)$ there exists $i_0 \in \{1, \ldots, \ell\}$ such that

$$\boldsymbol{x}^\top A_{i_0} \boldsymbol{y}^* + \boldsymbol{b}_{i_0}^\top \boldsymbol{y}^* = c_{i_0}\,, \tag{9}$$

i.e., one of inequalities that define $P(\boldsymbol{y}^*)$ is tight at $\boldsymbol{x}$. But since $\boldsymbol{x}^*, \boldsymbol{y}^*$ satisfies $\mathcal{S}$ with slack ε, we also have that $(\boldsymbol{x}^*)^\top A_{i_0} \boldsymbol{y}^* + \boldsymbol{b}_{i_0}^\top \boldsymbol{y}^* \leq c_{i_0} - \varepsilon$. Subtracting Equation (9) from the previous inequality gives

$$\begin{aligned} -\varepsilon &\geq (\boldsymbol{x}^* - \boldsymbol{x})^\top A_{i_0} \boldsymbol{y}^* \\ &= \lambda(\boldsymbol{x}^\top A_{i_0} \boldsymbol{y}^*) \\ &= \lambda(c_{i_0} - \boldsymbol{b}_{i_0}^\top \boldsymbol{y}^*)\,. \end{aligned}$$

Since $\varepsilon, \lambda > 0$ this entails that $c_{i_0} - \boldsymbol{b}_{i_0}^\top \boldsymbol{y}^* < 0$ and hence

$$\begin{aligned} \lambda^{-1} &\leq \varepsilon^{-1} |c_{i_0} - \boldsymbol{b}_{i_0}^\top \boldsymbol{y}^*| \\ &\leq \varepsilon^{-1} \left(|c_{i_0}| + \|\boldsymbol{b}_{i_0}\| \kappa_1\right) \end{aligned} \tag{10}$$

We deduce that

$$\begin{aligned} |\boldsymbol{u}^\top \boldsymbol{x}^*| &\leq |\boldsymbol{u}^\top (\boldsymbol{x}^* - \boldsymbol{x})| + |\boldsymbol{u}^\top \boldsymbol{x}| \\ &= |\boldsymbol{u}^\top (\boldsymbol{x}^* - \boldsymbol{x})| \, (1 + \lambda^{-1}) \\ &\leq \omega(m) \left(1 + \varepsilon^{-1}(|c_{i_0}| + \|\boldsymbol{b}_{i_0}\| \kappa_1)\right) \qquad \text{by (8) and (10).} \end{aligned}$$

Thus, defining

$$\kappa_2 := \omega(m) \left(1 + H \varepsilon^{-1} (1 + m^{1/2} \kappa_1)\right), \tag{11}$$

we have $|\boldsymbol{u}^\top \boldsymbol{x}^*| \leq \kappa_2$.

In summary, we have that $|\boldsymbol{u}^\top \boldsymbol{x}^*| \leq \kappa_2$ for every integer point $\boldsymbol{x}^*$ of $P(\boldsymbol{y}^*)$, as required in the proposition. ◀

4.3 Decision Procedure

In this section we describe a decision procedure for the Gap Satisfiability Problem for bounded MIB systems. This is a recursive procedure based on the relaxation construction in the preceding section. We first present a conceptually simple version of the procedure, with no complexity bound, and then give a more detailed treatment from which bounds can be extracted.

▶ **Theorem 11.** *The Gap Satisfiability Problem is decidable for bounded MIB systems.*

Proof. The procedure to solve the Gap Satisfiability Problem is as follows. Consider an instance of the problem, consisting of an MIB system in the form (3) and slack $\varepsilon > 0$. If there are no non-linear constraints then the problem instance is just a system of linear inequalities in real and integer variables, whose satisfiability is straightforward to discern. Thus we may assume that there is at least one non-linear constraint. We construct the associated relaxed system $\mathcal{S}'$, which has the form (6). Using a decision procedure for the existential theory of real-closed fields we determine whether the system $\mathcal{S}'$ is satisfiable.

If $\mathcal{S}'$ is satisfiable then Proposition 9 guarantees that the original MIB system $\mathcal{S}$ is also satisfiable. We can then find a satisfying assignment of $\mathcal{S}$ by enumerating over all values $\boldsymbol{x}^* \in \mathbb{Z}^m$ and solving a linear program to decide whether there exists $\boldsymbol{y}^* \in \mathbb{R}^n$ such that $\boldsymbol{x}^*, \boldsymbol{y}^*$ satisfies $\mathcal{S}$.

If the relaxed problem has no satisfying assignment then Proposition 10 furnishes a finite set $\mathcal{E}$ of linear equations of the form $\boldsymbol{u}^\top \boldsymbol{x} = b$, with coefficients $\boldsymbol{u} \in \mathbb{Z}^m$ and $b \in \mathbb{Z}$, such that for any solution $\boldsymbol{x}^* \in \mathbb{Z}^m, \boldsymbol{y}^* \in \mathbb{R}^n$ of (2) that has slack ε, the integer part $\boldsymbol{x}^*$ satisfies an equation in $\mathcal{E}$. We iterate through all such equations $\boldsymbol{u}^\top \boldsymbol{x} = b$ and in each case we apply Proposition 5 to write

$$\{\boldsymbol{x} \in \mathbb{Z}^m : \boldsymbol{u}^\top \boldsymbol{x} = b,\, \boldsymbol{x} \geq \boldsymbol{0}\}$$

as a linear set $L(B, P)$ for some finite set $B \subseteq \mathbb{Z}^m$ and matrix $P \in \mathbb{Z}^{m \times m-1}$. Then for each vector $\boldsymbol{w} \in B$, we apply the change of variables $\boldsymbol{x} = \boldsymbol{w} + P\boldsymbol{z}$ to obtain a MIB system in one fewer integer variable to which we can recursively apply the procedure to determine satisfiability. ◀

In the following result we retrace the proof of Theorem 11, this time keeping track of the size of the integers involved. We thereby obtain an upper bound on the smallest satisfying assignment, showing that the gap satisfiability problem can be solved in nondeterministic exponential time.

▶ **Theorem 12.** *Consider a MIB system* (3) *in which the integer constants have absolute value at most* H*. If such a system is satisfiable with slack* ε *then there is a satisfying assignment under which the integer variables have absolute value at most* $2^{\kappa_3^{O(m^3(m+n))}}$*, where* $\kappa_3 := \left(\frac{mH^{m^2}}{\varepsilon}\right)$.

Proof. We first analyse the effect of a single variable-elimination step on the size of the integers in the system (3). Recall that to eliminate an integer variable we assert a linear equation $\boldsymbol{u}^\top \boldsymbol{x} = b$, where $\|\boldsymbol{u}\| \leq \frac{2w(m)}{\delta}$ and $|b| \leq \kappa_2$. Combining the lower bound $\delta \geq \frac{\varepsilon}{m^{1/2} H \kappa_1}$ from (5), the definition $\kappa_1 := m^{1/2} H^{(m^2+m)}$, the definition of κ_2 in (11), and the bound $\omega(m) = O(m^{3/2})$, we obtain that $\|\boldsymbol{u}\|, |b| = \kappa_3^{O(1)}$, for $\kappa_3 := \left(\frac{mH^{m^2}}{\varepsilon}\right)$.

Employing Proposition 5, the equation $\boldsymbol{u}^\top \boldsymbol{x} = b, \boldsymbol{x} \geq \boldsymbol{0}$, determines a substitution $\boldsymbol{x} = P\boldsymbol{z} + \boldsymbol{w}$ in which the elements of P and $\boldsymbol{w}$ have absolute value at most $\kappa_3^{O(m)}$. Since there are m integer variables, the constants appearing over all MIB instances arising through the process of variable elimination have absolute value at most $\kappa_3^{O(m^2)}$.

Consider a version of the relaxed system (6) in which the integer constants have magnitude at most $\kappa_3^{O(m^2)}$. For the purposes of our complexity analysis we augment the system with a new variable r and constraints $r \geq \|\boldsymbol{x}\| + 1$ and $r \geq \|\boldsymbol{w} \pm w(m)\boldsymbol{u}\|$ for each vertex $\boldsymbol{w}$ of the polyhedron $P(\boldsymbol{y})$ (as defined in (4)) and $\boldsymbol{u} \in U$. The integer constants in the resulting system have absolute value at most $\kappa_3^{O(m^3)}$ by Hadamard's determinant inequality. By construction, if $\boldsymbol{x}^*, \boldsymbol{y}^*$ is a satisfying assignment of (6) then the convex set $\{\boldsymbol{x} \in P(\boldsymbol{y}^*) : \|\boldsymbol{x}\| \leq r\}$ has lattice width at most $w(m)$ and hence contains an integer point. By Proposition 6 an upper bound for r is $2^{\kappa_3^{O(m^3(m+n))}}$, which concludes the proof. ◀

▶ Remark 13. It is evident that the double exponential dependence of the magnitude of the smallest satisfying assignment on the number of variables in Theorem 12 is unavoidable. Indeed, consider the following MIB system:

$$x_i y_i \leq 1 \quad (i = 1, \ldots, n)$$
$$x_{i+1} y_i \geq x_i y_0 \quad (i = 1, \ldots, n-1)$$
$$x_1 = 2, y_0 = 1$$
$$x_1, \ldots, x_n \in \mathbb{Z}_{\geq 0},\ y_0, \ldots, y_n \in \mathbb{R}_{\geq 0}$$

Then any satisfying assignment satisfies $x_{i+1} \geq \frac{x_i}{y_i} \geq x_i^2$ for $I = 1, \ldots, n-1$, whence $x_n \geq 2^{2^{n-1}}$. The system moreover has a satisfying assignment with slack ε for any $\varepsilon > 0$, obtained by successively setting $y_i := \frac{1+\varepsilon}{x_i}$ and $x_{i+1} := \lfloor \frac{x_i+\varepsilon}{y_i} \rfloor$ for $i = 1, \ldots, n-1$.

Proposition 1 and Corollary 2 give an exponential-time Turing reduction of the Gap Domination Problem for MPTA to the Gap Satisfiability Problem for bounded MIB systems, such that resulting instances of the Gap Satisfiability Problem have size polynomial in that of the input MPTA. We thus obtain our second main result.

▶ **Theorem 14.** *The Gap Domination Problem for MPTA is decidable in non-deterministic exponential time.*

5 Conclusion

Our main result shows that pareto curve of undominated reachable observer values of a given MPTA can be approximated to arbitrary precision. This is in contrast with the situation for weighted timed games, where it was recently shown that the optimal value of a weighted timed game with positive and negative rates cannot be computed to arbitrary precision [15].

Throughout this paper we have worked with MPTA with clock guards defined by conjunctions of non-strict inequalities. However, we claim that for an MPTA $\mathcal{A}$ with guards comprising conjunctions of both strict and non-strict inequalities, there exists an MPTA $\mathcal{A}'$ with exclusively closed guards over the same set $\mathcal{Y}$ of observers, such that every observer valuation $\gamma \in \mathbb{R}^{\mathcal{Y}}$ reachable in $\mathcal{A}$ is also reachable in $\mathcal{A}'$ and, conversely, for every valuation $\gamma' \in \mathbb{R}^{\mathcal{Y}}$ reachable in $\mathcal{A}'$ and every $\varepsilon > 0$ there exists a valuation $\gamma \in \mathbb{R}^{\mathcal{Y}}$ reachable in $\mathcal{A}$ such that $|\gamma(c) - \gamma'(c)| < \varepsilon$ for all $c \in \mathcal{Y}$. Indeed, such an MPTA $\mathcal{A}'$ is obtained by directly applying the closure construction for timed automata in [22, Section 4] to MPTA. Then the ability to compute the pareto curve of undominated reachable observer values of $\mathcal{A}'$ to arbitrary precision allows one to achieve the same end for $\mathcal{A}$.

A direction for future work is to consider the feasibility of approximate pareto analysis over infinite runs of MPTA. For double-priced timed automata, that is, MPTA with a single cost and reward observer, it is known how to compute the optimal reward-to-cost ratio over infinite computations using the corner-point abstraction [8]. For more general MPTA it is natural to consider specifications that refer to multiple reward-to-cost ratios.

References

1 R. Alur and D. Dill. A theory of timed automata. *TCS*, 126(2):183–235, 1994. doi:10.1016/0304-3975(94)90010-8.

2 R. Alur, S. La Torre, and G. J. Pappas. Optimal paths in weighted timed automata. In *HSCC*, volume 2034 of *LNCS*, pages 49–62. Springer, 2001. doi:10.1007/3-540-45351-2_8.

3 S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2006.

4 Devendra B., Krishna S., and Trivedi A. On nonlinear prices in timed automata. In *Proceedings of the The First Workshop on Verification and Validation of Cyber-Physical Systems, V2CPS@IFM*, volume 232 of *EPTCS*, pages 65–78, 2016. doi:10.4204/EPTCS.232.9.

5 W. Banaszczyk, A. E. Litvak, A. Pajor, and S. J Szarek. The flatness theorem for nonsymmetric convex bodies via the local theory of banach spaces. *Mathematics of operations research*, 24(3):728–750, 1999. doi:10.1287/MOOR.24.3.728.

6 G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. W. Vaandrager. Minimum-cost reachability for priced timed automata. In *HSCC*, volume 2034 of *LNCS*, pages 147–161. Springer, 2001. doi:10.1007/3-540-45351-2_15.

7 P. Bouyer, E. Brinksma, and K. G. Larsen. Optimal infinite scheduling for multi-priced timed automata. *Formal Methods in System Design*, 32(1):3–23, 2008. doi:10.1007/S10703-007-0043-4.

8 P. Bouyer, K. G. Larsen, and N. Markey. Model checking one-clock priced timed automata. *Logical Methods in Computer Science*, 4:1–28, 2008.

9 Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, and Nicolas Markey. Quantitative analysis of real-time systems using priced timed automata. *Commun. ACM*, 54(9):78–87, 2011. doi:10.1145/1995376.1995396.

10 T. Brihaye, V. Bruyère, and J.-F. Raskin. On model-checking timed automata with stopwatch observers. *Inf. Comput.*, 204(3):408–433, 2006. doi:10.1016/J.IC.2005.12.001.

11 I. Diakonikolas and M. Yannakakis. Small approximate pareto sets for biobjective shortest paths and other problems. *SIAM J. Comput.*, 39(4):1340–1371, 2009. doi:10.1137/080724514.

12 M Fränzle, M Shirmohammadi, M Swaminathan, and J Worrell. Costs and rewards in priced timed automata. *Inf. Comput.*, 282:104656, 2022. doi:10.1016/J.IC.2020.104656.

13 M. Fränzle and M. Swaminathan. Revisiting decidability and optimum reachability for multi-priced timed automata. In *The 7th International Conference on Formal Modelling and Analysis of Timed Systems*, pages 149–163. Springer Verlag, September 2009.

14 M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

15 Q. Guilmant and J. Ouaknine. Inapproximability in weighted timed games. In *Proceedings of CONCUR 24*, volume 311 of *LIPIcs*, 2024.

16 R. Hemmecke, M. Köppe, J. Lee, and R. Weismantel. *Nonlinear integer programming*. Springer, 2010.

17 T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *J. Comput. Syst. Sci.*, 57(1):94–124, 1998. doi:10.1006/JCSS.1998.1581.

18 R. Kannan and L. Lovász. Covering minima and lattice-point-free convex bodies. *Annals of Mathematics*, pages 577–602, 1988.

19 L. Khachiyan and L. Porkolab. Integer optimization on convex semialgebraic sets. *Discrete & Computational Geometry*, 23:207–224, 2000. doi:10.1007/PL00009496.

20 K. G. Larsen, G. Behrmann, E. Brinksma, A. Fehnker, T. Hune, P. Pettersson, and J. Romijn. As cheap as possible: Efficient cost-optimal reachability for priced timed automata. In *CAV*, volume 2102 of *LNCS*, pages 493–505. Springer, 2001. doi:10.1007/3-540-44585-4_47.

21 K. G. Larsen and J. I. Rasmussen. Optimal reachability for multi-priced timed automata. *TCS*, 390(2-3):197–213, 2008. doi:10.1016/J.TCS.2007.09.021.

22 J. Ouaknine and J. Worrell. Revisiting digitization, robustness, and decidability for timed automata. In *18th IEEE Symposium on Logic in Computer Science (LICS, Proceedings*, pages 198–207. IEEE Computer Society, 2003. doi:10.1109/LICS.2003.1210059.

23 L. Pottier. Minimal solutions of linear diophantine systems: bounds and algorithms. In *International Conference on Rewriting Techniques and Applications*, pages 162–173. Springer, 1991.

24 M. Schaefer and D. Stefankovic. Fixed points, nash equilibria, and the existential theory of the reals. *Theory Comput. Syst.*, 60(2):172–193, 2017. doi:10.1007/S00224-015-9662-0.

25 R. G. Tollund, N. S. Johansen, K. Ø. Nielsen, A. Torralba, and K. G. Larsen. Optimal infinite temporal planning: Cyclic plans for priced timed automata. In *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling, ICAPS*, pages 588–596. AAAI Press, 2024. doi:10.1609/ICAPS.V34I1.31521.

26 Z. Zhang, B. Nielsen, K. G. Larsen, G. Nies, M. Stenger, and H. Hermanns. Pareto optimal reachability analysis for simple priced timed automata. In *ICFEM*, volume 10610 of *LNCS*, pages 481–495. Springer, 2017. doi:10.1007/978-3-319-68690-5_29.

Two-Way One-Counter Nets Revisited

Shaull Almagor
Department of Computer Science, Technion, Haifa, Israel

Michaël Cadilhac
DePaul University, Chicago, IL, USA

Asaf Yeshurun
Department of Computer Science, Technion, Haifa, Israel

Abstract

One Counter Nets (OCNs) are finite-state automata equipped with a counter that cannot become negative, but cannot be explicitly tested for zero. Their close connection to various other models (e.g., PDAs, Vector Addition Systems, and Counter Automata) make them an attractive modeling tool.

The two-way variant of OCNs (2-OCNs) was introduced in the 1980's and shown to be more expressive than OCNs, so much so that the emptiness problem is undecidable already in the deterministic model (2-DOCNs).

In a first part, we study the emptiness problem of natural restrictions of 2-OCNs, under the light of modern results about Vector Addition System with States (VASS). We show that emptiness is decidable for 2-OCNs over *bounded languages* (i.e., languages contained in $a_1^* a_2^* \cdots a_k^*$), and decidable and Ackermann-complete for *sweeping* 2-OCNs, where the head direction only changes at the end-markers. Both decidability results revolve around reducing the problem to VASS reachability, but they rely on strikingly different approaches. In a second part, we study the expressive power of 2-OCNs, showing an array of connections between bounded languages, sweeping 2-OCNs, and semilinear languages. Most noteworthy among these connections, is that the bounded languages recognized by sweeping 2-OCNs are precisely those that are semilinear. Finally, we establish an intricate pumping lemma for 2-DOCNs and use it to show that there are OCN languages that are not 2-DOCN recognizable, improving on the known result that there are such 2-OCN languages.

2012 ACM Subject Classification Theory of computation → Automata extensions

Keywords and phrases Counter Net, Two way, Automata

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.19

Related Version *Full Version*: `https://arxiv.org/abs/2410.22845`

Funding *Shaull Almagor*: supported by the ISRAEL SCIENCE FOUNDATION (grant No. 989/22).

Acknowledgements We are grateful to Dmitry Chistikov for shedding light on some claims made in [4] and to an anonymous reviewer for providing some key references.

1 Introduction

A *One-Counter Net (OCN)* is a finite state automaton equipped with a counter that cannot decrease below zero, but cannot be explicitly tested for zero. It is a natural restriction of several computational models: One-Counter Automata without zero tests, and Pushdown Automata with a single letter stack alphabet. It can also be thought of as 1-dimensional Vector Addition Systems with accepting States and an alphabet.

OCNs are an attractive model for studying the border of decidability, as several problems for them lie close to the decidability frontier (e.g., both determinization and universality may be decidable or undecidable, depending on the precise definition and context [16, 17, 2, 3, 1]).

OCNs suffer from an unsettling asymmetry: the language $L = \{a^n b^m \mid n \geq m\}$ is OCN-recognizable (by counting $+1$ on a and -1 on b), whereas the language $\{a^n b^m \mid n \leq m\}$ is not OCN-recognizable. In particular, they are not closed under reversal (nor under intersection). A natural way of making OCNs more robust is therefore to look at their two-way variant.

This approach was taken in [26, 7], where the model of *two-way one-counter nets (2-OCNs)* is introduced and studies. A 2-OCN is a one counter net that receives its input on a tape, surrounded by end-markers $\vdash$ and $\dashv$, and is allowed to move a read-only head back and forth on the tape. As the following examples witness, the introduction of a two-way tape significantly increases the expressive power of the model, as well as its deterministic fragment *(2-DOCN)*. For uniformity, we henceforth refer to one-way OCNs as *1-OCN*.

▶ **Example 1.** Consider the language $L_1 = \{a^n b^n \mid n \in \mathbb{N}\}$, which is easily shown not to be 1-OCN-recognizable. L_1 can be recognized by a 2-OCN (in fact 2-DOCN, see Fig. 1) by using a "counter reset": it starts with a forward scan verifying that the format is $a^n b^m$ while counting $+1$ on a and -1 on b. Thus, upon reaching $\dashv$, the counter has value $n - m$ (unless $n < m$, in which case the run terminates due to the counter becoming negative, and the word is not accepted), and in particular $n \geq m$. It then moves backwards to $\vdash$, counting -1 on a and $+1$ on b, thus resetting the counter back to 0. Next, it goes forward again to $\dashv$, then similarly computes $m - n$ from right to left. Intuitively, this approach recognizes $\{a^n b^m \mid n \geq m\} \cap \{a^n b^m \mid n \leq m\}$ and uses the closure of 2-DOCN under intersection [26].

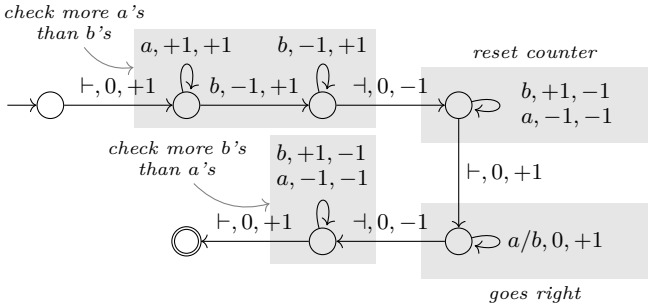


Figure 1 2-DOCN recognizing $\{a^n b^n \mid n \in \mathbb{N}\}$ using counter-reset. A transition σ, e, h means "read letter σ, change counter by e, and move head by h".

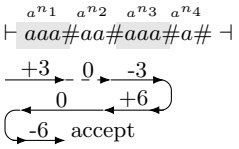


Figure 2 An accepting run of a 2-OCN for L_2. The dashed line signifies a nondeterministic guess.

The next example demonstrates the fact [7] that 2-OCN are more expressive than 2-DOCN. In addition, it demonstrates that sometimes it is (seemingly) necessary for the head to "change direction" inside the tape, rather than at the end markers.

▶ **Example 2.** Let L_2 be the language $\{a^{n_1}\# \cdots \#a^{n_k}\# \mid k \in \mathbb{N} \wedge (\exists i > 1)[n_1 = n_i]\}$, i.e., words where the length of the first a-segment is matched in some later segment. This language is not 1-OCN recognizable, and one may observe that a similar approach to Theorem 1 fails, because we would need to guess the index i for which $n_1 = n_i$, but in order to reset the counter, we must "forget" the guess. Thus, in order to recognize this with a 2-OCN, we must compare n_1 to n_i via a single visit to the i-th segment.

To achieve this, the 2-OCN counts $+1$ on the first segment a^{n_1}, then scans forward and guesses when the correct segment i is reached. It then counts -1 on a^{n_i}. Thus, upon reaching $\#$, the counter holds $n_1 - n_i$ (and so guarantees that $n_1 \geq n_i$ if it survives). Then, a backward scan is performed from $\#$ counting $+2$ on a in a^{n_i}, until $\#$ is reached, and then

continues left to $\vdash$ and finally counts -2 on the a's in a^{n_1}. Thus, the counter ends with value $n_1 - n_i + 2n_i - 2n_1 = n_i - n_1$. Again, if $n_i < n_1$ the run terminates, so the whole run survives if $n_1 \leq n_i$, which combined with $n_1 \geq n_i$ gives $n_1 = n_i$ (see Fig. 2).

A 2-OCN that only changes its head direction at the end marks is called *sweeping*. Theorem 2 prompts studying the expressive power of sweeping 2-OCNs, which we take on in this work.

Related Work. Two-way automata have received much attention since their introduction for finite automata in [30]. 2-PDAs were studied in [24, 14, 4, 27], focusing mainly on closure properties and languages recognizable by 2-PDAs. As noted in [4], languages not expressible by 2-PDA can be derived from the complexity results obtained in [14] (specifically, 2-PDA languages are in linear space and polynomial time, and the time/space hierarchy theorems offer languages outside of these classes). No automata-based, combinatorial techniques are known to the authors to show that some languages is not expressible with a 2-PDA, and most questions about 2-PDA, including whether the deterministic variant can recognize all context-free languages, are open [13].

In the world of counter machines, [12] shows that DFAs with a "blind tape" (a form of a nonnegative counter that cannot be 0-tested) are less expressive than 2-DPDAs. A significant body of work by Ibarra et al. studies *reversal bounded* counters [19, 21, 18, 10, 20], i.e., counters that cannot increase and decrease unboundedly many times. The heavy reliance on reversal-boundedness makes these works orthogonal to ours. Atig and Ganty [5] study a VASS-CFG hybrid, and use also make use of reductions to VASS with 0-tests, but with significant technical differences.

Closer to our setting are *two-way one-counter automata* (2-OCAs), that allow explicit zero tests. In [29] certain languages computing squaring or exponentiation are shown not to be 2-DOCA recognizable (the deterministic variant), whereas [11] exhibit nonsemilinear 2-OCA languages. Ibarra and Su [21] note that 1-sweep 2-DOCA have an undecidable emptiness problem; in contrast, we show that this problem is decidable for sweeping 2-OCN. They also note that, over bounded languages, emptiness of 2-DOCA is decidable *provided* that the counter reaches zero only a constant number of times; in contrast, we show that this emptiness over bounded languages is decidable for any 2-OCN.

A nearly identical model to 2-OCN has been introduced in [26] and further studied in [7]. In their model, the counter updates are ± 1, which essentially amounts to encoding counters in unary. In [26], the focus is on 2-DOCN recognizable languages. It is shown that this class is closed under intersection, but not under complementation nor union. Further, over bounded languages, 2-DOCN can be made sweeping[1], and the Parikh images of their languages are semilinear. An example is also given to show that 2-DOCN are less expressive than 2-OCN.

The algorithmic contribution in [7] is that the emptiness problem for 2-DOCN is undecidable. Additionally, it is shown that if a 2-OCN accepts a word of length n, then it accepts it with a run of polynomial length in n. The latter result no longer holds for our model, due to the binary encoding of the counters.

Contribution and Paper Organization. In this work we provide a modern view of 2-OCNs and some of its interesting and decidable fragments, namely *bounded-language 2-OCNs* and *sweeping 2-OCNs*. In Section 3.1 we give a counter-machine based proof of the undecidability of 2-DOCN emptiness. In Section 3.2, we consider 2-OCNs whose languages are *bounded*, i.e.,

[1] The proof in [26] is actually missing significant details, but those can be reconstructed.

of the form $L \subseteq a_1^* a_2^* \cdots a_k^*$. We show that under this restriction, emptiness becomes decidable. In Section 3.3, we consider *sweeping* 2-OCNs, i.e., 2-OCNs whose head-movement direction only changes at the end-markers. We show that here too emptiness becomes decidable and we give precise complexity bounds, using recent results regarding VASS reachability. In Section 4, we investigate the relationship between semilinear languages and 2-OCN languages. We show that for bounded languages, the set of *lengths* of words in the language of a 2-OCN is semilinear, and derive explicit languages that are not 2-OCN-recognizable. We also show that a bounded language is semilinear if and only if it is recognizable by a sweeping 2-OCN. In Section 5, we show an elaborate pumping lemma for 2-DOCNs. Apart from its independent usefulness for understanding 2-DOCNs, this enables us to demonstrate a 1-OCN whose language is not 2-DOCN recognizable, establishing a new separation result. We conclude in Section 6 with some open problems.

We encourage the reader to initially skip the proofs and garner a bird's eye view of our results. Thereafter, we note that Section 3.2 and Theorems 14, 15, and 17 have the most interesting proofs. Detailed proofs appear in the appendix or in the full version.

2 Preliminaries

Two-Way OCN and Automata. For an alphabet set Σ, we denote $\Sigma_{\vdash\dashv} = \Sigma \cup \{\vdash, \dashv\}$ where $\vdash, \dashv$ are two symbols not appearing in Σ, called *left end-marker* and *right end-marker*, respectively. A *Two-Way One-Counter Net (2-OCN)* is a tuple $\mathcal{A} = \langle Q, \Sigma, \Delta, q_{\text{init}}, q_{\text{acc}} \rangle$ where Q is a finite set of *states*, Σ is a finite alphabet, $\Delta \subseteq Q \times \Sigma_{\vdash\dashv} \times \mathbb{Z} \times \{-1, +1\} \times Q$ is a finite set of *transitions*, $q_{\text{init}}, q_{\text{acc}} \in Q$ are the initial and accepting states, respectively. Intuitively, a transition $\tau = (q, \sigma, e, h, q')$ dictates that from state q, when reading letter σ, we add e to the counter value, move the position of the head on the tape by h, and reach state q'. We call e the *effect* and h the *head shift* of the transition. Unless explicitly stated otherwise, we assume that the effect e is encoded in binary. We require that if $(q, \vdash, e, h, q') \in \Delta$ then $h = 1$ and if $(q, \dashv, e, h, q') \in \Delta$ then $h = -1$. That is, upon reaching the end-markers, the head cannot proceed beyond them.

We say that $\mathcal{A}$ is *deterministic* (denoted 2-DOCN) if for every $q \in Q, \sigma \in \Sigma_{\vdash,\dashv}$ there is at most one $q' \in Q$ and e, h such that (q, σ, e, h, q'). We say that $\mathcal{A}$ is *one way* (denoted 1-OCN) if every transition moves the head to the right (i.e., the head shift is 1).

A *configuration* of $\mathcal{A}$ is $(q, c, p) \in Q \times \mathbb{N} \times \mathbb{N}$ representing the current state q, counter value c, and head position p. Note that we do not consider configurations with negative counter values, thus enforcing the semantics of Counter Nets. Consider a word $w = \sigma_1 \cdots \sigma_m \in \Sigma^*$, we augment it to $\vdash w \dashv$ by setting $\sigma_0 = \vdash$ and $\sigma_{m+1} = \dashv$. A *run* of $\mathcal{A}$ on w is a finite sequence of configurations $\rho = (q_1, c_1, p_1), (q_2, c_2, p_2), \ldots, (q_n, c_n, p_n)$ such that $0 \leq p_i \leq m + 1$ for all i, and for every $1 \leq i < n$ we have $(q_i, \sigma_{p_i}, c_{i+1} - c_i, p_{i+1} - p_i, q_{i+1}) \in \Delta$. That is, each configuration follows from the previous one by the transition relation. A run is *initial* if $q_1 = q_{\text{init}}$ and $p_1 = c_1 = 0$. A run is *accepting* if $q_n = q_{\text{acc}}$. We tacitly assume that runs do not continue after the accepting state. The word w is accepted by $\mathcal{A}$ if $\mathcal{A}$ has an initial and accepting run on it. We define $L(\mathcal{A}) = \{w \in \Sigma^* \mid \mathcal{A} \text{ accepts } w\}$. Note that, just like OCNs, 2-OCNs are *monotone*, i.e., every word accepted from initial configuration (q, c, p) is also accepted from (q, c', p) for every $c' > c$. We sometimes consider sequences of pseudo-configurations where the counter value becomes negative. In such cases, we say that the run is *cut short by a counter violation*, and we mean that the sequence of configurations is a run up to a prefix in which the counter becomes negative.

Finally, we sometimes discuss classical Boolean automata (NFA, DFA, 2-NFA and 2-DFA). For our purposes, they can be thought of as 2-OCN where all the counter updates are 0, and are therefore omitted.

VASS and VASS with bounded 0-tests. A *Vector Addition System with States (VASS)* is a tuple $\mathcal{V} = \langle S, T\rangle$ where S is a finite set of states and $T \subseteq S \times \mathbb{Z}^d \times S$ is the transition relation. A *configuration* of T is $(s, \boldsymbol{v})$ where $s \in S$ and $\boldsymbol{v} \in \mathbb{N}^d$ (note that this is a vector of nonnegative integers). We say that configuration $(s_2, \boldsymbol{y})$ is *reachable* from configuration $(s_1, \boldsymbol{x})$ if there is a finite sequence of configurations $(p_1, \boldsymbol{v_1}), \ldots, (p_m, \boldsymbol{v_m})$ such that $(p_1, \boldsymbol{v_1}) = (s_2, \boldsymbol{y}), (p_m, \boldsymbol{v_m}) = (s_2, \boldsymbol{y})$ and for every $1 \leq i < m$ we have $(p_i, \boldsymbol{v_{i+1}} - \boldsymbol{v_i}, p_{i+1}) \in T$. Intuitively, a VASS runs by repeatedly adding vectors to the configuration, as long as all the *counters* (i.e., the entries of the vector) remain nonnegative. The reachability problem for VASS asks whether $(s_2, \boldsymbol{y})$ is reachable from $(s_1, \boldsymbol{x})$ and is known to be decidable [9].

While VASS do not allow explicit 0-tests, it is possible to simulate a fixed number of 0-tests within the context of reachability [8, 9]. That is, we can extend the transition relation so that $T \subseteq S \times \mathbb{Z}^d \times \{=0, \geq 0\}^d \times S$, and a transition can be taken only when the counters that have $=0$ tests are 0. As long as there is a constant B such that there are no more than B 0-tests along every run of the VASS, then reachability remains decidable. The simplest approach to achieve this is to increase the dimension by $d \cdot B$, keeping B copies of every counter, and whenever a counter needs to be tested for 0, a copy of it is "frozen" (i.e., never changes again) and the reachability target has 0 on all frozen entries. Note that we can mark which entries are frozen in the states of the VASS, since there are at most B of them. We remark that more elaborate approaches can keep the number of added counters small [8, 9]. We call such machines *VASS with bounded 0-tests*.

3 The Emptiness and Membership Problems in 2-OCNs and Variants

3.1 Emptiness of 2-DOCNs is Undecidable

It is shown in [7] that already for 2-DOCNs, the emptiness problem, namely the problem of deciding whether $L(\mathcal{D}) = \emptyset$ for a given 2-DOCN $\mathcal{D}$, is undecidable. This is shown by a reduction from Hilbert's 10th problem. More precisely, it is shown that 2-DOCN can simulate multiplication, in a sense.

An arguably cleaner way of obtaining this result while staying in the world of counter machines, is via reduction from the halting problem for Two-Counter Machines, as follows:

▶ **Theorem 3** ([7]). *The emptiness problem for 2-DOCN is undecidable.*

Proof. We show the undecidability of 2-DOCN emptiness by reduction from the (complement of the) halting problem for two-counter machines, which is known to be undecidable [25].

Given a two-counter machine $\mathcal{M}$, we construct a 2-DOCA $\mathcal{A}$ which reads input words of the form $\#(x^*y^*\#)^*$, representing the values of the counters x and y along the run of $\mathcal{M}$, with each step of the machine separated with $\#$. Intuitively, $\mathcal{A}$ tracks the location of $\mathcal{M}$ in its state, uses the counter values to determine the location after a jump, and at each steps checks that the counter values between steps are consistent, similarly to Theorem 1. Then, $\mathcal{A}$ accepts if at any point the computation reaches the location `halt`. If a counter violation is encountered, $\mathcal{A}$ moves to a rejecting sink.

More precisely, $\mathcal{A}$ starts by checking (using a single right-to-left pass with a DFA) that the word adheres to the format $\#(x^*y^*\#)^*$. It then resets the head to $\vdash$, records the location l_1 in its state, moves the head to the first $\#$ and begins the simulation. $\mathcal{A}$ simulates each step

starting from the # preceding the segment $x^m y^k$ representing the current counter values, and along the simulation scans only the segment of the word of the form $\#x^m y^k \# x^{m'} y^{m'} \#$ starting at the current head location. Note that crucially, the format of the segment is fixed, so that the head can end the simulation in the middle #, thus correctly viewing the counter values of the next step.

If the command at the current location is `goto` l_i, then $\mathcal{A}$ checks that $m = m'$ and $k = k'$, similarly to Theorem 1. Similarly, if the current command is `inc(x)` or `dec(y)`, then $\mathcal{A}$ checks that $k' = k + 1$ and $m = m'$, or that $k = k'$ and $m' = m - 1$, respectively, by a similar comparison prefixed by a ± 1 counter change.

If the current command is `if x=0 goto` l_i `else goto` l_j, then $\mathcal{A}$ checks that $k = k'$ and $m = m'$, and then checks whether $k = 0$ (by checking if there are any x's after the first # in the segment), and if so moves to l_i and otherwise to l_j.

Finally, if $\mathcal{A}$ ever reaches the command `halt`, it accepts, and if any of the comparisons fails, the counter goes below 0 and the word is rejected. If $\dashv$ is reached without reaching `halt`, then $\mathcal{A}$ moves to a rejecting sink.

We then have that $L(\mathcal{A}) \neq \emptyset$ if and only if there exists a word representing the counter values in a halting run of $\mathcal{M}$, if and only if $\mathcal{M}$ halts. ◀

3.2 Emptiness of 2-OCNs over Bounded Languages is Decidable

Consider an alphabet $\Sigma = \{a_1, \ldots, a_k\}$. A language L is *bounded* if $L \subseteq a_1^* a_2^* \cdots a_k^*$. In this section, we show that the emptiness problem of 2-OCNs over bounded languages is decidable.

▶ Remark 4. Generally, a bounded language is a subset of $x_1^* x_2^* \cdots x_k^*$ where each $x_i \in \Sigma^*$ (in particular, e.g., $a^* b^* a^*$ is also bounded). Our results readily extend to these languages, but they add a cumbersome notational layer, which we opt to avoid.

▶ **Theorem 5.** *The following problem is decidable: given a 2-OCN $\mathcal{A}$ is $L(\mathcal{A}) \cap a_1^* a_2^* \cdots a_k^* = \emptyset$?*

We prove Theorem 5 in the remainder of this section. The proof is by reduction to the VASS nonreachability problem, known to be Ackermann-complete [9].

Consider a 2-OCN $\mathcal{A}$. We henceforth assume that membership in $a_1^* a_2^* \cdots a_k^*$ is already tested within $\mathcal{A}$, i.e., that $L(\mathcal{A})$ is a bounded language. We further assume that $L(\mathcal{A}) \subseteq a_1^+ a_2^+ \cdots a_k^+$, namely that every letter occurs at least once in every word in L (dubbed *properly-bounded*). Indeed, for the purpose of emptiness we can split $L(\mathcal{A})$ to a union over all subsets $\Gamma \subseteq \Sigma$ of properly-bounded languages over Γ.

We obtain from $\mathcal{A}$ a VASS with bounded 0-tests $\mathcal{V}$ such that $L(\mathcal{A}) \neq \emptyset$ iff $(s_{\text{init}}, \mathbf{0})$ reaches $(s_{\text{fin}}, \mathbf{0})$ for certain states $s_{\text{init}}, s_{\text{fin}}$ of $\mathcal{V}$.

Construction of $\mathcal{V}$ – Intuitive Overview

We present the intuition behind $\mathcal{V}$. The formal construction can be found in Appendix A. $\mathcal{V}$ simulates the behavior of $\mathcal{A}$ in several components using several counters. We start by describing the counters of $\mathcal{V}$ and their intuitive roles.

- `c` – tracking the counter of $\mathcal{A}$
- `c_freeze_1` – a copy of `c`, to be frozen at the beginning of a positive loop.
- `c_freeze_2` – a copy of `c`, to be frozen at the end of a positive loop.
- `L[1]` ... `L[k]` and `R[1]` ... `R[k]` – pairs of counters marking the position of the head from the `L`eft and from the `R`ight for each segment a_i^*.

The operation of $\mathcal{V}$ is as follows. The *initialization component* guesses a word $a_1^{n_1} \cdots a_k^{n_k}$ by guessing the numbers n_i and storing them in the `R[i]` counters of $\mathcal{V}$. This is done with self loops that allow to arbitrarily increase each of the relevant counters.

Once initialization is complete, $\mathcal{V}$ moves to the *simulation component*. There, $\mathcal{V}$ aims to simulate runs of $\mathcal{A}$ on the guessed word. Simulating the counter of $\mathcal{A}$ is straightforward, using the counter `c` to match it. In addition, if at any point the accepting state of $\mathcal{A}$ is reached, then $\mathcal{V}$ moves to the *finalizing component* (described below). In order to simulate the head movement, we refer to each $a_i^{n_i}$ as a *segment* (treating $\vdash$ and $\dashv$ also as segments), and we split the simulation to two types of moves:

- A *move within a segment* simulates $\mathcal{A}$ reading a_i and the head staying within the a_i segment. To do this, we set the two counters for the a_i segment `L[i]`, `R[i]` such that `L[i]` keeps track of the head location within the segment $a_i^{n_i}$ and `R[i]` $= n_i -$ `L[i]` (using the initialization counter). To simulate a move to the right we increase `L[i]` by 1 and decrease `R[i]` by 1, and vice-versa for a move left. Note that since n_i is constant after initialization, the VASS semantics prevents the simulation from going over the segment borders.
- A *move between segments* occurs when $\mathcal{V}$ guesses that a border of the a_i segment is reached, and an adjacent segment should be moved to. If the move is to the right, then `R[i]` should be 0, and we therefore perform a 0-test on it, and move to segment a_{i+1}. Note that from then on, the head movement counters are `L[i+1]` and `R[i+1]` (until a_i is reached again). This segment index is stored in the states of $\mathcal{V}$.
  Since we are only allowed a fixed number of 0-tests, we store in the states how many moves from segment a_i to a_{i+1} were performed. If this number exceeds $|Q| + 1$, where $|Q|$ is the number of states of $\mathcal{A}$, then the run cannot proceed (it reaches a sink). We justify this in the proof.

An accepting run of $\mathcal{A}$ might require the counter to be "charged" by repeating a loop, possibly involving moves between segments. Since we cannot simulate this with a bounded number of 0-tests, we instead detect loops, as follows. While moving between segments, $\mathcal{V}$ may guess that the current move leads to a repetition of the state and head position in the run of $\mathcal{V}$. Thus, $\mathcal{V}$ may nondeterministically record in its state the current state q of $\mathcal{A}$ (i.e., the state from which the move occurs), and also freezes a copy of the current counter of $\mathcal{A}$ in `c_freeze_1`. Then, upon another move between the same two segments, $\mathcal{V}$ may check whether the current state is again q. If so, $\mathcal{V}$ records the counter at the second move as well in `c_freeze_2`, at which point the simulation moves to the *2-NFA Phase*.

The *2-NFA Phase* assumes (and later verifies) that during its run, $\mathcal{A}$ encountered a positive loop, i.e., a path from configuration (q, c, p) to (q, c', p) with $c' > c$. In this case, $\mathcal{A}$ can charge the counter to an arbitrarily large value. Then, the word is accepted if it is accepted in the 2-NFA obtained from $\mathcal{A}$ by ignoring the counter, starting from (q, p). In this phase we repeat the simulation phase above, ignoring the counter of $\mathcal{A}$, but still limiting to $|Q|$ moves between each two segments, with the reasoning that more than $|Q|$ moves imply a cycle, and therefore if there is an accepting run, there is also a shorter one.

Finally, upon encountering the accepting state of $\mathcal{A}$, we move to the *finalizing component*. There, it remains to verify that if the 2-NFA phase was reached, then indeed a positive loop was encountered. To do so, the two frozen counter values of the loop (`c_freeze_1`, `c_freeze_2`) are decreased simultaneously, and at some point a nondeterministic transition decreases only the (presumably higher) counter `c_freeze_2` and moves to the state s_{fin}. Then, all the counters are allowed to be decreased, so that we can reach $(s_{\text{fin}}, \mathbf{0})$. This ensures that $(s_{\text{fin}}, \mathbf{0})$ only if `c_freeze_2` was indeed higher than `c_freeze_1`, witnessing that a positive loop was encountered.

3.3 Emptiness of Sweeping 2-OCNs is Decidable

In this section we consider the sweeping fragment of 2-OCNs. A run ρ of a 2-OCN is said to be *sweeping* if the head shift of the transitions changes only at the end-markers (i.e., the run "sweeps" forward from $\vdash$ to $\dashv$, and then sweeps backward to $\vdash$). A 2-OCN is *sweeping* if all its runs are sweeping. We assume w.l.o.g. that a sweeping 2-OCN only accepts when reading $\vdash$ (otherwise we add states to complete a backward sweep and then accept). We show that for sweeping 2-OCN, the emptiness problem is Ackermann-complete. The proof is split to the upper bound (Theorem 6) and lower bound (Theorem 7).

▶ **Theorem 6.** *The emptiness problem for sweeping 2-OCNs is decidable in Ackermannian complexity.*

Proof. We obtain the Ackermann upper bound of Theorem 6 by reducing nonemptiness of Sweeping 2-OCN to VASS-reachability. Similarly to Theorem 5, our proof relies on the observation that once a positive loop is encountered, we can replace the 2-OCN with a Boolean automaton. The challenge here, however, is that the language is not bounded, and therefore we cannot keep track of the head in a fixed number of segments. Instead, we use the sweeping property to simulate all the sweeps of the 2-OCN within a single pass, in a radically different approach than in the previous section.

Specifically, Consider a sweeping 2-OCN $\mathcal{A} = \langle Q, \Sigma, \Delta, q_{\text{init}}, q_{\text{acc}} \rangle$. We obtain from $\mathcal{A}$ a VASS with bounded 0-tests $\mathcal{V}$ such that $L(\mathcal{A}) \neq \emptyset$ iff $(s_{\text{init}}, \mathbf{0})$ reaches $(s_{\text{fin}}, \mathbf{0})$ for certain states $s_{\text{init}}, s_{\text{fin}}$ of $\mathcal{V}$. Moreover, this can be done in single-exponential time, and the size of $\mathcal{V}$ is single-exponential in that of $\mathcal{A}$.

The formal construction appears in Appendix B. We present the intuition here.

Construction of $\mathcal{V}$ – Intuitive Overview. As an example, assume that $\mathcal{A}$ accepts a word $w \in \Sigma^*$ using two forward sweeps and two backward sweeps, as depicted in Fig 3.

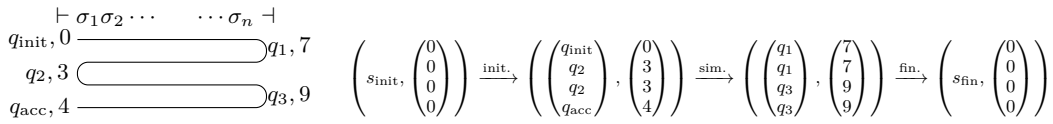


Figure 3 Left: A run of a sweeping 2-OCN. The pairs represent the state and counter value at the end of each sweep. Right: The corresponding successful run of $\mathcal{V}$.

We can simulate the entire run of $\mathcal{A}$ on w using a VASS $\mathcal{V}$ by, intuitively, tracking all the forward sweeps in parallel with all the backward sweeps, where for the backward sweeps we simulate the *reverse* 2-OCN of $\mathcal{A}$, denoted $\mathcal{A}^R$. The latter is a 2-OCN obtained from reversing the transitions, as well as negating the counter effects and the head shifts in all transitions. In order to make sure the sweeps concatenate correctly, we guess the states and the counter values. Specifically, we proceed as follows. We start by guessing the state in which each backward sweep ends. In this case, a successful guess is $(q_{\text{init}}, q_2, q_{\text{acc}})$ (note that all guesses must start and end in the initial and accepting state). We then guess the counter values with which each backward sweep ends. We store two copies of each guess, one used to simulate the backward run, and the other to simulate the next forward run (except for the last sweep). In our example, the successful guess is $(0, 3, 3, 4)$ (where 0 is the initial counter for the first forward sweep).

At each step, we now nondeterministically guess letters and corresponding transitions from $\mathcal{A}$ for the forward sweep, and from $\mathcal{A}^R$ for the backward sweep. After guessing the correct word w and keeping track of the counters in all components, we would then reach the

counters $(7, 7, 9, 9)$. Indeed, the forward run from 0 reaches counter 7, whereas the backward run from 3 starts from counter 7, and hence $\mathcal{A}^R$ would yield 7 along the reversal of the run. In addition, we track the states of $\mathcal{A}$ and of $\mathcal{A}^R$ along each run.

The last component of $\mathcal{V}$ verifies that the concatenation is correct, i.e., that each pair of adjacent counters are equal, and that the reached states match. The latter is encoded in the states, and the former is tested by decreasing both counters together, and checking reachability of $\mathbf{0}$.

In order to generalize the example above, it is crucial that the number of sweeps be *uniformly bounded*. To this end, we observe that if there are more than $|Q|$ forward-backward sweeps along a run, then a state is visited twice at $\vdash$, i.e., forms a loop. If this loop decreases the counter, then there is a shorter run with higher counters, so we need not simulate this former run. If this loop increases the counter, then by repeating it we can obtain an arbitrarily high counter. We can then discard the counter and look for a short accepting run, based on the 2-NFA obtained from $\mathcal{A}$, similarly to Section 3.2. However, unlike Section 3.2, we cannot simply simulate the 2-NFA, since we are committed to a single sweep on the guessed word. In order to overcome this, we *initially* guess which state q will repeat in a positive loop, and then simulate a DFA that is equivalent [31, 30, 32] to the 2-NFA above, starting from state q. At the end of the run, we verify using the reachability query that the loop is indeed positive. Combining these ideas yields a VASS with the conditions above. ◀

▶ **Theorem 7.** *The emptiness problem for sweeping 2-OCN is Ackermann-hard.*

Proof. We show a reduction from VASS reachability, known to be Ackermann-hard [23, 9], to nonemptiness of sweeping 2-OCN. Specifically, we assume that the reachability query is from $(s_1, \mathbf{0})$ to $(s_2, \mathbf{0})$, as this preserves the hardness.

Consider a VASS $\mathcal{V} = \langle S, T\rangle$ of dimension d and configurations $(s_1, \mathbf{0})$ to $(s_2, \mathbf{0})$. We construct a sweeping 2-OCN $\mathcal{A} = \langle Q, T, \Delta, q_{\text{init}}, q_{\text{acc}}\rangle$ where the alphabet is T, i.e., the transitions of $\mathcal{V}$. $\mathcal{A}$ works as follows: given a word $w \in T^*$, first $\mathcal{A}$ sweeps forward and checks that the states given by the transitions follow a run from s_1 to s_2 in $\mathcal{V}$ (ignoring the counter values). If this fails, $\mathcal{A}$ rejects.

Next, $\mathcal{A}$ checks that the counter updates are valid and take the configuration from $\mathbf{0}$ to $\mathbf{0}$. This is done in two phases. In the first phase, $\mathcal{A}$ makes d forward sweeps, where sweep i simulates the i-th counter of $\mathcal{V}$ in the transitions given by w, and each backward sweep resets the counter to 0 by negating the effect of the transition. Thus, if the d sweeps complete successfully, we are assured that w prescribes a run that reaches from $(s_1, \mathbf{0})$ to $(s_2, \boldsymbol{v})$ for some $\boldsymbol{v} \geq \mathbf{0}$. Then, $\mathcal{A}$ makes a forward sweep while keeping the counter at 0 and starts the second phase, where we perform the same simulation but backwards: we check that the run prescribed by w^R (the reverse of w) leads from $(s_2, \mathbf{0})$ to $(s_1, \boldsymbol{v'})$ for some $\boldsymbol{v'} \geq \mathbf{0}$ (note that here we negate the counter operations).

The main observation required to complete the proof is that a for a run ρ of $\mathcal{V}$ from $(s_1, \mathbf{0})$ to $(s_2, \boldsymbol{v})$ we have that ρ^R (with negated counters) is a valid run from $(s_2, \mathbf{0})$ iff $\boldsymbol{v} = 0$. This follows immediately from the VASS semantics.

We conclude that $L(\mathcal{A}) \neq \emptyset$ iff $(s_2, \mathbf{0})$ is reachable from $(s_1, \mathbf{0})$ in $\mathcal{V}$. ◀

3.4 The Membership Problem for 2-OCN is Polytime

Recall that by [7], if a 2-OCN $\mathcal{A}$ has counter updates in $\{-1, 0, 1\}$, we can compute in polynomial time (in the description of $\mathcal{A}$) a polynomial p such that if a word w is accepted, then it is accepted with a run of length at most $p(|w|)$. Unfortunately, this does not carry through to our setting, where counters are encoded in binary, as the following example shows:

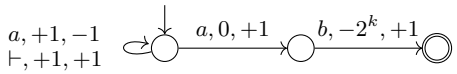


Figure 4 A 2-OCN $\mathcal{A}_k$ parameterized by $k \in \mathbb{N}$. The word $\vdash ab \dashv$ is accepted, but the shortest accepting run is of length $2^k + 1$, since the counter "charges" for 2^k steps reading only the $\vdash a$ prefix.

Nevertheless, we show that 2-OCN retain efficient membership even with binary counters:

▶ **Theorem 8.** *The membership problem for 2-OCN (given a 2-OCN $\mathcal{A}$ and a word w, is $w \in L(\mathcal{A})$?) is decidable in polynomial time.*

Proof. Consider a 2-OCN $\mathcal{A} = \langle Q, \Sigma, \Delta, q_{\text{init}}, q_{\text{acc}} \rangle$ and a word $w \in \Sigma^*$. We implicitly assume that w already includes the end-markers. Similarly to the reasoning in Sections 3.2 and 3.3, we observe that if an accepting run of $\mathcal{A}$ on w visits the same state/position pair (q, p) twice, then if the counter effect between the visits is nonpositive, a shorter accepting run exists, and if the counter effect is positive, then this cycle can be pumped arbitrarily, so that acceptance is dependent only on the 2-NFA obtained by ignoring the counter effects starting from (q, p). Since the number of state/position pairs is $|w| \cdot |Q|$, it follows that within $|w| \cdot |Q| + 1$ steps we visit a cycle. Similarly, in the 2-NFA if an accepting run exists, then there is also one of length at most $|w| \cdot |Q|$.

Deciding whether there exists an accepting run in the 2-NFA from (q, p) is straightforward: we keep track of the reachable set of configurations (q', p') at each step of the run, up to $|w| \cdot |Q|$ steps, and if q_{acc} is reached then the run is accepting.

It remains to detect acceptance in the 2-OCN, or to detect a positive cycle. To do so, we proceed as follows. We simulate $\mathcal{A}$ for $|w| \cdot |Q|$ steps, and for each step $1 \leq i \leq |w| \cdot |Q|$ we store a function $f_i \colon Q \times \{1, \ldots, |w|\} \to \mathbb{N}$ that stores for each configuration (q, p) the maximal counter with which (q, p) can be reached after i steps. It is easy to compute f_i from f_{i-1} in polynomial time. Once this simulation is complete, if q_{acc} does not appear, then we want to look for a positive cycle. To this end, for each $1 \leq j \leq |w| \cdot |Q|$ and configuration (q, p), we again construct functions g_j similarly to the above, with initial configuration $f_i(q, p)$. If at any point we encounter (q, p) with a counter higher than $f_i(q, p)$, then we have a positive cycle. Otherwise, the word is not accepted.

It is easy to verify that all the computations can be carried out in polynomial time. ◀

4 On the Semilinearity of 2-OCN Languages

Preliminaries. Emptiness decidability has been historically intimately tied to *semilinearity*. A set $E \subseteq \mathbb{N}^d$ is semilinear if it is expressible in first-order logic with addition (there are many equivalent definitions and we will not need a specific one). The *Parikh image* of a language L over an alphabet $\{a_1, \ldots, a_k\}$ is the set of vectors $(n_1, \ldots, n_k)$ such that there is a word in L with precisely n_i letters a_i (in any order), for all i. In other words, the Parikh image counts the number $|w|_\ell$ of times each letter ℓ appears in a word $w \in L$, resulting in a set of vectors in $\mathbb{N}^{|\Sigma|}$. A class of languages is *semilinear* if for any language L in that class, the Parikh image of L is semilinear. In that case, if there is an algorithmic way to obtain a representation of that semilinear set from a representation of L, then one can check whether $L = \emptyset$, since satisfiability is decidable for first-order logic with addition [6].

General 2-OCN. Since the emptiness problem is undecidable for 2-DOCN, it is an unsurprising fact that there are nonsemilinear 2-DOCN languages (e.g., $\{(a^n\#)^n \mid n > 1\}$). A tantalizing question surging from the study in Section 3.2 is: *Are all 2-OCN bounded*

languages semilinear? For 2-DOCN, it is proved in [26] that this is indeed the case. For 2-OCN, it is likely that a proof of this statement would provide an alternative proof to Theorem 5. We conjecture that, indeed, 2-OCN are in good company with a wealth of computational models whose bounded languages are precisely those with a semilinear Parikh image. We come short of showing this, but prove, using the construction of Section 3.2 and a result of [15], that:

▶ **Theorem 9.** *For any 2-OCN $\mathcal{A}$ with $L(\mathcal{A}) \subseteq a_1^* a_2^* \cdots a_k^*$ and for any $\Gamma \subseteq \{a_1, a_2, \ldots, a_k\}$, the set $\left\{\sum_{\ell \in \Gamma} |w|_\ell \mid w \in L(\mathcal{A})\right\} \subseteq \mathbb{N}$ is effectively semilinear.*

Proof. In [15, Corollary 4.3] the following is shown.[2] Given a VASS $\mathcal{V} = \langle S, T\rangle$ of dimension d and two states s_1, s_2, consider the set $\mathcal{R}_{s_1,s_2} \subseteq \mathbb{N}^d \times \mathbb{N}^d \times \mathbb{N}^T$ such that $(\boldsymbol{p}, \boldsymbol{p'}, \boldsymbol{t}) \in \mathcal{R}_{s_1,s_2}$ iff $(s_2, \boldsymbol{p'})$ is reachable from $(s_1, \boldsymbol{p})$ via a run ρ that takes transition τ exactly $\boldsymbol{t}(\tau)$ times. Then the image of every morphism $\pi\colon \mathcal{R}_{s_1,s_2} \to \mathbb{N}$ is effectively semilinear.

In our setting, take $\mathcal{V}$ to be the VASS obtained in Section 3.2 with the states $s_{\text{init}}, s_{\text{fin}}$ and $\boldsymbol{p} = \boldsymbol{p'} = \mathbf{0}$. Let π be the morphism that counts the number of transitions used to initialize n_i for all $a_i \in \Gamma$, and recall that by the correctness of the construction of $\mathcal{V}$, we have that along a run from $(s_{\text{init}}, \mathbf{0})$ to $(s_{\text{fin}}, \mathbf{0})$, the value of n_i after the Initialization Component corresponds to the length of the a_i segment in an accepted word. Then, [15, Corollary 4.3] readily shows that the set $\left\{\sum_{a_i \in \Gamma} n_i \mid a_1^{n_1} \cdots a_k^{n_k} \in L(\mathcal{A})\right\}$ is effectively semilinear. ◀

The previous theorem implies that 2-OCN languages over a singleton alphabet are semilinear. In addition, we have the following:

▶ **Corollary 10.** *The languages $\{0^n 1^{n^2} \mid n \geq 1\}$ and $\{0^n 1^{2^n} \mid n \geq 1\}$, known to be 2-DOCA languages [29], are not 2-OCN languages.*

Sweeping 2-OCN. We note that Theorem 9 also applies to sweeping 2-OCN without the bounded language restriction, with a similar proof.

▶ **Theorem 11.** *For any sweeping 2-OCN $\mathcal{A}$ over Σ and $\Gamma \subseteq \Sigma$, $\left\{\sum_{\ell \in \Gamma} |w|_\ell \mid w \in L(\mathcal{A})\right\}$ is effectively semilinear.*

However, unlike the case of bounded languages, here we can show that the Parikh image of sweeping 2-OCN language are not always semilinear.

▶ **Proposition 12.** *There are nonsemilinear languages recognized by sweeping 2-DOCN.*

Proof. Consider the Dyck language over $\{a, b\}$, i.e., well-parenthesized expressions where a corresponds to an opening parenthesis and b to a closing one. Write P for the prefixes of that language. Clearly, P is a 1-DOCN language. Write P' for the language P where a and b swap roles. We can construct a sweeping 2-DOCN that recognizes $K = P \cap a^* P'$ using the closure of 2-DOCN under intersection [26] (which preserves the sweeping property). In any word w in K, if n is the length of the first block of a's, then $0 \leq |w|_a - |w|_b \leq n$. Finally, using a few more sweeps, we can express:

$$L = \{c^n \# w \mid w \in K \wedge w \in a^n (b^+ a^+)^{n-1} b^n\}.$$

Indeed, this requires checking that the number of c's is equal to the number of a's in the first block, the number of b's in the last block, and the number of ab infixes.

Now, for a fixed value n, the longest word in L that can appear after $c^n\#$ is $(a^n b^n)^n$, of length $2n^2$, showing that this language is not semilinear. ◀

[2] In [15] the result is phrased for Petri Nets, but the equivalence between Petri Nets and VASS gives a straightforward translation. In [22], a slightly weaker result is presented in the language of VAS.

We will show, in Section 5, that there is a 1-OCN language (thus a sweeping 2-OCN language) this is not expressible using a 2-DOCN. Conversely, Theorem 12 provides an easy way to show that:

▶ **Corollary 13.** *There is a 2-DOCN language that is not expressible using a sweeping 2-OCN.*

Proof. Consider the language L over the alphabet $\{a, b, c, \#, \underline{b}, x\}$ defined as:

$$L = \{a^n \# b^m x^m \# \underline{b}^m x^{2m} \# \underline{b}^m x^{3m} \# \cdots \# \underline{b}^m c^{nm} \mid n, m > 0\}$$

By applying the morphism $h\colon \Sigma^* \to \Sigma^*$ that erases $\underline{b}, \#$, and x, we have $h(L) = \{a^n b^m c^{nm}\}$, which can be seen as multiplication.

The language L is recognizable by a 2-DOCN $\mathcal{A}$ by repeatedly using a similar approach to Theorem 1:

- Check that the input has the form $a^*\#b^*(x^*\#\underline{b}^*)^*c^*$,
- Check that the number of a's is the same as the number of $\#$'s,
- Check that the first sequence of $\underline{b}$'s is as long as the sequence of b's,
- Check that the lengths of all sequences of $\underline{b}$'s are equal, by checking that each neighboring sequence $\#\underline{b}^i x^* \# \underline{b}^j x^*$ satisfies $i = j$,
- Check for each sequence $x^i \# \underline{b}^m x^j$ that $i + m = j$, and the same for the last segment where x is replaced with c.

This 2-DOCN proceeds segment by segment, always making sure that the number of x's increases by exactly m. This, together with the initial test that the number of segments is n, forces the number of c's to be exactly nm.

However, the lengths of words in this language do not form a semilinear set, so by Theorem 11 it cannot be recognized by a sweeping 2-OCN. ◀

Over bounded languages, the following exact characterization holds:

▶ **Theorem 14.** *A bounded language is recognized by a sweeping 2-OCN iff it is semilinear.*

Proof. The fact that any semilinear bounded language can be expressed with a sweeping 2-OCN is easily deduced from the fact that semilinear sets are those expressible with *quantifier-free* first-order formulas with addition, order, and modulo. Putting such a formula in DNF, we simply need to guess which conjunction is to hold, and the conjunction can be checked by a sweeping 2-DOCN using the closure of 2-DOCN under intersection [26] (which preserves the sweeping property).

We turn to the fact that bounded languages expressed by sweeping 2-OCN are semilinear. We split the proof into two parts.

Constant number of sweeps. Let us first assume that the sweeping 2-OCN $\mathcal{A}$ executes a constant number of sweeps, say C, with C odd, for each input word (we assume here that the machine ends at the right end-marker). With $L(\mathcal{A}) \subseteq a_1^* \cdots a_k^*$, consider the following transformation that produces a word over the alphabet $\Gamma = \{a_{i,j} \mid 1 \leq i \leq C \wedge 1 \leq j \leq k\}$:

$$w = a_1^{n_1} \cdots a_k^{n_k} \quad \rightsquigarrow \quad w' = a_{1,1}^{n_1} \cdots a_{1,k}^{n_k} \# a_{2,k}^{n_k} \cdots a_{2,1}^{n_1} \# \cdots \# a_{C,1}^{n_1} \cdots a_{C,k}^{n_k}.$$

That is, the input is replicated C times, with every other copy reversed, and each copy on its own alphabet. Let us call w' the *replica* of w. Since $\mathcal{A}$ makes at most C sweeps, we can build a 1-OCN $\mathcal{B}$ that reads the replica of any word $w \in a_1^* \cdots a_k^*$ and accepts if and only if w is accepted by the 2-OCN (and the input is of the format $R = a_{1,1}^* \cdots a_{1,k}^* \# a_{2,k}^* \cdots a_{2,1}^* \# \cdots \# a_{C,1}^* \cdots a_{C,k}^*$). Naturally, $\mathcal{B}$ also accepts words that are *not* replicas.

We circumvent this problem by using an "external" semilinear set, as follows. Let E be the Parikh image of the language where for all i, each $a_{\bullet,i}$ appears the same number of times; clearly, a word in R is a replica iff its Parikh image is in E. Since $\mathcal{B}$ is a 1-OCN language (and in particular context-free), the Parikh image F of its language is semilinear by Parikh's Theorem [28]. It is then easy to check that $E \cap F$ is exactly the set of Parikh images of replicas of words in $L(\mathcal{A})$, showing that the Parikh image of $L(\mathcal{A})$ itself is semilinear (as a projection of the former).

Any number of sweeps. If $\mathcal{A}$ is sweeping but does not do a constant number of sweeps for each input, we adapt the recurring idea of Sections 3.2 and 3.3. Let k be the number of states of $\mathcal{A}$. After $2k+1$ sweeps, a state is repeated at the left end-marker; if the counter effect between the two repetitions is nonpositive, then that run is not useful for acceptance, and can be disregarded. If the counter *increases*, then it can be increased arbitrarily, and the behavior of the 2-OCN is the same as a 2-NFA $\mathcal{N}$ from that point on. Consider the 1-OCN $\mathcal{B}$ constructed above, with $C = 2k+1$. We can additionally have $\mathcal{B}$ guess the state that repeats, if any, and check that it indeed repeats. Next, $\mathcal{B}$ assumes that the counter strictly increases in the repetition, and branches into a DFA for $\mathcal{N}$. In order to "kill" the runs of $\mathcal{B}$ where this assumption is incorrect, any transition that $\mathcal{B}$ takes, in between the first and second appearance of the repeated state, should be followed by input symbols indicating the counter effect, increasing ($+^*$) or decreasing ($-^*$), and by how much. $\mathcal{B}$ then verifies that the correct effect is matched to the transition of $\mathcal{A}$ it guesses. For instance, $\mathcal{B}$ may read the following replica of *aabbbc* with extra counter information (here the original alphabet is $\{a,b,c\}$ and the replica alphabet has $1,\dots,5$ as indices):

$$\underbrace{a_1a_1b_1b_1b_1c_1\#c_2b_2b_2b_2a_2a_2}_{\mathcal{B}\text{ simulates two sweeps of }\mathcal{A}}\#\underbrace{a_3{+}{+}a_3{+}b_3{-}b_3{+}b_3{-}c_3{+}\#c_4{-}b_4{+}b_4{+}b_4{-}a_4{-}a_4{-}}_{\text{counter updates appear in input showing action of }\mathcal{A}\text{'s transitions}}\#\underbrace{a_5a_5b_5b_5b_5c_5\#}_{\text{A DFA for }\mathcal{N}\text{ is ran}}$$

$\mathcal{B}$ guesses state repetition

$\mathcal{B}$ checks state repetition and assumes that counter increased

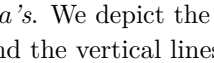
As mentioned, this modified version of $\mathcal{B}$ may jump into a DFA for $\mathcal{N}$ when it should not. However, we can augment the external semilienar set E, checking that the input word is a replica, to also check that the number of + is strictly greater than the number of -. This corresponds to the counter updates between the state repetitions having a strict positive impact, and that $\mathcal{B}$ correctly jumps into $\mathcal{N}$'s simulation. To conclude, let again F be the (semilinear) Parikh image of the language of $\mathcal{B}$. The vectors in $E \cap F$ correspond precisely to the replicas (with possibly explicit counter updates) that are accepted by $\mathcal{A}$, hence the language of $\mathcal{A}$ is semilinear. ◀

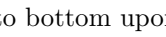
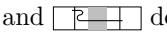
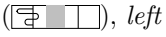
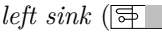
5 2-DOCN vs 1-OCN: A Pumping Lemma for 2-DOCN

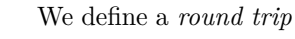
In [7] it is shown that the language $\{a^n b^m \mid n \neq m\}$ is 2-OCN recognizable but not 2-DOCN recognizable. We strengthen this result in two directions: first we present a general pumping lemma that we can then use to establish that various languages are not 2-DOCN-recognizable. Second, using our lemma we are able to find a language that is not 2-DOCN recognizable, but is already 1-OCN recognizable. This shows that nondeterminism can be used to compensate for two-wayness, in some settings.

▶ **Lemma 15.** *Let Σ be an alphabet and $a \in \Sigma$. For every 2-DOCN-recognizable language $L \subseteq \Sigma^*$ there exists $K \in \mathbb{N}$ such that for every $x, y \in \Sigma^*$, if $xa^K y \notin L$ then there exists $K' \neq K$ such that $xa^{K'} y \notin L$ as well.*

We give a rough sketch of the main ideas of the proof, see the full version for the complete arguments and definitions. Consider a 2-DOCN $\mathcal{A} = \langle Q, \Sigma, \Delta, q_{\text{init}}, q_{\text{acc}} \rangle$, and think of K as some large constant. Our goal is to devise a way to pump certain infixes of a word $xa^K y$, so that we can reason about the resulting run of $\mathcal{A}$ on it. Naturally, the challenge lies in synchronizing the behaviors of forward and backward head movements in the run.

We think of the a^K infix as partitioned to three parts: two short $a^{|Q|}$ segments referred to as *outer left a's* and *outer right a's*, and a long $a^{K-2|Q|}$ segment in the middle, called the *inner a's*. We depict the word $xa^K y$ as  , with the grayed area representing the inner a's, and the vertical lines marking the end of x and the beginning of y.

We then consider the form of the run of $\mathcal{A}$ on $xa^K y$. Specifically, we divide runs according to whether they cross into the inner a's. We show that runs that do reach the inner a's, must continue to the end of the tape (either from right or left). Runs that do not cross the inner a's must therefore get stuck in a small part of the tape. By depicting runs as going from top to bottom upon head reversal, we have e.g., the type  which depicts runs going left, and  depicts runs going right. The possible types of runs are then *left loop* (), *left sink* (), *left crossing* () and their right counterparts. Each type signifies a specific behavior of the runs, e.g., whether they repeat a state, or stop due to the counter becoming negative.

We define a *round trip* to be a concatenation of runs of the form  , and show that either (1) $\mathcal{A}$ has a run on $xa^K y$ with many round-trips, or (2) it has a run with a few round trips that gets stuck in a loop, or (3) its maximal run has few round trips and ends with the counter becoming negative. We then provide pumping arguments according to each type of run.

A crux of the proof lies in handling Form (3) above. The main idea is the following: consider a run with several round-trips that ends due to a counter becoming negative. Since the inner a's are a long infix, we can find two indices that are visited with the same sequence of states in the round trips. We can then attempt to "pump" or "cut" the infix between these indices, but this may cause the counter to ultimately increase, thus yielding an accepting run from a previously-rejecting one. We therefore carefully analyze the effect of the counter between these two indices on each pass of the form or . If at any point the cumulative sum of effects is negative, we can pump the run enough so that the counter ultimately becomes negative, and the word is not accepted. Otherwise, all the cumulative sums are non-negative, in which case we can cut the cycle, causing the counter to either become negative or to complete the run in the same state as the original run, which is not accepting. The precise details are involved and appear in the full version.

▶ **Corollary 16.** *$L = \{a^\ell b^m c^n \mid \ell > m \vee m > n\}$ is 1-OCN but not 2-DOCN recognizable.*

Proof. L is 1-OCN recognizable by guessing whether $\ell > m$ or $m > n$, and checking by either increasing the counter on a and decreasing on b, or increasing on b and decreasing on c (with another decrease at the end to make the inequality strict). Assume by way of contradiction that $L = L(\mathcal{D}_2)$ for a 2-DOCN $\mathcal{D}_2$. Let K be the constant provided by Theorem 15, then $a^K b^K c^K \notin L = L(\mathcal{D}_2)$. Then, Theorem 15 guarantees that there exists $K' \neq K$ such that $a^K b^{K'} c^K \notin L(\mathcal{D}_2)$, but $a^K b^{K'} c^K \in L$, since either $K > K'$ or $K' > K$, a contradiction. ◀

We remark that we can similarly show that $L_2 = \{a^n b^m \mid n \neq m\}$ is also not 2-DOCN recognizable (but is recognizable by a sweeping 2-OCN), giving an alternative proof to the result in [26].

6 Research Directions

Separations. The classes we consider, 1-OCN, 1-DOCN, 2-OCN, 2-DOCN, sweeping and nonsweeping, over bounded languages or not, are all provably distinct, except for the following: are all bounded 2-OCN languages expressible with a sweeping machine? We conjecture that they are, and leave this question open (this is known for 2-DOCN from [7]).

Closure. The examples of languages outside 2-DOCN readily show that the class of languages recognized by 2-DOCN is not closed under union nor complement, but is closed under intersection (also shown in [26]). For 2-OCN, the class is closed under union, but we conjecture that it is closed under neither intersection nor complement. We leave these questions open, together with the same questions for sweeping 2-OCN.

Decidability. Beyond emptiness, our proofs imply that it is decidable whether the intersection of two (sweeping or bounded) 2-OCN is empty. The next natural question is then the decidability of *inclusion*. We conjecture that this problem is decidable for sweeping and bounded 2-OCN.

References

1 Shaull Almagor, Guy Avni, Henry Sinclair-Banks, and Asaf Yeshurun. Dimension-minimality and primality of counter nets. In *International Conference on Foundations of Software Science and Computation Structures*, volume 14575, pages 229–249. Springer, Springer, 2024. doi:10.1007/978-3-031-57231-9_11.

2 Shaull Almagor, Udi Boker, Piotr Hofman, and Patrick Totzke. Parametrized universality problems for one-counter nets. In *31st International Conference on Concurrency Theory (CONCUR 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CONCUR.2020.47.

3 Shaull Almagor and Asaf Yeshurun. Determinization of one-counter nets. In *33rd International Conference on Concurrency Theory (CONCUR 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CONCUR.2022.18.

4 Setsuo Arikawa. On some properties of length-growing functions on two-way pushdown automata. *Memoirs of the Faculty of Science, Kyushu University. Series A, Mathematics*, 22(2):110–127, 1968.

5 Mohamed Faouzi Atig and Pierre Ganty. Approximating petri net reachability along context-free traces. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 152–163. Dagstuhl, Germany: Leibniz-Zentrum für Informatik, 2011. doi:10.4230/LIPIcs.FSTTCS.2011.152.

6 Leonard Berman. The complexity of logical theories. *Theoretical Computer Science*, 11(1):71–77, 1980. doi:10.1016/0304-3975(80)90037-7.

7 Tat-hung Chan. On two-way weak counter machines. *Mathematical systems theory*, 20(1):31–41, 1987. doi:10.1007/BF01692057.

8 Wojciech Czerwiński, Sławomir Lasota, Ranko Lazić, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for petri nets is not elementary. *Journal of the ACM (JACM)*, 68(1):1–28, 2020. doi:10.1145/3422822.

9 Wojciech Czerwiński and Łukasz Orlikowski. Reachability in vector addition systems is ackermann-complete. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1229–1240. IEEE, 2022. doi:10.1109/FOCS52979.2021.00120.

10 Zhe Dang, Oscar H Ibarra, and Zhi-Wei Sun. On two-way nondeterministic finite automata with one reversal-bounded counter. *Theoretical computer science*, 330(1):59–79, 2005. doi:10.1016/j.tcs.2004.09.010.

11 Marzio De Biasi and Abuzer Yakaryılmaz. Unary languages recognized by two-way one-counter automata. In *International Conference on Implementation and Application of Automata*, pages 148–161. Springer, 2014. doi:10.1007/978-3-319-08846-4_11.

12 Pavol Duris and Zvi Galil. Fooling a two way automation or one pushdown store is better than one counter for two way machines. *Theoretical Computer Science*, 21(1):39–53, 1982. doi:10.1016/0304-3975(82)90087-1.

13 Zvi Galil. Some open problems in the theory of computation as questions about two-way deterministic pushdown automaton languages. *Mathematical systems theory*, 10(1):211–228, 1976. doi:10.1007/BF01683273.

14 James N Gray, Michael A Harrison, and Oscar H Ibarra. Two-way pushdown automata. *Information and Control*, 11(1-2):30–70, 1967. doi:10.1016/S0019-9958(67)90369-5.

15 Dirk Hauschildt and Matthias Jantzen. Petri net algorithms in the theory of matrix grammars. *Acta Informatica*, 31:719–728, 1994. doi:10.1007/BF01178731.

16 Piotr Hofman, Richard Mayr, and Patrick Totzke. Decidability of weak simulation on one-counter nets. In *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 203–212. IEEE, 2013. doi:10.1109/LICS.2013.26.

17 Piotr Hofman and Patrick Totzke. Trace inclusion for one-counter nets revisited. In *Reachability Problems: 8th International Workshop, RP 2014, Oxford, UK, September 22-24, 2014. Proceedings 8*, pages 151–162. Springer, 2014. doi:10.1007/978-3-319-11439-2_12.

18 Oscar H Ibarra and Zhe Dang. On two-way fa with monotonic counters and quadratic diophantine equations. *Theoretical computer science*, 312(2-3):359–378, 2004. doi:10.1016/j.tcs.2003.10.027.

19 Oscar H Ibarra, Tao Jiang, Nicholas Tran, and Hui Wang. On the equivalence of two-way pushdown automata and counter machines over bounded languages. In *STACS 93: 10th Annual Symposium on Theoretical Ascpects of Computer Science Würzburg, Germany, February 25–27, 1993 Proceedings 10*, pages 354–364. Springer, 1993. doi:10.1007/3-540-56503-5_36.

20 Oscar H Ibarra, Tao Jiang, Nicholas Tran, and Hui Wang. New decidability results concerning two-way counter machines. *SIAM Journal on Computing*, 24(1):123–137, 1995. doi:10.1137/S0097539792240625.

21 Oscar H Ibarra and Jianwen Su. Counter machines: decision problems and applications. In *Jewels are Forever: Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 84–96. Springer, 1999.

22 Hans Kleine Büning, Theodor Lettmann, and Ernst W. Mayr. Projections of vector addition system reachability sets are semilinear. *Theoretical Computer Science*, 64(3):343–350, 1989. doi:10.1016/0304-3975(89)90055-8.

23 Jérôme Leroux. The reachability problem for petri nets is not primitive recursive. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1241–1252. IEEE, 2022. doi:10.1109/FOCS52979.2021.00121.

24 Daniel Martin and John Gwynn. Two results concerning the power of two-way deterministic pushdown automata. In *Proceedings of the ACM annual conference*, pages 342–344, 1973. doi:10.1145/800192.805729.

25 Marvin Lee Minsky. *Computation*. Prentice-Hall Englewood Cliffs, 1967.

26 Satoru Miyano. Two-way deterministic multi-weak-counter machines. *Theoretical Computer Science*, 21(1):27–37, 1982. doi:10.1016/0304-3975(82)90086-X.

27 Burkhard Monien. Deterministic two-way one-head pushdown automata are very powerful. *Information processing letters*, 18(5):239–242, 1984. doi:10.1016/0020-0190(84)90001-2.

28 Rohit J. Parikh. On context-free languages. *J. ACM*, 13(4):570–581, 1966. doi:10.1145/321356.321364.

29 Holger Petersen. Two-way one-counter automata accepting bounded languages. *ACM SIGACT News*, 25(3):102–105, 1994. doi:10.1145/193820.193835.

30 Michael O Rabin and Dana Scott. Finite automata and their decision problems. *IBM journal of research and development*, 3(2):114–125, 1959. doi:10.1147/rd.32.0114.

31 John C Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3(2):198–200, 1959. doi:10.1147/rd.32.0198.
32 Moshe Y. Vardi. A note on the reduction of two-way automata to one-way automata. *Information Processing Letters*, 30(5):261–264, 1989. doi:10.1016/0020-0190(89)90205-6.

A Detailed Construction of $\mathcal{V}$ in Section 3.2

We follow the intuition given in Section 3.2. Let $\mathcal{A} = \langle Q, \Sigma, \Delta, q_{\text{init}}, q_{\text{acc}} \rangle$. Recall that $\Sigma = \{a_1, \ldots, a_k\}$. When constructing $\mathcal{V}$ we refer to vector entries as *counters*, and we specify vector operations on individual counters, e.g., if c is a specific counter, then $c \leftarrow +1$ refers to the vector that is all 0 apart from $+1$ in the entry corresponding to c. Similarly, we can combine several counter operations in a single vector (e.g., $c_1 \leftarrow +1, c_2 \leftarrow -1$).

We start by describing the counters of $\mathcal{V}$ and their intuitive roles.

- `c` – tracking the counter of $\mathcal{A}$
- `c_freeze_1` – a copy of `c`, to be frozen at the beginning of a positive loop.
- `c_freeze_2` – a copy of `c`, to be frozen at the end of a positive loop.
- `L[1]` ... `L[k]` and `R[1]` ... `R[k]` – a pair of counters marking the position of the head from the Left and from the Right of each segment.

We now describe the states and transitions of $\mathcal{V}$ by listing the information stored in each state, and its intuitive meaning. We divide the states according to the components of $\mathcal{V}$.

Initialization Component. This component consists of a single state s_{init}. The transitions include a k self loops for $i \in \{1, \ldots, k\}$, each with operation $R[i] \leftarrow +1$. These transitions allow $\mathcal{V}$ to "charge" the number of letters in each a_i segment. Another transition (with operation $\mathbf{0}$) leads to the simulation component with current state q_{init} and head position on $\vdash$ (i.e., current segment 0, see below).

Simulation Component. A state contains the following information:

- `cur_state` $\in Q$ – the current state of $\mathcal{A}$.
- `cur_seg` $\in \{0, \ldots, k+1\}$ – the current segment, with 0 and $k+1$ representing $\vdash$ and $\dashv$, respectively.
- `num_moves`: $\{(i, i+1), (i+1, i) \mid i \in \{0, \ldots, k\}\} \to \{0, \ldots, |Q|+1\}$ – how many times did we move from segment i to segment $i+1$ (bounded by $|Q|+1$).
- `loop_seg_move` $\in \{(i, i+1), (i+1, i) \mid i \in \{0, \ldots, k\}\} \cup \{\bot\}$ – the border between segments where we guess a loop occurs, or $\bot$ if this is not guessed yet.
- `loop_state` $\in Q \cup \{\bot\}$ – the state where we guess a loop occurs, or $\bot$ if this is not guessed yet.

The transitions within segments are induced by those of $\mathcal{A}$: if `cur_state` $= q$ and `cur_seg` $= i$, then each transition $(q, a_i, e, h, q') \in \Delta$ induces a transition to a state with `cur_state` $= q'$ and the counter operations `c` $\leftarrow +e$, `c_freeze_2` $\leftarrow +e$, `L[i]` $\leftarrow +h$, `R[i]` $\leftarrow -h$. Additionally, if `loop_seg_move` $=$ `loop_state` $= \bot$ then `c_freeze_1` behaves like `c`.

The transition between segments are similarly induced, but instead of updating `L[i]` and `R[i]`, if $h = 1$ then $\mathcal{V}$ guesses that we move to a state with *cur_seg* $= i+1$. Then, `R[i]` is 0-tested and `num_moves`$(i, i+1)$ is increased by 1, if possible (otherwise there is no transition). The left moves ($h = -1$ are dual).

In addition, upon moving between segments, $\mathcal{V}$ may nondeterministically record the current state and segments in `loop_statea` and `loop_seg_move` respectively. Note that from then on, `c_freeze_1` no longer changes. If those are already recorded then it may check whether the current state and segments match the recorded ones, in which case we move to the 2-NFA Component (which also freezes `c_freeze_2`).

If at any point we have $\texttt{cur\_state} = q_{\text{acc}}$, we move to the Finalizing Component.

2-NFA Component. A state contains the following information:

- $\texttt{cur\_state} \in Q$ – the current state of $\mathcal{A}$.
- $\texttt{cur\_seg} \in \{0, \ldots, k+1\}$ – the current segment, with 0 and $k+1$ representing $\vdash$ and $\dashv$, respectively.
- $\texttt{num\_moves}\colon \{(i, i+1), (i+1, i) \mid i \in \{0, \ldots, k\}\} \to \{0, \ldots, |Q|+1\}$ – how many times did we move from segment i to segment $i+1$ (bounded by $|Q|+1$).
- $\texttt{loop\_seg\_move} \in \{(i, i+1), (i+1, i) \mid i \in \{0, \ldots, k\}\} \cup \{\bot\}$ – the border between segments where we guess a loop occurs, or $\bot$ if this is not guessed yet.
- $\texttt{loop\_state} \in Q \cup \{\bot\}$ – the state where we guess a loop occurs, or $\bot$ if this is not guessed yet.

The transitions are similar to those of the Simulation Component, with the exception that $\texttt{c}$, $\texttt{c\_freeze\_1}$, $\texttt{c\_freeze\_2}$ are no longer changed (i.e., the counter of $\mathcal{A}$ is no longer tracked). Note that upon reaching the 2-NFA Component, the function $\texttt{num\_moves}$ is reset to 0 to begin a fresh count.

Finalizing Component. This component consists of two states: $s_{\text{check\_loop}}$ and s_{fin}. $s_{\text{check\_loop}}$ has a self loop with operation $\texttt{c\_freeze\_1} \leftarrow -1, \texttt{c\_freeze\_2} \leftarrow -1$, and a transition to s_{fin} with operation $\texttt{c\_freeze\_2} \leftarrow -1$. Lastly, s_{fin} has a self loop allowing to decrease each counter separately, except for $\texttt{c\_freeze\_1}$.

We prove the correctness of the construction in Appendix A.

▶ **Lemma 17.** *$L(\mathcal{A}) \neq \emptyset$ iff $(s_{fin}, \mathbf{0})$ is reachable from $(s_{init}, \mathbf{0})$ in $\mathcal{V}$.*

Proof. It is not hard to see that by construction, the states of the Simulation Component of $\mathcal{V}$ correctly track the word guessed by $\mathcal{V}$ in the Initialization Component provided no more than $|Q|$ segment moves occur. Thus, in order to prove correctness, it is enough to prove that if $\mathcal{A}$ accepts some word w, then it also accepts w with at most $|Q|$ moves between segments i and $i+1$ for all i. Unfortunately, this is not always the case. However, note that if more than $|Q|$ moves between segments i and $i+1$ occur, then there must be a state q that is repeated in two of those moves. If the counter effect of this loop is nonpositive, then the loop can be cut to obtain a "better" run (i.e., one whose counter values are not lower). Otherwise, if the counter effect of the loop is positive, then it can be repeated to make the counter of $\mathcal{A}$ arbitrarily high. Then, $\mathcal{A}$ accepts the word iff the 2-NFA obtained from $\mathcal{A}$ by discarding the counter effects is accepted from the loop state q and the position of the head at the end of segment i. Moreover, such an accepting run can be assumed not to loop, since otherwise a shorter accepting run can be found. Thus, In the latter case, the 2-NFA component tracks an accepting run, if one exists.

Finally, note that the finalization component can reach s_{fin} with $\texttt{c\_freeze\_1} = 0$ only if $\texttt{c\_freeze\_1} < \texttt{c\_freeze\_2}$, i.e., the loop taken was indeed strictly positive. ◀

B Proof of Theorem 6

We present the detailed construction of $\mathcal{V}$ from $\mathcal{A}$, starting with the counters of $\mathcal{V}$ and their intuitive roles.

- $\texttt{f[1]}, \ldots, \texttt{f[|Q|+1]}$ – the counter value before each forward sweep.
- $\texttt{b[1]}, \ldots, \texttt{b[|Q|+1]}$ – the counter value before each backward sweep.
- $\texttt{f\_loop\_1}$ – the counter before the first forward sweep that starts with a loop state q.
- $\texttt{f\_loop\_2}$ – the counter before the second forward sweep that starts with a loop state q.

We now turn to describe the states and transitions of $\mathcal{V}$, split into three components.

Initialization Component. This component starts in state s_{init} and has other states containing the following information:

- `qf[1]`, ..., `qf[|Q|+1]` $\in Q \cup \{\bot\}$ – the state *beginning* each forward sweep, or $\bot$ if the sweep is not taken.
- `qb[1]`, ..., `qb[|Q|]` $\in Q \cup \{\bot\}$ – the state *ending* each backward sweep, or $\bot$ if the sweep is not taken.
- `q_loop` – a state that appears twice in the forward sweeps.

The transitions from s_{init} guess the components in the next state, with the following restrictions:

- if $\bot$ appears at some entry, then all later entries are also $\bot$ (i.e., once a certain sweep is not taken, not further sweeps are taken).
- the first forward state is forced to be q_{init} and if `q_loop` $= \bot$ then the last forward state that is not $\bot$ is q_{acc}.
- `qb[i]` $=$ `qf[i+1]` for all $i \leq |Q|$, i.e., the backward sweep ends with the same state the next forward sweep starts with.
- The field `q_loop` is guessed in a consistent way, i.e., it must be the first state that appears twice in the `qf[i]` states.

Each initialization state has self loops to charge the counters with the following operations: for every $i \in 1, \ldots, |Q|$, we have a loop with operations `b[i]` $\leftarrow +1$, `f[i+1]` $\leftarrow +1$, corresponding to the fact that the counter with which the i-th backward sweep ends is the same as the counter with which the $i+1$-th forward sweep starts. In addition, if `q_loop` $=$ `qf[i]` $=$ `qf[j]` for some $i < j$, then `f_loop_1` $\leftarrow +1$ is applied together with `f[i]` $\leftarrow +1$ and `f_loop_2` $\leftarrow +1$ together with `f[j]` $\leftarrow +1$. Thus enforcing the semantics mentioned above.

The initialization states can nondeterministically move to the Simulation Component.

Simulation Component. The initialization component keeps track of the forward and backward runs of $\mathcal{A}$, as well as acceptance in the DFA equivalent to $\mathcal{A}$ from the loop state `q_loop` as follows. Each state in this component carries the same information as the initialization component, but updates it using the transition function.

First, we obtain $\mathcal{A}^R = \langle Q, \Sigma, \Delta^R, q_{\text{acc}}, q_{\text{init}} \rangle$ by defining $\Delta^R = \{(q', \sigma, -e, -h, q) \mid (q, \sigma, e, h, q') \in \Delta,\ \sigma \in \Sigma\}$ (observe that we omit transitions on $\vdash, \dashv$, this is explained below). Thus, $\mathcal{A}^R$ essentially reverses the runs of $\mathcal{A}$. Furthermore, for each state $q \in Q$ we obtain from $\mathcal{A}$ a DFA $\mathcal{D}_q$ by discarding the counter effects of $\mathcal{A}$, setting the initial state to q and converting the resulting 2-NFA to a DFA [31, 30, 32] of single-exponential size.

The main behavior in this component is as follows. From state s of $\mathcal{V}$, we guess a letter $\sigma \in \Sigma_{\vdash,\dashv}$ (the guesses of $\vdash$ and $\dashv$ have special status, see below). The transition then guesses, for each state `qf[i]` in s, a transition on σ, and updates the corresponding `f[i]` counter with the transition effect. Dually, for each $[qb[i]]$ we guess a transition on σ in $\mathcal{A}^R$ and update `b[i]` accordingly. Finally, we update *q_loop* using the DFA $\mathcal{D}_q$ with the transition on σ.

To treat the end-markers, we note that reading an end-marker is joint between the forward and backward runs, and we therefore only want to simulate each end-marker in one of them, and we choose the forward runs (which is why we omit these transitions from $\mathcal{A}^R$). We force the first letter guessed to be $\vdash$, and we only update the forward elements (the `qf[i]` states and `f[i]` counters). Upon guessing $\dashv$, we again only update the forward elements, and we transition to the Finalizing Component.

Finalizing Component. In order to verify that the sweeps are concatenated correctly, we proceed as follows. In state $s_{\text{check\_sweeps}}$ we first check that $\texttt{qf[i]} = \texttt{qb[i]}$, i.e., that the forward and backward sweeps meet at the same state. We then have loops with operation $\texttt{f[i]} \leftarrow -1, \texttt{b[i]} \leftarrow -1$ for all $1 \leq i \leq |Q|$. Note that such transitions can be taken to reach $\mathbf{0}$ iff the counters are the same, implying that the forward and backward sweeps meet with the same counter value.

The above checks that the forward and backward sweeps form a valid run of $\mathcal{A}$ on the guessed word. If the last forward state is q_{acc}, this ensures the word is accepted. It remains to check acceptance in the case of a loop. To this end, if $\texttt{q\_loop} \neq \bot$, we check that the state reached in $\texttt{q\_loop}$ is an accepting state of $\mathcal{D}_q$. However, we also need to check that the loop taken is indeed positive. To achieve this we move to a state with operation $\texttt{f\_loop\_1} \leftarrow -1, \texttt{f\_loop\_2} \leftarrow -1$, and then a final transition with only the operation $\texttt{f\_loop\_2} \leftarrow -1$ to s_{fin}. Then, reaching $\mathbf{0}$ ensures that the loop was indeed positive, as the counter increased between the first and second visits to the loop states.

The correctness of the construction follows from the observation that if $w \in L(\mathcal{A})$, then either w accepted within $|Q|$ forward sweeps, or a positive loop is reached, and from this loop there is a run to an accepting state. In this case, the loop can be arbitrarily pumped so that the run to the accepting state can be taken in the corresponding DFA.

Finally, since the size of $\mathcal{D}_q$ is single-exponential in the size of $\mathcal{A}$ [31, 30, 32], the construction is single-exponential. ◀

Boundedness of Cost Register Automata over the Integer Min-Plus Semiring

Andrei Draghici
Department of Computer Science, University of Oxford, UK

Radosław Piórkowski
Department of Computer Science, University of Oxford, UK

Andrew Ryzhikov
Department of Computer Science, University of Oxford, UK

Abstract

Cost register automata (CRAs) are deterministic automata with registers taking values from a fixed semiring. A CRA computes a function from words to values from this semiring. CRAs are tightly related to well-studied weighted automata. Given a CRA, the boundedness problem asks if there exists a natural number N such that for every word, the value of the CRA on this word does not exceed N. This problem is known to be undecidable for the class of linear CRAs over the integer min-plus semiring $(\mathbb{Z} \cup \{+\infty\}, \min, +)$, but very little is known about its subclasses. In this paper, we study boundedness of copyless linear CRAs with resets over the integer min-plus semiring. We show that it is decidable for such CRAs with at most two registers. More specifically, we show that it is, respectively, NL-complete and in coNP if the numbers in the input are presented in unary and binary. We also provide complexity results for two classes with an arbitrary number of registers. Namely, we show that for CRAs that use the minimum operation only in the output function, boundedness is PSPACE-complete if transferring values to other registers is allowed, and is coNP-complete otherwise. Finally, for each f_i in the hierarchy of fast-growing functions, we provide a stateless CRA with i registers whose output exceeds N only on runs longer than $f_i(N)$. Our construction yields a non-elementary lower bound already for four registers.

2012 ACM Subject Classification Theory of computation → Quantitative automata

Keywords and phrases cost register automata, boundedness, decidability

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.20

Funding All authors are supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 852769, ARiAT).

Acknowledgements We thank the anonymous reviewers for their helpful comments.

1 Introduction

A cost register automaton (CRA), introduced by Alur et al. [3], is a deterministic finite automaton over finite words equipped with a finite set of registers storing values from a fixed semiring. When a CRA reads a word, the values of its registers are updated in a write-only way using the semiring operations, and at the end it outputs a value from the semiring. Thus, a CRA defines a function from finite words to elements of a semiring. CRAs can hence be used to model quantitative behaviour of systems, and are tightly related to well-studied weighted automata (WAs) introduced by Schützenberger in [36], see also surveys [18, 19]. In this context, a fundamental question is, given a CRA or WA, to decide if a certain property of the function computed by it (or even just a property of the image of this function) holds.

In this paper, we study the boundedness problem, which, given a CRA, asks if there exists a natural number N such that the output of the CRA on every input is smaller than N. As discussed in more detail below, very little is known about decidability of this problem

compared to other natural problems for WAs and CRAs. Our work thus addresses and partially closes this gap, by studying it for restricted classes of CRAs. Below we provide an overview of such classes, and put them in the context of known classes of WAs.

We concentrate on CRAs over the integer min-plus semiring $(\mathbb{Z} \cup \{+\infty\}, \min, +)$. All CRAs are thus assumed to be over this semiring unless stated otherwise. Such CRAs can be seen as a variant of counter automata, specifically, as an extension of integer vector addition systems with states [21, 8]. Reachability properties of the latter are characterised by Presburger arithmetic, the first order theory of integer numbers with addition and order [21], which has good algorithmic features. For CRAs over the integer min-plus semiring, the situation changes significantly due to the presence of minimum operations in the updates. This makes the computed functions highly nonlinear: for example, they can compute iterated minimums. The results for integer vector addition systems thus cannot be directly used for CRAs. On the positive side, for subclasses of CRAs with decidable properties, this opens a possibility of finding new decidable extensions of Presburger arithmetic, by finding logical characterisation of the functions computed by CRAs from such subclasses.

More generally, many fundamental properties of CRAs and WAs can be defined by simple formulas in first-order logic, and can thus be considered as model checking CRAs against a fixed formula. For example, boundedness can be expressed by the formula

$$\exists N \in \mathbb{Z} \cup \{+\infty\}.\ N < +\infty \wedge \forall v \in \mathbb{Z} \cup \{+\infty\}.\ \mathcal{I}(v) \to (v < N),$$

where $\mathcal{I}(v)$ is the predicate that holds true for $v \in \mathbb{Z} \cup \{+\infty\}$ if and only if the CRA outputs the value v on some word. Understanding the decidability landscape of natural properties such as boundedness can thus be seen as a first step towards much more general model checking algorithms for subclasses of CRAs and WAs.

Relations between CRAs and WAs

In general, CRAs are strictly more expressive than WAs [3]. However, WAs are equally expressive to linear CRAs, which are CRAs where the updates of the registers are restricted to affine transformations. Transforming a linear CRA into an equivalent WA and vice versa can be done in polynomial time [3]. Hence, linear CRAs can be seen as a deterministic model for inherently nondeterministic WAs. WAs, and thus CRAs, find their applications in the areas of language and speech processing [34], verification [9], image processing [10], and the analysis of on-line algorithms [5] and probabilistic systems [39]. Functions computable by WAs are exactly those that can be defined in weighted monadic second order logic, a natural extension of monadic second order logic [16, 17]. Functions computed by a subclass of CRAs were also characterised in [31] by maximal partition logic, a logic with regular quantifiers that allow to partition words into segments and then aggregate the values computed for them.

Ambiguity hierarchy of WAs

To make decision problems tractable for WAs, it is usually required to restrict their expressiveness. The most well-studied way of doing that is by bounding the ambiguity, that is, the number of accepting runs labelled by a word. A WA is called finitely (respectively, linearly, polynomially or exponentially) ambiguous if there exists a constant (respectively, a linear, polynomial or exponential) function $f(n)$ such that for every word w the number of accepting runs labelled by w is bounded by $f(|w|)$. If $f(n) = 1$, a WA is called unambiguous. Most classical decision problems, such as universality, inclusion and equivalence, are undecidable already for linearly ambiguous WAs over the integer min-plus semiring [28, 1, 12]. For finitely ambiguous WAs over this semiring, universality, inclusion and equivalence become decidable [40, 23].

For the boundedness problem, the situation is very different. A seminal paper [1] establishes undecidability of several classical decision problems for linearly ambiguous WAs over the integer min-plus semiring in a uniform fashion. However, it only proves boundedness to be undecidable for general (that is, exponentially ambiguous) WAs, and provides no classes with decidable boundedness. This indicates that boundedness is somehow different to other mentioned decision problems.

We remark that boundedness is PSPACE-complete for WAs over the *natural* min-plus semiring $(\mathbb{N} \cup \{+\infty\}, \min, +)$ [1, 22, 29, 37], which requires completely different techniques than the integer case. We also remark that it is decidable for copyless linear CRAs over the semiring of positive rational numbers with usual addition and multiplication, and is undecidable for general WAs over the same semiring [11]. Transferring any such results to the min-plus semiring is unlikely, since these semirings has very different properties.

Restricted classes of CRAs

One useful feature of CRAs is that, by adding syntactic restrictions on them, it is possible to introduce subclasses whose expressiveness is incomparable to known classes of WAs. This allows to obtain a finer decidability landscape compared to the case where only the formalism of WAs is used.

One notable example of that is the class of so called copyless linear CRAs. Informally, a CRA is called copyless if for every transition, the value of each of its registers can only be used once in the updates. Copyless linear CRAs are strictly less expressive than linearly ambiguous WAs, and their expressivity is incomparable to unambiguous WAs [2]. A further restriction of copyless linear CRAs to the case where the minimum operation is only allowed in the output function makes them equally expressive to finitely sequential WAs, which are unions of WAs whose underlying NFAs are deterministic [4, 13].

Restricting the number of registers in a class of CRAs also usually provides a subclass whose expressivity is incomparable to that of known classes of WAs. For example, there exists a copyless (but not linear) CRA with only 3 registers that computes a function not computed by any polynomially ambiguous WA [32]. For copyless linear CRAs restricted to only three registers universality is undecidable [15]. Moreover, there exist copyless linear CRAs with only two registers that are not equivalent to any unambiguous WA [2], see Example 6 on page 6 for one such CRA. All these results indicate that CRAs with few registers are quite expressive. Results on finding a CRA with the minimum number of registers computing a given function are presented in [13, 14, 25, 26].

More generally, for many problems the case of automata with two counters or registers often turns out to be already difficult enough. For example, most decision problems are undecidable for two-counter automata [33]. For vector addition systems with states, decidability of the reachability problem in the two-counter case was established in a seminal paper [24] several years before the proof that it is decidable for arbitrary number of counters was found [30], and it took over 20 more years to establish the precise computational complexity of the two-counter case [7].

Our contributions

The main technical contribution of the paper, Theorem 17, states that boundedness is decidable for copyless linear CRAs with resets with at most two registers. Namely, we show that it is NL-complete if the numbers in the updates are presented in unary, and is in coNP is they are presented in binary. Functions computed by CRAs, even when they have only two

registers, are highly nonlinear and complex, mainly due to the presence of nested minimum operations. We illustrate this by providing in Section 6 a series of CRAs with very long shortest runs outputting a given value. To show that the two-register case is decidable, we identify several possible shapes of small witnesses of unboundedness (Section 4.2). The main challenge is then showing that if a CRA is unbounded, it contains one of these witnesses, which we do by carefully analysing the growth of the output value for a run providing a large enough value, and showing how to rearrange the cycles of this run to obtain a witness (Section 4.3). This requires in particular some geometrical arguments on the cones generated by the weight vectors of the cycles.

Our second contribution is establishing the complexity of boundedness for copyless linear CRAs with resets where the number of registers is arbitrary, but the minimum operation only occurs in the output function. As mentioned above, such CRAs compute the same class of functions as finitely sequential WAs. We show that boundedness is PSPACE-complete if registers are allowed to transfer values to other registers (Theorem 33), and is coNP-complete otherwise (Theorem 32). For upper bounds on the complexity, our techniques again rely on the combinatorial and geometrical analysis of cycles.

2 Main definitions

Symbols $\mathbb{N}$ and $\mathbb{Z}$ stand for natural and integer numbers respectively. Let $\mathbb{N}_+ \coloneqq \mathbb{N} \setminus \{0\}$. We assume that the reader is familiar with the basic concepts in the area of formal languages and automata, see e.g. [38]. The Kleene star is denoted as $(\cdot)^*$. In regular expressions, we write $\cup$ for the sum and ε for the empty string. The language of a nondeterministic finite automaton (NFA) $\mathcal{A}$ is denoted as $\mathcal{L}(\mathcal{A})$. When speaking of underlying digraphs of NFAs, we use standard terms like simple cycle, reachability and backwards-reachability. We emphasise that by a simple cycle we mean a cycle that does not visit the same vertex more than once, except for its first and last vertex. General cycles do not have this restriction.

In this paper, we focus on CRAs over the integer min-plus semiring $(\mathbb{Z} \cup \{+\infty\}, \min, +)$. Hence, in what follows, we fix $\mathbb{K} \coloneqq (\mathbb{Z} \cup \{+\infty\}, \min, +)$. For the proofs we present, the main focus lies on the operations on registers performed by CRAs, so we look at them in depth in Section 2.1. Then, in Section 2.2, we define CRAs and the boundedness problem.

2.1 Expressions, valuations and substitutions

Fix a finite set of variables X. By $\mathrm{Expr}(X)$ we denote the set of *expressions* constructed using operations $\min\{\cdot,\cdot\}$ and $+$, variables from X and constants from $\mathbb{K}$. Expressions can be seen as polynomials over $\mathbb{K}$. Due to associativity, we allow arbitrary arity of the semiring operations in the expression notation. For an expression $e \in \mathrm{Expr}(\emptyset) \subseteq \mathrm{Expr}(X)$ without variables, we denote by $\mathrm{eval}(e) \in \mathbb{K}$ its value.

A *substitution over* X is a function $\nu\colon X \to \mathrm{Expr}(X)$. We denote the set of all substitutions over X by $\mathrm{Sub}(X)$. When defining substitutions, we often treat them as sets of "*argument* $\leftarrow$ *value*" pairs. When X is clear from the context, we implicitly extend partial substitutions with the identity mapping for the omitted arguments. For example, if $X' = \{x_1, x_2\} \subsetneq X$, then $\nu = \{x_1 \leftarrow e_1,\ x_2 \leftarrow e_2\}$ denotes a substitution satisfying $\nu(x_i) = e_i$ for $x_i \in X'$ and $\nu(x) = x$ for $x \in X \setminus X'$. A *valuation over* X is a substitution $\mu\colon X \to \mathbb{K}$. We denote by $\mathrm{Val}(X) \subset \mathrm{Sub}(X)$ the set of all valuations over X. We write $\mathbf{0} \in \mathrm{Val}(X)$ for the valuation with $\mathbf{0}(x) = 0$ for every $x \in X$. We assume that there is a fixed order on the set of registers, and thus in particular consider valuations equivalently as vectors.

▶ **Example 1** (An expression, a substitution and a valuation). Fix $X \coloneqq \{x, y, z\}$.

$$\begin{aligned} e &\coloneqq \min\{10, x+5, y+z\} && \in \mathrm{Expr}(X) && \text{(an expression)} \\ \nu &\coloneqq \{x \leftarrow 2 + \min\{x, y\},\ y \leftarrow 3\} && \in \mathrm{Sub}(X) && \text{(a substitution)} \\ \mu &\coloneqq \{x \leftarrow 2,\ y \leftarrow 5,\ z \leftarrow 10\} && \in \mathrm{Val}(X) && \text{(a valuation)} \end{aligned}$$

Substitutions can be *applied* to expressions: given $e \in \mathrm{Expr}(X)$ and $\nu \in \mathrm{Sub}(X)$, by $e[\nu]$ we denote the result of simultaneously replacing each occurrence of x with $\nu(x)$ for every $x \in X$. Substitutions can be composed: for $\nu, \nu' \in \mathrm{Sub}(X)$, we define the *composition* $(\nu; \nu') \in \mathrm{Sub}(X)$ as $(\nu; \nu')(x) = \nu'(x)[\nu]$. Applying a valuation μ to an expression e yields an expression without variables – thus, with a defined value from $\mathbb{K}$. We hence define $\mathrm{eval}_\mu(e) \coloneqq \mathrm{eval}(e[\mu])$. For a valuation μ and a substitution ν, we let $\mathrm{eval}_\mu(\nu) \coloneqq \mathrm{eval}(\mu; \nu)$.

▶ **Example 2** (Application and composition of substitutions). Continuing Example 1, we have

$$\begin{aligned} e[\nu] &= \min\{10, (2 + \min\{x, y\}) + 5, (3) + (z)\} && \in \mathrm{Expr}(X) && (e \text{ with } \nu \text{ applied to it}) \\ e[\mu] &= \min\{10, (2) + 5, (5) + (10)\} && \in \mathrm{Expr}(\emptyset) && (e \text{ with } \mu \text{ applied to it}) \\ \mathrm{eval}_\mu(e) &= \mathrm{eval}(e[\mu]) = 7 && \in \mathbb{K} && \text{(the value of } e[\mu]) \end{aligned}$$

For $e, e' \in \mathrm{Expr}(X)$, we write $e \equiv e'$ if $\mathrm{eval}_\mu(e) = \mathrm{eval}_\mu(e')$ for every $\mu \in \mathrm{Val}(X)$. Additionally, for $\nu, \nu' \in \mathrm{Sub}(X)$, we write $\nu \equiv \nu'$ whenever $\nu(x) \equiv \nu'(x)$ for every $x \in X$. For an expression e, by $\mathrm{maxc}(e)$ we denote the maximal absolute value of constants different to $+\infty$ appearing in e, and 0 if there are none. We extend maxc naturally to substitutions.

Copyless linear substitution with resets

In this paper, our main focus is on a special family of substitutions which are *copyless* and *linear with resets*. An expression e *is in a canonical linear form* if

$$e = \min\{x_1 + c_1, x_2 + c_2, \ldots, x_k + c_k\}$$

for some pairwise-different $x_1, \ldots, x_k \in X$, $c_1, \ldots, c_k \in \mathbb{K}$, and $k \in \mathbb{N}$. Any expression e' such that $e' \equiv e$ for an expression e in a canonical linear form is called *linear*. A substitution ν is *linear with resets* if for every $x \in X$, $\nu(x)$ is either linear or 0 (a *reset*). A linear substitution with resets is *in a canonical form* if all its linear expressions are in a canonical linear form. We remark that the only reason why we use linear substitutions with resets instead of affine substitutions (that is, substitutions whose expressions are sums of linear and constant expressions) is to simplify the presentation of our techniques. Clearly, by adding more registers one can transform an affine CRA into a linear CRA with resets.

A substitution ν is *copyless* if for all pairs $x, x' \in X$ such that $x \neq x'$ the expressions $\nu(x)$ and $\nu(x')$ feature disjoint sets of variables. By $\mathrm{Expr}_{\mathrm{lin}}(X)$ and $\mathrm{Sub}_{\mathrm{clr}}(X)$ we denote the sets of linear expressions and copyless linear substitutions with resets, respectively.

▶ **Example 3** (Copyless linear substitutions with resets). Consider the substitutions ν and ν':

$$\nu \coloneqq \begin{cases} x \leftarrow y + 5 \\ y \leftarrow \min\{x, z - 2\} \\ z \leftarrow 5 - 5 \end{cases} \equiv \begin{cases} x \leftarrow \min\{y + 5\} \\ y \leftarrow \min\{x + 0, z + (-2)\} \\ z \leftarrow 0 \end{cases}, \qquad \nu' \coloneqq \begin{cases} x \leftarrow \min\{x\} \\ y \leftarrow \min\{x, y\} \end{cases}$$

We have that $\nu \in \mathrm{Sub}_{\mathrm{clr}}(\{x, y, z\})$ because it is equivalent to a linear substitution with resets in a canonical form, and variables x, y, z occur at most once in its expressions. In contrast, ν' is linear, but not copyless, because x occurs in both $\nu'(x)$ and $\nu'(y)$.

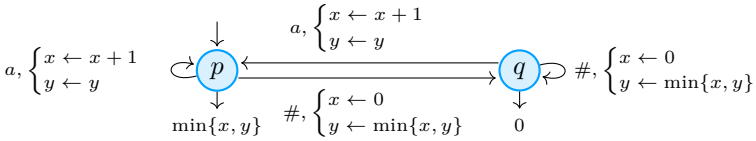


Figure 1 An example of a CRA over the semiring $\mathbb{K}$.

2.2 Cost register automata

▶ **Definition 4** (CRA). *A* copyless linear CRA with resets over $\mathbb{K}$ *is a tuple*

$$\mathcal{C} = (X, Q, \Sigma, \delta, q_{\text{ini}}, \text{out})$$

consisting of a finite set X of registers, a finite alphabet Σ, a finite set Q of control states, an initial state $q_{\text{ini}} \in Q$, and two functions:

- *a transition rule function* $\delta \colon Q \times \Sigma \to Q \times \text{Sub}_{\text{clr}}(X)$,
- *an output function* $\text{out} \colon Q \to \text{Expr}_{\text{lin}}(X)$.

We also make the following assumptions that clearly preserve the property of being bounded or unbounded. We assume that all substitutions and expressions in $\mathcal{C}$ are in a canonical form. Furthermore, we assume that $+\infty$ never occurs as a constant in a substitution, since every such transition can be replaced with a transition leading to a new state, a self-loop incrementing the value of the corresponding register in this new state, and then a transition back.

In this paper, we consider only copyless linear CRAs with resets, hence for the rest of the paper we simply call them CRAs.

▶ **Definition 5** (Semantics of a CRA). *A CRA $\mathcal{C} = (X, Q, \Sigma, \delta, q_{\text{ini}}, \text{out})$ naturally induces a (possibly infinite) deterministic labelled transition system $[\![\mathcal{C}]\!] := (\text{Conf}_{\mathcal{C}}, \Sigma, \Delta_{\mathcal{C}}, c_{\text{ini}})$, where $\text{Conf}_{\mathcal{C}} := Q \times \text{Val}(X)$ is the set of* configurations, *$c_{\text{ini}} := (q_{\text{ini}}, \mathbf{0})$ is the* initial configuration *and $\Delta_{\mathcal{C}} \colon \text{Conf}_{\mathcal{C}} \times \Sigma \to \text{Conf}_{\mathcal{C}}$ is a transition function defined as $\Delta_{\mathcal{C}}((q, \mu), \sigma) := (q', \text{eval}_{\mu}(\nu))$, where $(q', \nu) = \delta(q, \sigma)$. We extend $\Delta_{\mathcal{C}}$ to words as usually: for a configuration t, a letter $\sigma \in \Sigma$ and a word $w \in \Sigma^*$, $\Delta_{\mathcal{C}}(t, \sigma w) := \Delta_{\mathcal{C}}(\Delta(t, \sigma), w)$ and $\Delta_{\mathcal{C}}(t, \varepsilon) := t$. We further define the function $\mathcal{C} \colon \Sigma^* \to \mathbb{K}$ calculated by $\mathcal{C}$ as $\mathcal{C}(w) = \text{eval}_{\mu}(\text{out}(q))$, where $(q, \mu) = \Delta(c_{\text{ini}}, w)$. Note that since we are only interested in the boundedness problem, the assumption that the initial value of each register is zero does not restrict the expressiveness of CRAs.*

▶ **Example 6.** In Figure 1, we give an example of a CRA over the semiring $\mathbb{K}$ and alphabet $\Sigma = \{a, \#\}$. It has two states p and q, where p is the initial state, and two registers x and y. The output function for state p (words ending with a and the empty word) is $\min\{x, y\}$ and for state q (words ending with $\#$) is 0. Both registers are initialised with value 0. If the input word ends with $\#$ or is empty, this CRA outputs 0. Call a sequence of consecutive a's a *block*. A block is *maximal* if it is not contained in a longer block. If the input word ends with a, the CRA outputs the length of the shortest maximal block. Register x stores the number of a's read in the current block, and y stores the minimum length of maximal blocks read so far. This CRA is a copyless linear CRA with resets.

We say that a CRA $\mathcal{C}$ is *bounded* if there is $N \in \mathbb{N}$ such that $\mathcal{C}(w) < N$ for all $w \in \Sigma^*$. We are now ready to state the main decision problem we are interested in.

▶ **Problem 7** (CRA boundedness).
Input *CRA $\mathcal{C}$.*
Question *Is $\mathcal{C}$ bounded?*

3 Regular substitution languages

3.1 Substitution languages associated to CRAs

In the boundedness problem, the input alphabet of a CRA is redundant, since the formulation is existentially quantified for the word and the underlying finite automaton is deterministic. For this reason, in this paper we focus on sequences of substitutions that a CRA can perform.

A *language of substitutions* is an arbitrary subset of $\mathrm{Sub}_{\mathrm{clr}}(X)^*$. Every word $w \in \Sigma^*$ read by a CRA $\mathcal{C}$ induces a sequence of substitutions that $\mathcal{C}$ performs when reading w. Fix a CRA $\mathcal{C} = (X, \Sigma, Q, q_{\mathrm{ini}}, \delta, \mathrm{out})$. We define the language of substitutions $\mathcal{L}_x(\mathcal{C})$ induced by it. Observe that the set of substitutions that occur in the transitions and output expressions of $\mathcal{C}$ is finite. We denote it by $\Gamma_{\mathcal{C},x}$. The regular language $\mathcal{L}_x(\mathcal{C})$ is then formally defined as the language of the following automaton $\mathrm{NFA}_x(\mathcal{C})$. To simplify the presentation, we assume that the last substitution in the sequence corresponding to a word saves the output into a designated output register $x \in X$.

▶ **Definition 8** ($\mathrm{NFA}_x(\mathcal{C})$). *Given a CRA $\mathcal{C} = (X, Q, \Sigma, \delta, q_{\mathrm{ini}}, \mathrm{out})$, define a nondeterministic finite automaton $\mathrm{NFA}_x(\mathcal{C}) \coloneqq (Q', \Gamma_{\mathcal{C},x}, \delta', q_{\mathrm{ini}}, \{q_{\mathrm{fin}}\})$ where $Q' \coloneqq Q \cup \{q_{\mathrm{fin}}\}$ and the transition relation $\delta' \subseteq Q' \times \Gamma_{x,\mathcal{A}} \times Q'$ contains transitions:*

$$(q, \nu, q') \in \delta' \qquad \text{for every } q, q' \in Q, \text{ and } \nu, \sigma \text{ such that } \delta(q, \sigma) = (q', \nu),$$
$$(q, \{x \leftarrow \mathrm{out}(q)\}, q_{\mathrm{fin}}) \in \delta' \qquad \text{for every } q \in Q.$$

Fix an NFA $\mathcal{A} = (Q, S, \delta, q_{\mathrm{ini}}, Q_{\mathrm{fin}})$ and a set of registers $X = \{x_1, \ldots, x_d\}$ for the rest of this subsection. An alternating sequence $\pi = q_0 \xrightarrow{\nu_1} q_1 \xrightarrow{\nu_2} q_2 \to \cdots \to q_{n-1} \xrightarrow{\nu_n} q_n$ of states from Q and letters from S is called *a run in $\mathcal{A}$ labelled by the word $w = \nu_1\nu_2\cdots\nu_n$*. We write $q_0 \xrightarrow{\pi,w}\!\!\!\!\rightarrow\, q_n$ to denote the fact that π is a run labelled by w that begins in q_0 and ends in q_n. For such a run and $\mu \in \mathrm{Val}(X)$, we define $\mathrm{eval}_\mu(\pi) \coloneqq \mathrm{eval}_\mu(w)$. We identify words with compositions of the corresponding sequences of substitutions. The set of runs of $\mathcal{A}$ is denoted by $\mathrm{Runs}(\mathcal{A})$. A run π is *accepting* if $q_{\mathrm{ini}} \xrightarrow{\pi}\!\!\!\!\rightarrow\, q_{\mathrm{fin}}$. An NFA $\mathcal{A}$ *accepts* $w \in S^*$ whenever there exists an accepting run of $\mathcal{A}$ labelled by w. The *language of* $\mathcal{A}$, denoted $\mathcal{L}(\mathcal{A})$, is the set of words accepted by $\mathcal{A}$.

To be able to refer to segments of runs, we combine the notation for single transitions $p \xrightarrow{\nu} q$ and runs $q \xrightarrow{\pi,w}\!\!\!\!\rightarrow\, r$. For example, we may consider a run $\pi = p \xrightarrow{\nu} q \xrightarrow{\pi',w}\!\!\!\!\rightarrow\, r$ labelled by a word $\nu \cdot w$. When the labelling is not important, we write $q \xrightarrow{\pi}\!\!\!\!\rightarrow\, r$.

There is an obvious correspondence between runs in $[\![\mathcal{C}]\!]$ and in $\mathrm{NFA}_x(\mathcal{C})$. In particular, for $k \in \mathbb{K}$, there exists $w \in \Sigma^*$ such that $\mathcal{C}(w) = k$ if, and only if, there exists $u \in \mathcal{L}(\mathcal{A})$ such that $\mathrm{eval}_{\mathbf{0}}(u)(x) = k$. This allows us to restate the boundedness problems in terms of languages of substitutions. We say that a regular language $L \subseteq \mathrm{Sub}_{\mathrm{clr}}(X)^*$ *has bounded output* in $x \in X$ if there exists $N \in \mathbb{N}$ such that for every $w \in L$ we have $\mathrm{eval}_{\mathbf{0}}(s)(x) < N$.

▶ **Problem 9** (Boundedness of regular d-register substitution languages).

Input *NFA $\mathcal{A}$ over a finite set $S \subset \mathrm{Sub}_{\mathrm{clr}}(X)$, $|X| = d$, output register $x \in X$.*

Question *Does $\mathcal{L}(\mathcal{A})$ have bounded output in x?*

Note that boundedness for CRAs (Problem 7) easily reduces to this problem in deterministic logarithmic space. Indeed, it suffices to compute $\mathrm{NFA}(\mathcal{A})$ for a given CRA with Definition 8. This reformulation allows us to use a rich framework of regular languages, which streamlines the proofs presented in later sections.

▶ **Definition 10** (Elementary substitutions). *We say that a substitution $\nu \in \mathrm{Sub}_{\mathrm{clr}}(X)$ is* elementary *if it has one of the following forms for some $x, y \in X$, $c \in \mathbb{K}$:*

$$\{x \leftarrow x + c\} \quad \text{(additive sub.)} \qquad \{x \leftarrow y,\ y \leftarrow x\} \quad \text{(transposition)}$$
$$\{x \leftarrow \min\{x, y\},\ y \leftarrow 0\} \quad \text{(minimum sub.)} \qquad \{x \leftarrow 0\} \quad \text{(reset sub.)}$$

Let $\mathrm{Sub}_{\mathrm{elem}}(X) \subseteq \mathrm{Sub}_{\mathrm{clr}}(X)$ be the set of elementary substitutions, and let $T_X \cup R_X \cup A_X \cup M_X$ be its partition into sets of transpositions, and reset, additive, and minimum substitutions.

▶ **Lemma 11.** *For every $\nu \in \mathrm{Sub}_{\mathrm{clr}}(X)$ in a canonical form, there exists a word of substitutions $u \in \mathrm{Sub}_{\mathrm{elem}}(X)^*$ of length $O(d^2)$ such that $u \equiv \nu$.*

Note that the statement of the above lemma is not true in the general setting of $\mathrm{Sub}(X)$, as its proof relies on copylessness. Note also that Lemma 11 implies the existence of a homomorphism $\text{to-elem} \colon \mathrm{Sub}_{\mathrm{clr}}(X)^* \to \mathrm{Sub}_{\mathrm{elem}}(X)^*$ such that $\nu \equiv \text{to-elem}(\nu)$ for every $\nu \in \mathrm{Sub}_{\mathrm{clr}}(X)$. For a finite set $S \subset \mathrm{Sub}(X)$, we define $\mathrm{maxc}(S) \coloneqq \max\{\mathrm{maxc}(\nu) \mid \nu \in S\}$. The following two claims are not difficult to prove.

▷ Claim 12 (Maximal constant grows linearly w.r.t. length). Fix a finite $S \subset \mathrm{Sub}_{\mathrm{elem}}(X)$. For every $w \in S^*$, we have $\mathrm{maxc}(w) \leq |w| \cdot \mathrm{maxc}(S)$.

▷ Claim 13 (Elementary substitutions assumption). We may assume w.l.o.g. that the alphabet S of $\mathcal{A}$ consists only of elementary substitutions, i.e., $S \subset \mathrm{Sub}_{\mathrm{elem}}(X)$.

3.2 The structure of witnesses with additive and reset substitutions only

In this subsection, we show how to simplify and decompose runs that do not contain any minimum substitutions or transpositions. We then use these results in the complexity upper bounds and to analyse runs with a more complex structure.

For the rest of this subsection, fix a set of registers $X = \{x_1, \ldots, x_d\}$, an output register $x \in X$, and an NFA $\mathcal{A}$ over a finite alphabet $S \subset A_X \cup R_X \subset \mathrm{Sub}_{\mathrm{elem}}(X)$ of elementary additive and reset substitutions. Similarly to Claim 13, this covers a more general case where the alphabet of $\mathcal{A}$ consists of substitutions adding integer values to some registers and resetting other registers.

Let $w \in A_X^*$ be such that $w \equiv \{x_1 \leftarrow x_1 + c_1, \ldots, x_d \leftarrow x_d + c_d\}$ for some $c_1, \ldots, c_d \in \mathbb{K}$. We define $\mathrm{eff}(w) \coloneqq (c_1, \ldots, c_d) \in \mathbb{K}^X$, and we call it the *effect* of w. The *integer conic hull* $\mathrm{Cone}_{\mathbb{N}}(V)$ of a set of vectors V is the set of linear combinations of vectors from V with nonnegative integer coefficients. The following theorem is a direct consequence of a classical result by Carathédory, see, e.g., [35].

▶ **Theorem 14** (Only d vectors are sufficient to represent a positive point). *Let $V \subseteq \mathbb{Z}^d$. If all components of a vector $\vec{b}$ are strictly positive and $\vec{b} \in \mathrm{Cone}_{\mathbb{N}}(V)$, then there exists $V' \subseteq V$ with $|V'| \leq d$ and a constant $\lambda > 0$ such that $\lambda\vec{b} \in \mathrm{Cone}_{\mathbb{N}}(V')$.*

We now state two important lemmas describing the shape of runs labelled by words over A_X and $A_X \cup R_X$, respectively.

▶ **Lemma 15** (Decomposition lemma for additive runs). *For every run $q \xrightarrow{\pi, w}\!\!\!\!\rightarrow q'$ of $\mathcal{A}$ such that $w \in A_X^*$, there exist $n \in \mathbb{N}$, words $w_1, \ldots, w_n, , w', z_1, \ldots, z_{n+1} \in A_X^*$, and integers $a_1, \ldots, a_n \in \mathbb{N}$ such that*

- *for every word $z \in z_1 w_1^* z_2 \ldots z_n w_n^* z_{n+1}$, there exists a run $q \xrightarrow{z}\!\!\!\!\rightarrow q'$,*
- $\mathrm{eff}(w) = \mathrm{eff}(w') + a_1\mathrm{eff}(w_1) + \cdots + a_n\mathrm{eff}(w_n)$, *and*
- $|w_1|, \ldots, |w_n|, |w'| \leq |Q|$ *and* $|z_1 \ldots z_{n+1}| \leq |Q|^2$.

Proof idea. We iterate a process that eliminates all the simple cycles from π. These simple cycles are labelled by some words $w_1, \dots, w_d$ and the process returns a cycle-free run π' that is labelled by a word w'. We can describe the additive effect of the run π as the effect of w' plus the effect of all the simple cycles that we eliminated. Finally, we argue there exists a short run from q to q' that contains a vertex from each of these simple cycles. ◀

Proof. Consider the following iterative process on π. Initialise $\vec{b} = \mathbf{0} \in \mathbb{Q}^X$.

- Identify the first simple cycle $r \xrightarrow{\pi_i, w_i}\!\!\!\!\rightarrow r$ in π and replace it by r. By simple cycle we mean a run where only the first and last states are equal.
- Update the value of the vector $\vec{b}$ to $\vec{b} + \mathrm{eff}(w_i)$.

Let $q \xrightarrow{\pi', w'}\!\!\!\!\rightarrow q'$ be the resulting run. This process guarantees that π' is cycle-free and that $\mathrm{eff}(w) = \mathrm{eff}(w') + \vec{b} = \mathrm{eff}(w') + a_1\mathrm{eff}(w_1) + \cdots + a_n\mathrm{eff}(w_n)$, where a_i is the number of times we have eliminated a simple cycle with effect w_i, for all $1 \leq i \leq n$.

Let $Q' \subseteq Q$ be the set of states visited by π. The shortest run that visits all states of Q' has length at most $|Q|^2$. Since every word w_i corresponds to a simple cycle eliminated by the process, it follows that there exists words $z_1, \dots, z_{n+1}$ such that for every word $z \in z_1 w_1^* z_2 \dots z_n w_n^* z_{n+1}$, there exists a run $q \xrightarrow{z}\!\!\!\!\rightarrow q'$ and $|z_1 \dots z_{n+1}| \leq |Q|^2$ even if n can be as large as $2^{|Q|}$. ◀

▶ **Lemma 16** (Pumping lemma). *If $S \subseteq A_X \cup R_X$, then there exists a word $w \in \mathcal{L}(\mathcal{A})$ with $\mathrm{eval}_{\mathbf{0}}(w) > 2d|Q|C$, where $C \coloneqq \mathrm{maxc}(S)$ if and only if there exist words $\alpha_1, \dots, \alpha_{d+1}$, $\beta_1, \dots, \beta_d \in A_X^*$ such that*

- $\alpha_1 \beta_1^+ \alpha_2 \dots \alpha_d \beta_d^+ \alpha_{d+1} \eta_X \subseteq \mathcal{L}(\mathcal{A})$,
- $|\beta_1|, \dots, |\beta_d| \leq |Q|$
- $|\alpha_1 \dots \alpha_{d+1}| < (d+1)|Q|^2$, *and*
- *for each $N \in \mathbb{N}$, there are $a_1, \dots, a_d \in \mathbb{N}$ with $\mathrm{eval}_{\mathbf{0}}(\alpha_1 \beta_1^{a_1} \alpha_2 \dots \alpha_d \beta_d^{a_d} \alpha_{d+1} \eta_X)(x_1) > N$.*

Proof idea. Given a run in $\mathcal{A}$ of sufficiently large value, we can split it into different segments such that in each segment we know for every register if it is going to be reset in the future or not. Thus, for every register, we can determine if a segment of the run is relevant for determining its final value. Subsequently, we identify all cycles of this run and argue that since all the registers hold large values at the end of the run, the total !! effect of the relevant cycles (the ones after the last reset of the register) must be a large positive number for each register. This allows us to use Lemma 15 to conclude that we do not have too many such cycles and that we can pump them up in order to achieve unbounded register values. ◀

Proof. The right to left implication of this lemma is immediate, so we focus on the left to right implication. Assume there exists a run $\pi = q_{\mathrm{ini}} \xrightarrow{w} q_{\mathrm{fin}}$, for some $q_{\mathrm{fin}} \in Q_{\mathrm{fin}}$ such that $\mathrm{eval}_{\mathbf{0}}(w)(x_1) > 2d|Q|C$.

We can assume that the value of each register is reset at least once along π, since we can add additional reset substitutions to the beginning of w without changing the value of $\mathrm{eval}_{\mathbf{0}}(w)(x_1)$. For each i, $1 \leq i \leq d$, the ν_i be the last reset substitution for register x_i in w. Without loss of generality, we can assume that in w the registers are reset for the last time in the increasing order of their indices. Then π can be represented as

$$\pi = q_{\mathrm{ini}} \xrightarrow{\pi_1, w_1}\!\!\!\!\rightarrow q_1 \xrightarrow{\nu_1} q_1' \xrightarrow{\pi_2, w_2}\!\!\!\!\rightarrow q_2 \xrightarrow{\nu_2} \dots \xrightarrow{\pi_d, w_d}\!\!\!\!\rightarrow q_d \xrightarrow{\nu_d} q_d' \xrightarrow{\pi_{d+1}, w_{d+1}}\!\!\!\!\rightarrow q_{d+1} \xrightarrow{\eta_X} q_{\mathrm{fin}}\,.$$

Observe that for each i, $1 \leq i \leq d-1$, we can replace in w_i all reset substitutions of registers $x_{i+1}, \dots, x_d$ with the identity substitution without changing the value of $\mathrm{eval}_{\mathbf{0}}(w)(x_1)$.

Thus, $w_1, \dots, w_{d+1}$ become words over additive substitutions only. Hence we now get that $\mathrm{eval}_{\mathbf{0}}(w_1\nu_1 \cdots w_{d+1}) = \mathrm{eff}(w_1) + \cdots + \mathrm{eff}(w_{d+1})$. By applying Lemma 15 to each run $\pi_1, \dots, \pi_{d+1}$, we obtain that

$$\begin{aligned}\mathrm{eval}_{\mathbf{0}}(w_1\nu_1 \dots w_{d+1}) = \quad & \mathrm{eff}\left(w'_1\right) \quad + \quad a_1^{(1)}\mathrm{eff}\left(w_1^{(1)}\right) \quad + \cdots + \quad a_{n_1}^{(1)}\mathrm{eff}\left(w_{n_1}^{(1)}\right) \\ & \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ & \mathrm{eff}\left(w'_{d+1}\right) + a_1^{(d+1)}\mathrm{eff}\left(w_1^{(d+1)}\right) + \cdots + a_{n_{d+1}}^{(d+1)}\mathrm{eff}\left(w_{n_{d+1}}^{(d+1)}\right)\end{aligned}$$

where $w_1^{(1)}, \dots, w_{n_{d+1}}^{(d+1)}, w'_1, \dots, w'_{d+1}$ are all words of length smaller than $|Q|$. Since

$$\mathrm{eval}_{\mathbf{0}}(w_1\nu_1 \cdots w_{d+1})(x) > 2d|Q|C \qquad \text{and} \qquad \mathrm{eff}\left(w'_1 \cdots w'_{d+1}\right)(x) \leq (d+1)|Q|C$$

for every $x \in X$, there must exists a vector $\vec{b} \in \mathbb{N}_+^X$ such that $\vec{b} \in \mathrm{Cone}_{\mathbb{N}}\left(\left\{w_1^{(1)}, \dots, w_{d+1}^{(d+1)}\right\}\right)$. By Theorem 14, there exists a subset $\{\beta_1, \dots, \beta_d\} \subseteq \left\{w_1^{(1)}, \dots, w_{d+1}^{(d+1)}\right\}$ such that $\vec{b}' \in \mathrm{Cone}_{\mathbb{N}}(\{\beta_1, \dots, \beta_d\})$ and $\vec{b}' > \mathbf{0}$ (pointwise).

It remains to show that there exist words $\alpha_1, \dots, \alpha_{d+1}$ such that $|\alpha_1 \dots \alpha_{d+1}| < (d+1)|Q|^2$ and $\alpha_1\beta_1^+\alpha_2 \dots \alpha_d\beta_d^+\alpha_{d+1}\eta_X \in \mathcal{L}(\mathcal{A})$. Indeed, consider a run π_i for some $1 \leq i \leq d+1$. Either π_i contains no cycles from $\{\beta_1, \dots, \beta_d\}$, in which case there exists a word w'_i of length at most $|Q|$ corresponding to the run π'_i, or it does contain some cycles $\{\beta_j, \dots, \beta_{j+h}\}$ from $\{\beta_1, \dots, \beta_d\}$. In the latter case, we can use Lemma 15 to argue that there exist words $z'_j, \dots, z'_{j+h+1}$ such that for every word $z \in z'_j\beta_j^* z'_{j+1} \dots z'_{j+h}\beta_{j+h}^* z'_{j+h+1}$, there exists a run $q'_{i-1} \xrightarrow{z} q_i$. In both cases, either $|w'| \leq |Q|$ or $|z'_j \dots z'_{j+h+1}| \leq |Q|^2$. Thus, we can obtain the required words $\alpha_1, \dots, \alpha_{d+1}$ by possibly concatenating the w'_i and z'_i words that we identified. ◀

4 CRAs with two registers

In this section, we prove the following result.

▶ **Theorem 17.** *The boundedness problem for CRAs with two registers is* NL*-complete if the numbers in the substitutions are presented in unary, and in* coNP *if they are in binary.*

By the results of the previous section, this theorem is equivalent to the following statement.

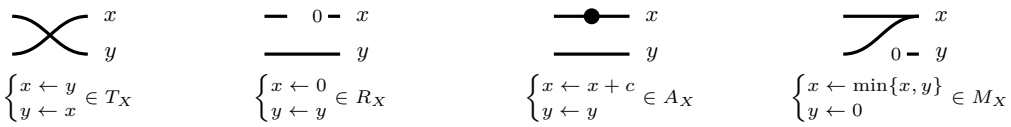
▶ **Proposition 18.** *Boundedness of regular* 2*-register substitution languages is* NL*-complete if the numbers in the substitutions are presented in unary, and in* coNP *if they are in binary.*

The remainder of the section is devoted to proving Proposition 18. For the rest of this section, fix the set of registers $X \coloneqq \{x, y\}$ and the output register x. Let us also fix for the rest of the section an input NFA $\mathcal{A} = (Q, S, \delta, q_{\mathrm{ini}}, Q_{\mathrm{fin}})$ such that $S \subset T_X \cup A_X \cup M_X \cup R_X \subset \mathrm{Sub}_{\mathrm{elem}}(X)$ (which we can assume by Claim 13). We begin by providing a series of simplifying assumptions for the automaton $\mathcal{A}$ in the input.

4.1 Simplifying assumptions

In this subsection, we introduce a normal form that significantly simplifies our arguments.

▶ **Definition 19** (Graphical notation for $\mathrm{Sub}_{\mathrm{elem}}(X)$)**.** *We depict elementary substitutions in* $\mathrm{Sub}_{\mathrm{elem}}(X)$ *as shown in Figure 2; details of the notation are discussed in its caption.*



(a) Transposition. **(b)** Reset substitution. **(c)** Additive substitution. **(d)** Minimum substitution.

Figure 2 Pictorial representation of elementary substitutions in $\mathrm{Sub}_{\mathrm{elem}}(\{x, y\})$. Register x is always drawn above y. A black dot stands for adding a constant – our graphical notation disregards the particular values of constants in the substitutions. A branching depicts the minimum operation. For the operations on y, the diagrams are symmetric. The effect of gluing several drawings together horizontally naturally corresponds to composition of depicted substitutions.

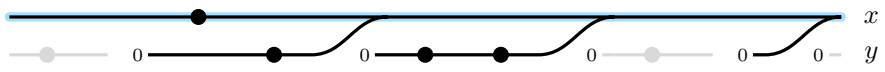
For a word $w = \nu_1\nu_2 \cdots \nu_n \in \mathrm{Sub}_{\mathrm{elem}}(X)^*$, we define the *output derivation tree* out-tree(w) as the derivation tree of the expression $w(x)$. This tree can have three kinds of leaves: constants 0 originating from reset or minimum substitutions, arbitrary constants from $\mathbb{K}$ coming from additive substitutions, and variables occurring in $\nu_1(x)$ or $\nu_1(y)$ of the first substitution ν_1. We say that a path from a leaf to the root in out-tree(w) is *leading* if its starting leaf comes from a substitution ν_i with the smallest i among all leaves that have a path to the root. A path is called *x-aligned* if all its vertices originate from expressions $(\nu_i(x))_{1 \le i \le n}$. A word w is called *x-aligned* if the leading path of out-tree(w) is x-aligned.

▶ **Example 20** (Depiction of out-tree(w))**.** For $w \in \mathrm{Sub}_{\mathrm{elem}}(X)^*$, out-tree$(w)$ corresponds to the tree rooted at the rightmost position corresponding to x in the pictorial representation. Consider

$$w := \begin{cases} x \leftarrow x \\ y \leftarrow y + 3 \end{cases} \begin{cases} x \leftarrow x \\ y \leftarrow 0 \end{cases} \begin{cases} x \leftarrow x + 2 \\ y \leftarrow y \end{cases} \begin{cases} x \leftarrow x \\ y \leftarrow y + 2 \end{cases} \begin{cases} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{cases} \begin{cases} x \leftarrow y \\ y \leftarrow x \end{cases} \begin{cases} x \leftarrow x + 3 \\ y \leftarrow y \end{cases}$$
$$\begin{cases} x \leftarrow x + 3 \\ y \leftarrow y \end{cases} \begin{cases} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{cases} \begin{cases} x \leftarrow x \\ y \leftarrow y + 5 \end{cases} \begin{cases} x \leftarrow x \\ y \leftarrow 0 \end{cases} \begin{cases} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{cases}$$

The depiction of w, in line with Definition 19, is as follows:

The out-tree(w) corresponds to the subgraph drawn in black. The leading path π of out-tree(w) is marked with a blue outline. Expressions in w corresponding to vertices of π are typeset on blue background. As not all of them come from $x \leftarrow e$ mappings, path π is not x-aligned. An x-aligned word w' such that $w(x) \equiv w'(x)$ has the following shape:



▶ **Lemma 21** (Leading branch x-aligned assumption)**.** *We can assume that each $w \in \mathcal{L}(\mathcal{A})$ is x-aligned.*

Therefore, in the remainder of this section we assume that each $w \in \mathcal{L}(\mathcal{A})$ is x-aligned. This means that only parts (b), (c) and (d) from Figure 2 (and not their symmetric y-counterparts) can be segments of runs.

4.2 Unboundedness witnesses

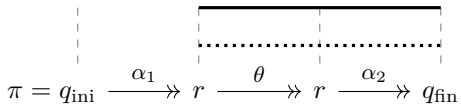
In this subsection, we define the shape of witnesses that prove unboundedness of CRAs. Next subsection shows that if a CRA is unbounded, it must contain such a witness. For the rest of this section, fix two substitutions

$$\eta = \{x \leftarrow \min\{x, y\},\ y \leftarrow 0\} \text{ and } \rho = \{x \leftarrow x,\ y \leftarrow 0\},$$

which will be referred to throughout the whole section. Let $N \coloneqq |Q|$ and $C \coloneqq \mathrm{maxc}(S)$. We introduce two new notations for special types of runs of $\mathcal{A}$:

used to signify that $w_1 \in A_X^*$ (i.e., has additive substitutions only), and $w_2 \in (A_X \cup \{\rho\})^*$.

▶ **Definition 22** (Unboundedness witness). *A run π in $\mathcal{A}$ is called a* trivial unboundedness witness *if it has the form*



such that $|\pi| \leq 3N$, $\mathrm{eff}(\theta)(x) > 0$ and $q_{\mathrm{fin}} \in Q_{\mathrm{fin}}$.

A run π in $\mathcal{A}$ is called a nontrivial unboundedness witness *if it has the form*

$$\pi = q_{\mathrm{ini}} \xrightarrow{\pi_a} q_a \xrightarrow{\pi_b} q_b \xrightarrow{\pi_c} q_c$$

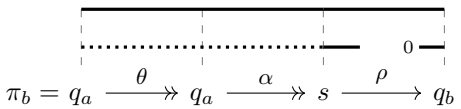
such that $|\pi_a| \leq N$, $|\pi_b|, |\pi_c| \leq 3N^2$, run π_b is pumpable, and π_c is sustainable, where pumpable and sustainable runs are defined below.

A trivial or nontrivial unboundedness witness is called just an unboundedness witness*.*

Intuitively, π_a from this definition is used to reach a gadget enabling pumping, π_b witnesses that we can pump up the value of x to an arbitrarily large number and then end up in q_b, and π_c certifies that we can maintain a large value of x to be output in $q_c \in Q_{\mathrm{fin}}$.

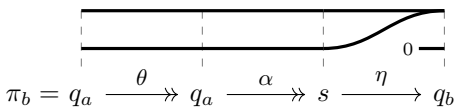
▶ **Definition 23** (Pumpable run). *A run π_b is called* pumpable *if it has one of the four forms:*

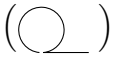
- **Type A.1** (a cycle with a positive effect on x, then a reset of y)



  such that θ is a cycle with $\mathrm{eff}(\theta)(x) > 0$ and α is a run with no η substitutions.

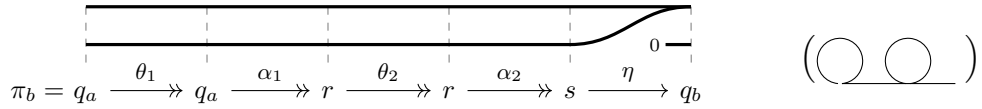
- **Type A.2** (a cycle with a positive effect on both x and y, then a minimum substitution)





  such that θ is a cycle with no η and no ρ substitutions and with $\mathrm{eff}(\theta) \in \mathbb{N}_+^2$, and α is a run with no η and no ρ substitutions.

- **Type A.3** (two cycles combining for a positive effect on both x and y, then a minimum)

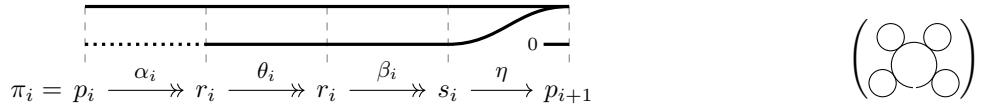


  such that θ_1, θ_2 are cycles with no η and no ρ substitutions and with $a_1\mathrm{eff}(\theta_1)+a_2\mathrm{eff}(\theta_2) \in \mathbb{N}^2_+$, for some $a_1, a_2 \in \mathbb{N}$. Furthermore, α_1, α_2 are runs with no η and no ρ substitutions.

- **Type B** (a cycle with a positive effect on x together with cycles supporting its value)

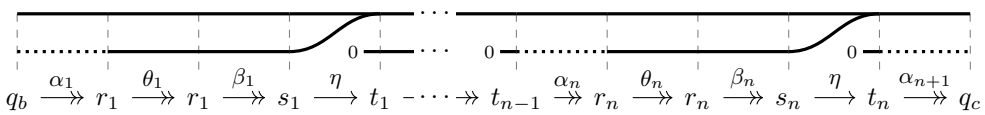
$$\pi = p_1 \xrightarrow{\pi_1} p_2 \xrightarrow{\pi_2} p_3 \twoheadrightarrow \cdots \twoheadrightarrow p_{n-1} \xrightarrow{\pi_{n-1}} p_n \xrightarrow{\pi_n} p_1$$

  for some $n \in \mathbb{N}$, where $p_1 = q_a = q_b$, and for every i, $1 \leq i \leq n$:



  such that α_i is a run with no η substitutions, θ_i is a cycle with no η and no ρ substitutions with $\mathrm{eff}(\theta_i) \in \mathbb{N} \times \mathbb{N}_+$ and β_i is a run with no η substitutions. Furthermore, we require $\mathrm{eff}(\alpha_1\theta_1\beta_1\alpha_2\theta_2\beta_2 \cdots \alpha_n\theta_n\beta_n)(x) > 0$.

▶ **Definition 24** (Sustainable run). *A run π_c is called* sustainable *if it is labelled by $w_c \in (A_X \cup \{\rho, \eta\})^*$ and has the following form:*



for $n \in \mathbb{N}$, runs α_i, β_i of the form as depicted in the picture, and cycles θ_i such that $\mathrm{eff}(\theta_i) \in \mathbb{Z} \times \mathbb{N}_+$ for every i, $1 \leq i \leq n$.

▶ **Proposition 25.** *Given $\mathcal{A}$, deciding if there exists a run in it which is an unboundedness witness is in* NL *if the numbers in the substitutions are presented in unary, and in* NP *if they are in binary.*

Proof. Intuitively, using nondeterminism, we can guess a nontrivial witness $\pi = \pi_a\pi_b\pi_c$ as in Definition 22 and verify that it has all the required properties. The case of a trivial witness is handled in a similar way and is thus omitted.

Indeed, starting in q_{ini}, we guess one transition at a time until we verify the existence of a witness or exceed the bound $N + 4N^2$ on its length. During the traversal, we guess the positions of states q_a, q_b, q_c at appropriate distances from q_{ini}. This splits the search into three phases corresponding to π_a, π_b and π_c. We need to verify that π_b is of one of four types of pumpable runs (cf. Definition 23). At any point in time, if the definition of a pumpable run requires it, we can guess that the current state r marks the start of the occurrence of a simple cycle θ. In this case, we store r, follow only labels from $A_X \cup \{\rho\}$ and compute the effect of the run until r occurs again. This can easily be done in NL or NP depending on the representation of the numbers in the substitutions. ◀

In order to complete the proof of Proposition 18, we need to show that the existence of an unboundedness witness is equivalent to the fact that $\mathcal{A}$ is not bounded. We show it by two implications stated below. We start with proving Lemma 26, and Lemma 29 is proved in the next subsection.

▶ **Lemma 26.** *If there exists an unboundedness witness then $\mathcal{L}(\mathcal{A})$ is not bounded.*

Proof. The proof is straightforward in case of a trivial unboundedness witness. Fix a nontrivial unboundedness witness π as in Definition 22:

$$\pi = q_{\text{ini}} \xrightarrow{\pi_a} q_a \xrightarrow{\pi_b} q_b \xrightarrow{\pi_c} q_c \,.$$

Fix an arbitrary number $N \in \mathbb{N}$. We construct a run π' such that $\text{eval}_{\mathbf{0}}(\pi')(x) \geq N$. We first prove that

▷ Claim 27. For every $M \in \mathbb{N}$ there is a run π'_b of $\mathcal{A}$ from q_a to q_b such that $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b) = (M', 0)$ for some $M' > M$.

First, by Claim 12 we have that $-CN \leq \text{eval}_{\mathbf{0}}(\pi_a)(x) \leq CN$. The claim is immediate if π_b contains a certificate of type A.1, A.2, or A.3. Indeed, we simply repeat the cycles that occur there a sufficient number of times. If π_b contains a certificate of type B, then $\pi_b = p_1 \xrightarrow{\pi_1} p_2 \xrightarrow{\pi_2} p_3 \twoheadrightarrow \cdots \twoheadrightarrow p_{n-1} \xrightarrow{\pi_{n-1}} p_n \xrightarrow{\pi_n} p_1$ is a cycle such that the overall effect on x is positive. Since every sub-path $\pi_1, \ldots, \pi_n$ contains a cycle with a positive effect on y and a non-negative effect on x, there is a path π''_b that repeats each such cycle $M + CN$ times. Note that π''_b is now a cycle with positive effect on x for any value of x smaller than $M + CN$. Thus, we can take π'_b to be the cycle π''_b taken $M + CN$ times.

This finishes the proof of the claim. Now, it suffices to show the following:

▷ Claim 28. For every $M \in \mathbb{N}$ there exists a run π'_c such that $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b \pi'_c)(x) \geq M$.

We prove this claim by induction on the number of occurrences of η in π_c. Assume that π_c does not contain any occurrences of η. By Claim 27, there exists a path π'_b such that $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b)(x) \geq M + CN$. Thus, $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b \pi_c)(x) \geq M$ by Claim 12, since the length of π_c is bounded.

Assume now that there is at least one occurrence of η in π_c. Then π_c can be represented as follows:

$$\pi_c = q_b \xrightarrow{\rho_1, \alpha_1} q_1 \xrightarrow{\rho_2, \alpha_2} q_2 \xrightarrow{\rho_3, \alpha_3} \cdots \xrightarrow{\rho_r, \alpha_r} q_{\text{fin}} \,,$$

where the cutting points are states reached after reading η. Assume that there is a run $q_{\text{ini}} \xrightarrow{\pi_{i-1}, w_{i-1}} q_{i-1}$ such that $\text{eval}_{\mathbf{0}}(w_{i-1})(x) \geq M'$, for every $M' \in \mathbb{N}$. Then we can use the cycle with positive effect on y inside ρ_i in order to conclude that there exists a run $q_{\text{ini}} \xrightarrow{\pi_i, w_i} q_i$ such that $\text{eval}_{\mathbf{0}}(w_i) \geq M'$, for every $M' \in \mathbb{N}$. This concludes the proof. ◀

▶ **Lemma 29.** *If $\mathcal{L}(\mathcal{A})$ is not bounded, there exists an unboundedness witness.*

4.3 Unboundedness implies the existence of a witness

Proof of Lemma 29. Assume that $\mathcal{L}(\mathcal{A})$ is not bounded. Let $M := 15C^2N^3$ and $W := 12CN^2$. Let π be the shortest accepting run of $\mathcal{A}$ such that $\text{eval}_{\mathbf{0}}(\pi)(x) > M + W$, and let w be the word labelling π. Since π is x-aligned, it can be split into two parts

$$\pi = q_{\text{ini}} \xrightarrow{\pi_{\text{pre}}} q_0 \xrightarrow{\pi_{\text{suf}}} q_{\text{fin}}$$

for some $q_0 \in Q$ and $q_{\text{fin}} \in Q_{\text{fin}}$, such that π_{suf} is the shortest suffix of π satisfying out-tree(π) = out-tree(π_{suf}). Note that $\text{eval}_{\mathbf{0}}(\pi_{\text{pref}})(x) = 0$, and π_{suf} features no substitution ν that resets x (i.e., for which $\nu(x) = 0$). Recall that we defined

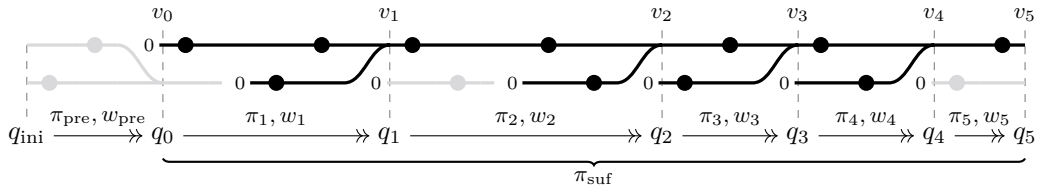
$$\eta = \{x \leftarrow \min\{x, y\},\ y \leftarrow 0\} \text{ and } \rho = \{x \leftarrow x,\ y \leftarrow 0\}.$$

Since π is x-aligned, we have that $w_{\text{suf}} \in (A_X \cup \{\eta, \rho\})^*$, where w_{suf} is the word labelling π_{suf}. Let $n \in \mathbb{N}$ be the number of substitutions η in w. Split the run π_{suf} into segments (some possibly empty)

$$q_0 \xrightarrow{\pi_1, w_1}\!\!\!\!\rightarrow\, q_1 \xrightarrow{\pi_2, w_2}\!\!\!\!\rightarrow\, q_2 \twoheadrightarrow \cdots \twoheadrightarrow q_n \xrightarrow{\pi_{n+1}, w_{n+1}}\!\!\!\!\rightarrow\, q_{n+1}$$

such that $q_1, \ldots, q_n$ are all the states reached directly after reading η each time. Define $v_i \coloneqq \text{eval}_{\mathbf{0}}(\pi_{\text{pref}}\pi_1 \cdots \pi_i)(x)$ for $0 \leq i \leq n+1$. Observe that $v_0 = 0$ and $v_{n+1} \geq M + W$.

▶ **Example 30.** Consider a run π partitioned into segments as defined above:



Let us overlook the slight inaccuracy that a run reaching a large value would have many more additive transitions (cf. Claim 12). The out-tree(w) is drawn in black, other (irrelevant) lines are drawn in light grey. Run π_{suf} is the shortest one that contains all black lines. There are $n = 4$ occurrences of η in w_{suf}, thus π_{suf} is split into 5 parts, and runs $\pi_1, \ldots, \pi_4$ end with a transition labelled by η.

We first show that unboundedness guarantees the existence of a sustainable part π_c of a witness. Define $m \coloneqq \max\{i \mid v_i < M\}$.

▷ Claim 31. For every $i > m$ and every transition $s \xrightarrow{\nu} t$ in $\mathcal{A}$ such that t is a state visited by π_i and $\nu(y) = 0$, there exists a sustainable run π_c from t to q_{fin} of length at most $3N^2$.

We prove the claim by downward induction on i, the index of the segment π_i incident with the state t of ν. Base case ($i = n+1$) is trivial, as $w_{n+1} \in A_X^*$. Assume our claim holds for $i+1$. Fix a transition $s \xrightarrow{\nu} t$ such that t is a state visited by π_i, and $\nu(y) = 0$. Similarly to the case of a trivial boundedness witness, a steep increase on y implies existence of a run $\pi_c' = t \xrightarrow{\alpha}\!\!\!\!\rightarrow\, r \xrightarrow{\theta, w_\theta}\!\!\!\!\rightarrow\, r \xrightarrow{\beta, w_\beta}\!\!\!\!\rightarrow\, s \xrightarrow{\eta} q_i$ such that $w_\theta, w_\beta \in A_X^*$, α, β are simple paths and θ a simple cycle, and that $\text{eff}(\theta)(y) > 0$. Finally, by applying the inductive hypothesis to $s \xrightarrow{\eta} q_i$, we get a pumpable π_c'' from q_i to q_{fin}; we have thus constructed a sustainable run $\pi_c'\pi_c''$, as required.

It remains to show how to find π_b, the pumpable part of the run. We consider two cases.

Case 1: $v_{i+1} - v_i > 4CN$ for some $i \in \mathbb{N} \cap [m, n]$. (aim: pumpable run of type A) Fix such $i \in \mathbb{N}$. Let $\mathcal{A}' = (S, Q \cup \{q_{i+1}'\}, q_i, \{q_{i+1}'\}, \delta')$, where δ' is constructed from δ by removing transitions with minimum substitutions, and adding $q \xrightarrow{\eta} q_{i+1}'$ whenever $q \xrightarrow{\eta} q_{i+1}$ for some $q \in Q$. Note that $w_i \in \mathcal{L}(\mathcal{A}')$ and that $\text{eval}_{\mathbf{0}}(w_i)(x) > 4CN$, thus automaton $\mathcal{A}'$ satisfies the premises of Lemma 16. Using this lemma, we obtain a short pumpable run π_b' of type A of $\mathcal{A}'$ – cases A.1, A.2 and A.3 were designed to match all possible loop arrangements. It naturally induces π_b in $\mathcal{A}$ from q_i to q_{i+1} of the same properties. Since q_i is reachable from q_{ini}, there exists a short run π_a between them. Finally, by Claim 31, we obtain a sustainable part π_c of the witness, which completes the analysis of this case.

Case 2: $v_{i+1} - v_i \leq 4CN$ for all $i \in \mathbb{N} \cap [m, n]$. (aim: witness exists or contradiction) This case proves to be more difficult, as it involves a more complicated type B pumpable run. The proof is by contradiction. Here, since $v_{n+1} - v_m > (M + W) - M = 12CN^2$

and each v_{i+1} provides an increase of at most $4CN$ compared to the previous value v_i, any maximal increasing subsequence of $(v_i)_{m\leq i\leq n}$ must have at least $\frac{12CN^2}{4CN} = 3N$ elements. Therefore, there exist $k, \ell \in \mathbb{N}$ such that

$$m \leq k < \ell \leq n, \quad v_k < v_\ell, \quad M < v_i < M + W \quad \text{for} \quad k \leq i \leq \ell, \quad \text{and} \quad q_k = q_\ell$$

thus $\pi_{(k\ell]} \coloneqq \pi_{k+1} \cdots \pi_l$ is a cycle. For each i, $k \leq i \leq l$, define $c_i \in \mathbb{K}$ to be the effect on x of the run π_i without its last transition labelled with η. We have $v_{i+1} \leq v_i + c_i$, and therefore $\sum_{i=k}^{\ell-1} c_i > v_\ell - v_k > 0$. Assume that $\mathcal{A}$ has no pumpable run of type B from q_k to q_ℓ (otherwise we obtain a witness easily). Therefore, there exists $i \in \mathbb{N} \cap [k, l]$ such that π_i decomposes into

$$\pi_i \coloneqq q_{i-1} \xrightarrow{\alpha,u}\!\!\!\!\rightarrow r_1 \xrightarrow{\beta,v}\!\!\!\!\rightarrow r_2 \xrightarrow{\eta} q_i,$$

where β is the run of maximal length labelled by some $v \in A_X^*$ without ρ, run α is labelled by $u \in (A_X \cup \{\rho\})^*$ (possibly empty), and no simple cycle θ with $\operatorname{eff}(\theta) \in \mathbb{N} \times \mathbb{N}_+$ is reachable from r_1 and backwards-reachable from r_2. Note that $\operatorname{eval}_{\mathbf{0}}(\pi_{\text{pre}}\pi_1\pi_2 \cdots \pi_{i-1}\alpha)(y) = 0$.

Assume that α contains a simple cycle θ. If $\operatorname{eff}(\theta)(x) \leq 0$, this cycle can be removed from π without decreasing the output value, which contradicts the assumption that π is the shortest. If otherwise $\operatorname{eff}(\theta)(x) > 0$, we obtain a pumpable run of type A.1 featuring θ and a simple path to r_1. Again, a sustainable run to the final state q_{fin} is guaranteed by Claim 31, and thus in this case the proof is finished.

Hence, we can assume that α does not have simple cycles. Due to Claim 12, we have that $\operatorname{eff}(\alpha)(x) \in [-CN, CN]$. By Lemma 15, $\operatorname{eff}(\beta)$ decomposes into $\operatorname{eff}(\beta) = \operatorname{eff}(\beta') + (A, B)$ for a run β' from r_1 to r_2 of length $\leq N$, and $(A, B) \in \operatorname{Cone}_{\mathbb{N}}(\Theta)$, where Θ is the set of simple cycles that are reachable from r_1 and backwards-reachable from r_2 by transitions not involving ρ. By assumption, $\operatorname{eff}(\theta) \notin \mathbb{N} \times \mathbb{N}_+$ for any $\theta \in \Theta$. Thus, for each $\theta \in \Theta$, either $\operatorname{eff}(\theta)(x) < 0$ or $\operatorname{eff}(\theta)(y) \leq 0$. Hence, for θ with $\operatorname{eff}(\theta)(y) > 0$, we have $\operatorname{eff}(\theta)(x) < 0$. Take $\theta_{\circlearrowleft}$ such that $\operatorname{eff}(\theta_{\circlearrowleft}) = (a, b)$, $b > 0$ (and hence $a < 0$) and $\frac{b}{-a}$ is the largest possible among cycles satisfying these conditions. Every simple cycle has length at most N, therefore its effect belongs to $[-CN, CN]^2$. Thus, $\frac{CN}{1} \geq \frac{b}{-a}$. Let $\ell(t) \coloneqq \frac{b}{a}t$. If there exists θ' such that $\operatorname{eff}(\theta')$ lies above line ℓ, then we have identified two cycles that span a cone having a nonempty intersection with the positive quadrant; this yields a pumpable run of type A.3, and, by Claim 31, we get a sustainable run starting at q_i.

Otherwise, $\operatorname{Cone}_{\mathbb{N}}(\Theta)$ lies below ℓ. Since π_i ends with η and has effect at least M, $\operatorname{eff}(\beta)(y) > M$, therefore $B > 0$. This in turn implies $A < 0$, because (A, B) is bellow ℓ. Hence $B < \ell A = \frac{b}{a}A \leq -CNA$. We know that $\operatorname{eval}_{(v_{i-1},0)}(\pi_i) \geq M$, therefore

$$\begin{cases} \operatorname{eval}_{(v_{i-1},0)}(\alpha\beta)(x) \geq M \\ \operatorname{eval}_{(v_{i-1},0)}(\alpha\beta)(y) \geq M \end{cases} \quad \text{and thus} \quad \begin{cases} v_{i-1} + \operatorname{eff}(\alpha)(x) + \operatorname{eff}(\beta')(x) + A \geq M \\ 0 \qquad\qquad\quad + \operatorname{eff}(\beta')(y) + B \geq M \end{cases}$$

Since $v_i < M + W$, and effects of simple runs α and β' are bounded by Claim 12, we get

$$\begin{cases} \cancel{M} + W + 2CN + A \geq \cancel{M} \\ CN + B \geq M \end{cases} \quad \text{and} \quad \begin{cases} 12C^2N^3 + 2C^2N^2 \geq -CNA \\ B \geq 15C^2N^3 - CN \end{cases}$$

Since $B < -CNA$, we have $15C^2N^3 - CN < 12C^2N^3 + 2C^2N^2$ and finally $3C^2N^3 < 2C^2N^2 + CN$ which yields a contradiction that concludes the proof. ◀

5 Output-minimum CRAs

In this section, we consider CRAs in which minimum substitutions can only appear in the output function. We call such CRAs *output-minimum* for brevity.

The main results of this section are as follows.

▶ **Theorem 32.** *Boundedness of output-minimum CRAs with no transpositions is* coNP*-complete, even if the numbers in the substitutions are presented in unary.*

▶ **Theorem 33.** *Boundedness of output-minimum CRAs is* PSPACE*-complete, even if the numbers in the substitutions are presented in unary.*

For the rest of the section, fix the set of registers $X := \{x_1, \dots, x_d\}$. Let $\eta_{X'} := \{x_1 \leftarrow \min\{x \mid x \in X'\}\}$ for $X' \subseteq X$. Once again, we use the formalism of regular languages of substitutions presented in Section 3. Recall that $\mathrm{Sub}_{\mathrm{elem}}(X) = T_X \cup R_X \cup A_X \cup M_X$. Since we are considering output-minimum CRAs, similarly to Claim 13, we can assume that the alphabet contains only elementary substitutions from $T_X \cup R_X \cup A_X$, with the only exception of a minimum transition which comes at the end of the word. Thus, in Section 5.1 and Section 5.2 we consider the language boundedness problem for NFAs $\mathcal{A}$ such that for some $X' \subseteq X$, we have, respectively, $\mathcal{L}(\mathcal{A}) \subseteq (A_X \cup R_X)^* \eta_{X'}$ and $\mathcal{L}(\mathcal{A}) \subseteq (A_X \cup R_X \cup T_X)^* \eta_{X'}$. As shown in Section 3, this is enough to prove Theorems 32 and 33.

5.1 Output-minimum CRAs with no transpositions

▶ **Proposition 34.** *Boundedness for regular subsets of $(A_X \cup R_X)^* \eta_{X'}$ is* coNP*-complete, even if the numbers in the substitutions are presented in unary.*

Proof. As a certificate of unboundedness, we consider substitutions $\alpha_1, \dots, \alpha_{d+1}$, $\beta_1, \dots, \beta_d$ respecting the conditions of Lemma 16, together with a run π of $\mathcal{A}$ witnessing that $\alpha_1 \beta_1^+ \alpha_2 \dots \alpha_d \beta_d^+ \alpha_{d+1} \eta_{X'} \subseteq \mathcal{L}(\mathcal{A})$. Checking the second and third conditions of Lemma 16 is trivial and by having π in the certificate, it is also easy to check the first condition in linear time.

Checking the last conditions requires a bit more work. As argued in the proof of Lemma 16, all substitutions in $\beta_1, \dots, \beta_d$ can be modified so that they become additive substitutions without changing the value of $\mathrm{eval}_{\mathbf{0}}(\alpha_1 \beta_1^{a_1} \alpha_2 \dots \alpha_d \beta_d^{a_d} \alpha_{d+1} \eta_{X'})(x_1)$. Now, let the vectors $\vec{v}_1, \dots, \vec{v}_d$ be the effects of the modified substitutions $\beta_1, \dots, \beta_d$. By Theorem 14, we only need to solve the following linear program

$$\exists a_1, \dots, a_d \in \mathbb{Q}_{\geq 0} \text{ s.t. } a_1 \vec{v}_1 + a_2 \vec{v}_2 + \dots + a_d \vec{v}_d > 0,$$

which can be done in polynomial time.

To prove coNP-hardness, we reduce the satisfiability problem, which is NP-complete [20], to the complement of the boundedness problem.

▶ **Problem 35** (Satisfiability)**.**
Input *A set $C = \{c_1, \dots, c_m\}$ of clauses over boolean variables $p_1, \dots, p_n$.*
Question *Does there exist an assignment of Boolean values to the variables satisfying all the clauses?*

As registers, we take the set of all clauses: $X = C = \{c_1, \dots, c_m\}$. Let $C_p := \{c \in C \mid p \vDash c\}$ and $C_{\neg p} := \{c \in C \mid \neg p \vDash c\}$ be the sets of clauses satisfied by $p = \top$ and $p = \bot$, respectively. Also, for a literal x, let $\mathrm{inc}(x) := \{c \leftarrow c + 1 \mid c \in C_x\}$. Consider the generalised NFA $\mathcal{A}$ in Figure 3.

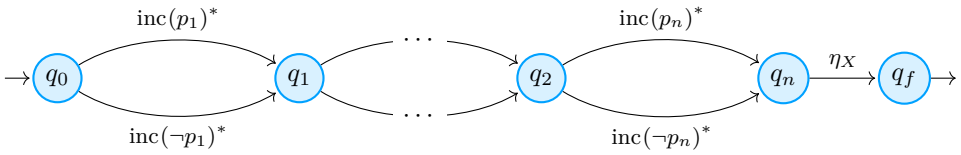


Figure 3 Generalised NFA $\mathcal{A}$ for satisfiability. Transitions are labelled by regular expressions.

It is readily seen that for any $N \in \mathbb{N}$, there exists a word

$$w \in \left(\mathrm{inc}(C_{p_1})^N \cup \mathrm{inc}(C_{\neg p_1})^N\right) \cdots \left(\mathrm{inc}(C_{p_n})^N \cup \mathrm{inc}(C_{\neg p_n})^N\right)\eta_X$$

such that $\mathrm{eval}_{\mathbf{0}}(w)(x_1) \geq N$ if and only if the variable assignment induced by w for variables $p_1, \ldots, p_n$ satisfies all the clauses $c_1, \ldots, c_m$. ◀

5.2 Output-minimum CRAs with transpositions

▶ **Proposition 36.** *Boundedness for regular subsets of* $(A_X \cup R_X \cup T_X)^*\eta_X$ *is in* PSPACE.

Proof idea. We prove containment in PSPACE by operating on an exponentially larger NFA $\mathcal{P}_\mathcal{A}$, called permutation NFA. This NFA encodes all possible register permutations inside its state space, and hence its alphabet contains only additive and reset substitutions. It can be checked in NPSPACE whether this larger NFA admits a certificate of the type presented in Lemma 16. By Savitch's theorem we get that the problem is in PSPACE [38]. ◀

Proof. Fix a finite alphabet $S \subset A_X \cup R_X \cup T_X$ and let $C = \mathrm{maxc}(S)$. Fix NFA $\mathcal{A} = (Q, S \cup \{\eta_{X'}\}, \delta, q_{\mathrm{ini}}, Q_{\mathrm{fin}})$ such that $\mathcal{L}(\mathcal{A}) \subseteq S^*\eta_{X'}$. Let G_X be the set of all permutations of the set X and let $\mathcal{P}_\mathcal{A} = (Q', S' \cup \{\eta_{X'}\}, \delta', q'_{\mathrm{ini}}, Q'_{\mathrm{fin}})$, where $Q' = Q \times G_X$, $S' = S \setminus T_X$, and $Q'_{\mathrm{fin}} = Q_{\mathrm{fin}} \times G_X$. We also take ${q_{\mathrm{ini}}}' = (q_{\mathrm{ini}}, \tau_0)$, where τ_0 is the identity permutation. Let $\mathrm{id} \in S$ be the additive substitution that adds 0 to every register. For $s \in T_X$ and $\tau \in_X$ we define $(\tau \circ s)(x) = s(\tau(x))$. Finally, δ'contains transitions

$$\begin{aligned} ((q,\tau), \mathrm{id}, (q', \tau \circ s)) &\in \delta' \qquad \text{for every } (q,s,q') \in \delta \text{ such that } s \in A_X, \\ ((q,\tau), s, (q', \tau)) &\in \delta' \qquad \text{for every } (q,s,q') \in \delta \text{ such that } s \notin T_X. \end{aligned}$$

Clearly, $\mathcal{A}$ is unbounded if and only if $\mathcal{P}_\mathcal{A}$ is unbounded if and only if there exist words $\alpha_1, \ldots, \alpha_{d+1}, \beta_1, \ldots, \beta_d$ that adhere to the conditions of Lemma 16 and run π of $\mathcal{P}_\mathcal{A}$ witnessing that $\alpha_1\beta_1^+\alpha_2 \ldots \alpha_d\beta_d^+\alpha_{d+1}\eta_{X'} \subseteq \mathcal{L}(\mathcal{P}_\mathcal{A})$. Since $|Q'| = |Q| \cdot d!$, we can store one state with polynomial space. Hence, an NPSPACE algorithm can non-deterministically search for the run π without constructing $\mathcal{P}_\mathcal{A}$ explicitly. Verifying the first three conditions of Lemma 16 can be done on the fly in linear space. Also, a non-deterministic algorithm can guess and verify that $\vec{v}_1 \ldots, \vec{v}_d$ are the effects of cycles $\beta_1, \ldots, \beta_d$ on the fly. Finally, it is easy to verify that a positive linear combination of them has positive effect on all registers. Thus, we can conclude the argument by recalling that NPSPACE = PSPACE by Savitch's theorem [38]. ◀

▶ **Proposition 37.** *Boundedness for regular subsets of* $(A_X \cup R_X \cup T_X)^*\eta_X$ *is* PSPACE*-hard, even if the numbers in the substitutions are presented in unary.*

Proof idea. We reduce the DFA intersection problem, which is PSPACE-complete [27]. For every state of each DFA, we create a separate register in the constructed CRA $\mathcal{C}$. We simulate reading a letter by all the DFAs by moving a large value to the registers corresponding to

the new active states of the DFAs, and keeping the values of all remaining registers zero. These large values come from a self-loop transition in the initial state of $\mathcal{C}$, and $\mathcal{C}$ cannot return to the initial state afterwards. All DFAs accept the same word if and only the large values can be simultaneously brought to the registers corresponding to the final states of the DFAs. The output of $\mathcal{C}$ is thus set to be the minimum of these registers. ◀

Proof. We reduce from the following PSPACE-complete problem [27]:

▶ **Problem 38** (DFA intersection).
Input $n \in \mathbb{N}$, *alphabet* Σ, n *DFAs* $\mathcal{A}_i = (Q_i, \Sigma, \delta_i, q_{\text{ini}}^{(i)}, Q_{\text{fin}}^{(i)})$, $1 \leq i \leq n$.
Question *Does there exist a word accepted by all the DFAs?*

We can assume that each DFA has only one final state, which can be ensured as follows: add a new letter # to Σ and two new states q_i^+, q_i^- to each $\mathcal{A}_i$. For each i, make this new letter # send all states from $Q_{\text{fin}}^{(i)}$ to q_i^+, and all other states of $\mathcal{A}_i$ to q_i^-. Make the new letter induce a self-loop for both q_i^+, q_i^- and make q_i^+ to be the only final state in each DFA.

We construct an NFA $\mathcal{A}$ such that the language $\mathcal{L}(\mathcal{A})$ is bounded if and only if $\bigcap_{1\leq i\leq n} \mathcal{L}(\mathcal{A}_i)$ is empty. Let $X = \left\{q_i^{(j)} \mid 1 \leq j \leq n, q_i \in Q_j\right\}$. The idea is that the registers correspond to the states of the DFAs, and register $r_i^{(j)}$ has a positive value after reading $w \in \Sigma$ if and only if the i'th state in $\mathcal{A}_j$ is active after reading w. Next, we describe $\mathcal{L}(\mathcal{A})$ in terms of a regular language.

Let $\nu_{\text{inc}} = \left\{q_{\text{ini}}^{(i)} \leftarrow q_{\text{ini}}^{(i)} + 1 \mid 1 \leq i \leq n\right\}$ be a substitution that increments the registers representing initial states for each $\mathcal{A}_i$. Also, for every $\sigma \in \Sigma$, let

$$T_{i,\sigma} = \{\{q' \leftarrow q\} \cup \{q \leftarrow 0 \mid q \neq q'\} \mid (q, \sigma, q') \in \delta_i, q \neq q'\} \text{ and}$$
$$T_\sigma = T_{1,\sigma} \cdot T_{2,\sigma} \cdot \cdots \cdot T_{n,\sigma},$$
$$T = \bigcup_{\sigma\in\Sigma} T_\sigma.$$

There is a substitution in $T_{i,\sigma}$ that simulates every transition inside $\mathcal{A}_i$ for letter $\sigma \in \Sigma$. The idea is that after we guess the next letter $\sigma \in \Sigma$, for each $\mathcal{A}_i$ we need to simulate the transition that is executed when reading this letter. If we pick the correct transition, we move our positive value from register q_i to q_i', and resetting all other registers does not change their values. However, if we pick a wrong transition, we reset our positive value and we can never recover. Then, a substitution from T_σ simulates executing a transition labelled by letter σ in all $\mathcal{A}_i$ and a substitution from T simulates choosing a letter $\sigma \in \Sigma$ and executing a transition labelled by σ in all $\mathcal{A}_i$. Finally let $\nu_{\text{out}} = \left\{x \leftarrow \min\{q_{\text{fin}}^{(i)} \mid 1 \leq i \leq n\}\right\}$. We argue that $\mathcal{L}(\mathcal{A}) = \nu_{\text{inc}}^* \cdot T^* \cdot \nu_{\text{out}}$ is unbounded if and only if $\bigcap_{1\leq i\leq n} L(A_i)$ is non-empty.

Consider a word $w = \sigma_1 \dots \sigma_m \in \Sigma^*$ and an integer $N \in \mathbb{N}$. For every $1 \leq i \leq n$ it follows inductively that there exists a word $w_j' \in \text{inc}^{N+1} \cdot T_{\sigma_1} \cdot T_{\sigma_2} \cdot \cdots \cdot T_{\sigma_j}$ such that $\text{eval}_{\mathbf{0}}(w_j')(q^{(i)}) = N + 1$, for $q \in Q_i$ if and only if $q_{\text{ini}}^{(i)} \xrightarrow{\sigma_1 \dots \sigma_j} q^{(i)}$. Thus, there exists a word $w_m' \in \nu_{\text{inc}}^* \cdot T^* \cdot \nu_{\text{out}}$ such that $\text{eval}_{\mathbf{0}}(w_m') = N + 1$ if and only if $\bigcap_{1\leq i\leq n} L(A_i)$ is non-empty. ◀

6 Stateless CRAs

In this section, for every $d \geq 2$ we present a fairly restricted family of unbounded CRAs with d registers such that the length of a shortest run outputting a value $N \in \mathbb{N}$ is lower bounded by $F_{d-1}(N)$, the $(d-1)$st function in the hierarchy of fast-growing functions. These functions

are defined as follows. Let $F_1(n) = 2n$ and for every $k \geq 2$ let $F_k(n) = F_{k-1} \circ \cdots \circ F_{k-1}(1)$, where $\circ$ denotes the composition of functions, and this composition is taken n times. For example, $F_2(n) = 2^n$ and $F_3(n) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} = \text{Tower}(n)$, where exponentiation is taken n times. We construct these CRAs inductively in the following theorem.

▶ **Theorem 39.** *For every $d \geq 2$, there exists a stateless CRA $\mathcal{C}$ with d registers and d transitions such that for every $N \in \mathbb{N}_+$, any run of $\mathcal{C}_d$ that outputs a value of at least N must have length at least $F_{d-1}(N)$.*

Proof. For $d = 2$ and $d = 3$ consider the two CRAs in Figure 4.

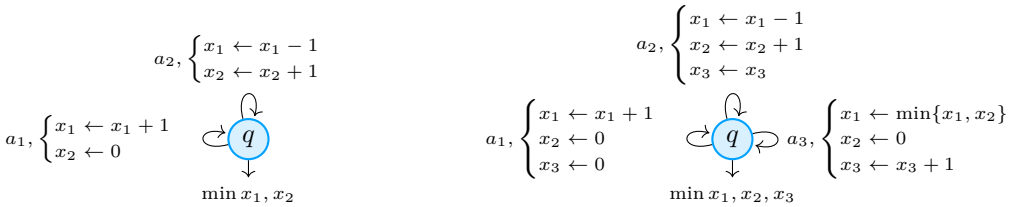


Figure 4 Unbounded CRAs $\mathcal{C}_2$ (left) and $\mathcal{C}_3$ (right) with 2 and 3 registers.

Let us prove that $\mathcal{C}_2$ and $\mathcal{C}_3$ satisfies the statement of the lemma. Let $N \in \mathbb{N}_+$ be an arbitrary positive integer. Since we are dealing with stateless CRAs, we denote a configuration $(q, \{x_1 \leftarrow k_1, x_2 \leftarrow k_2, \ldots, x_n \leftarrow k_n\})$ by a vector $(k_1, k_2, \ldots, k_n)$. Clearly, for both $\mathcal{C}_2$ and $\mathcal{C}_3$, the transitions labelled by a_1 are the only ones that can increase the value of register x_1 and since these transitions reset the values of all other counters, any run outputting N must start by taking this transition m many times, $m > 0$, reaching in $\mathcal{C}_2$ and $\mathcal{C}_3$ the configurations $(m, 0)$ and $(m, 0, 0)$, respectively. The value m can be seen as the initial budget that is necessary for increasing the values of other registers. Clearly, the shortest run that outputs N in $\mathcal{C}_2$ reaches the configuration $(2N, 0)$, then takes N times the transition labelled by a_2 and outputs. Since it must start by getting to the configuration $(2N, 0)$, its length is at most $F_1(N) = 2N$.

Let now π_3 be a shortest run in $\mathcal{C}_3$ that outputs N. Clearly, π_3 needs to increase the value of register x_3. The transition labelled by a_3 is the only one that increases x_3, however, it contains a minimum update for x_1. Since π_3 is a shortest path outputting N, before reading a_3 it reaches a configuration in which the values of registers x_2, x_3 are equal, otherwise some transitions can be removed from it without changing the output value.

Thus, π_3 initialises the budget by reading a_1^m, and then, before reading a_3, it reads a word in a_2^* which applies the function $F_1^{-1}(\cdot) = \frac{\cdot}{2}$ to the value of register x_1. We argue that in order to output N, $m = F_2^{-1}(N) = 2^N$. Indeed, π_3 must reach a value m in register x_1 and then apply N many times the function $F_1^{-1}(\cdot)$ to register x_1, so $m = F_2^{-1}(N)$. Thus, the length of π_3 must be longer than $F_2(N)$. Furthermore, we see that π_3 has the following shape $(0, 0, 0) \twoheadrightarrow (F_2(N), 0, 0) \twoheadrightarrow (N, N, N)$.

Assume now that there exists $\mathcal{C}_{d-1}$ with the property from the statement of the lemma. We modify it by adding a new letter a_d to the alphabet Σ, and extend the substitutions of the transitions as follows:

- add $x_d \leftarrow 0$ to a_1,
- add $x_d \leftarrow x_d$ to a_i for $1 < i < d - 1$, and
- let the substitution of a_d be $\{x_1 \leftarrow \min\{x_1, \ldots, x_{d-1}\}, x_2 \leftarrow 0, \ldots, x_{d-1} \leftarrow 0, x_d \leftarrow x_d + 1\}$.

We know that there exists a shortest path π_{d-1} with the following shape $(0,\dots,0) \twoheadrightarrow (F_{d-2}(N),0,\dots,0) \twoheadrightarrow (N,N,\dots,N,0)$. So, in order to increase the register x_d by one, we need to have enough budget on register x_1 to be able to apply the function F_{d-2}^{-1} to its value. Since we need to increase the value of x_d by one N times, it follows that we need to repeat this process N times so that π_d has shape $(0,\dots,0) \twoheadrightarrow (F_{d-1}(N),0,\dots,0) \twoheadrightarrow (N,N,\dots,N,N)$. Thus its length must be at least $F_{d-1}(N)$. ◀

7 Conclusions and open problems

The most obvious open problem left by this work is the decidability of boundedness for copyless linear CRAs with resets with more than two registers. We conjecture that it is decidable for arbitrary number of registers. Our techniques and the shapes of the witnesses for the two-register case might be useful for proving that.

Another interesting open problem is the precise complexity of the two-register case where numbers in the substitutions are presented in binary. We have proved that this problem is NL-hard and in coNP, but no better bound is known even if minimum substitutions are only allowed in the output. In the latter case, it follows from our results that the witness of unboundedness consists of at most two cycles and some paths connecting them. This relates to the following natural problem whose complexity we were not able to find in the literature. Let $G = (V, E)$ be a digraph, and $\omega : E \to \mathbb{Z}^2$ be a (bi-criteria) weighting function on its edges. Given G and ω, find a cycle in G such that the sum of weights of its edges is component-wise positive. This is of course a generalisation of the problem of finding a cycle of negative weight in a digraph, which can be solved in polynomial time by e.g. Bellman-Ford-Moore algorithm [6].

References

1 Shaull Almagor, Udi Boker, and Orna Kupferman. What's decidable about weighted automata? *Information and Computation*, 282:104651, 2022. doi:10.1016/j.ic.2020.104651.

2 Shaull Almagor, Michaël Cadilhac, Filip Mazowiecki, and Guillermo A. Pérez. Weak cost register automata are still powerful. *International Journal of Foundations of Computer Science*, 31(6):689–709, 2020. doi:10.1142/S0129054120410026.

3 Rajeev Alur, Loris D'Antoni, Jyotirmoy Deshmukh, Mukund Raghothaman, and Yifei Yuan. Regular functions and cost register automata. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2013)*, pages 13–22, 2013. doi:10.1109/LICS.2013.65.

4 Rajeev Alur and Mukund Raghothaman. Decision problems for additive regular functions. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 2013. doi:10.1007/978-3-642-39212-2_7.

5 Benjamin Aminof, Orna Kupferman, and Robby Lampert. Reasoning about online algorithms with weighted automata. *ACM Transactions on Algorithms*, 6(2):28:1–28:36, 2010. doi:10.1145/1721837.1721844.

6 Jørgen Bang-Jensen and Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

7 Michael Blondin, Alain Finkel, Stefan Göller, Christoph Haase, and Pierre McKenzie. Reachability in two-dimensional vector addition systems with states is pspace-complete. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 32–43. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.14.

8 Michael Blondin, Christoph Haase, Filip Mazowiecki, and Mikhail A. Raskin. Affine extensions of integer vector addition systems with states. *Log. Methods Comput. Sci.*, 17(3), 2021. doi:10.46298/LMCS-17(3:1)2021.

9 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Quantitative languages. *ACM Transactions on Computational Logic*, 11(4):23:1–23:38, 2010. doi:10.1145/1805950.1805953.

10 Karel Culík and Jarkko Kari. Digital images and formal languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 599–616. Springer, 1997. doi:10.1007/978-3-642-59126-6_10.

11 Wojciech Czerwiński, Engel Lefaucheux, Filip Mazowiecki, David Purser, and Markus A. Whiteland. The boundedness and zero isolation problems for weighted automata over non-negative rationals. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 15:1–15:13. ACM, 2022. doi:10.1145/3531130.3533336.

12 Laure Daviaud. Containment and equivalence of weighted automata: Probabilistic and max-plus cases. In Alberto Leporati, Carlos Martín-Vide, Dana Shapira, and Claudio Zandron, editors, *Language and Automata Theory and Applications - 14th International Conference, LATA 2020, Milan, Italy, March 4-6, 2020, Proceedings*, volume 12038 of *Lecture Notes in Computer Science*, pages 17–32. Springer, 2020. doi:10.1007/978-3-030-40608-0_2.

13 Laure Daviaud. Register complexity and determinisation of max-plus automata. *ACM SIGLOG News*, 7(2):4–14, 2020. doi:10.1145/3397619.3397621.

14 Laure Daviaud, Pierre-Alain Reynier, and Jean-Marc Talbot. A generalised twinning property for minimisation of cost register automata. In *31st Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS 2016)*, pages 857–866, 2016. doi:10.1145/2933575.2934549.

15 Laure Daviaud and Andrew Ryzhikov. Universality and forall-exactness of cost register automata with few registers. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, August 28 to September 1, 2023, Bordeaux, France*, volume 272 of *LIPIcs*, pages 40:1–40:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.MFCS.2023.40.

16 Manfred Droste and Paul Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. doi:10.1016/J.TCS.2007.02.055.

17 Manfred Droste and Paul Gastin. Weighted automata and weighted logics. In *Handbook of weighted automata*, pages 175–211. Springer, 2009.

18 Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata*. Springer Berlin, Heidelberg, 1st edition, 2009. doi:10.1007/978-3-642-01492-5.

19 Manfred Droste and Dietrich Kuske. Weighted automata. In Jean-Éric Pin, editor, *Handbook of Automata Theory*, pages 113–150. European Mathematical Society Publishing House, Zürich, Switzerland, 2021. doi:10.4171/Automata-1/4.

20 Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.

21 Christoph Haase and Simon Halfon. Integer vector addition systems with states. In Joël Ouaknine, Igor Potapov, and James Worrell, editors, *Reachability Problems - 8th International Workshop, RP 2014, Oxford, UK, September 22-24, 2014. Proceedings*, volume 8762 of *Lecture Notes in Computer Science*, pages 112–124. Springer, 2014. doi:10.1007/978-3-319-11439-2_9.

22 Kosaburo Hashiguchi. New upper bounds to the limitedness of distance automata. *Theor. Comput. Sci.*, 233(1-2):19–32, 2000. doi:10.1016/S0304-3975(97)00260-0.

23 Kosaburo Hashiguchi, Kenichi Ishiguro, and Shuji Jimbo. Decisability of the equivalence problem for finitely ambiguous automata. *International Journal of Algebra and Computation*, 12(03):445–461, 2002. doi:10.1142/S0218196702000845.

24 John E. Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8:135–159, 1979. doi:10.1016/0304-3975(79)90041-0.

25 Daniel Kirsten and Sylvain Lombardy. Deciding Unambiguity and Sequentiality of Polynomially Ambiguous Min-Plus Automata. In *26th International Symposium on Theoretical Aspects of Computer Science (STACS 2009)*, volume 3 of *LIPIcs*, pages 589–600, 2009. doi:10.4230/LIPIcs.STACS.2009.1850.

26 Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoretical Computer Science*, 327(3):349–373, 2004. doi:10.1016/j.tcs.2004.02.049.

27 Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 254–266. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.16.

28 Daniel Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *International Journal of Algebra and Computation*, 4(3):405–426, 1994. doi:10.1142/S0218196794000063.

29 Hing Leung and Viktor Podolskiy. The limitedness problem on distance automata: Hashiguchi's method revisited. *Theor. Comput. Sci.*, 310(1-3):147–158, 2004. doi:10.1016/S0304-3975(03)00377-3.

30 Ernst W. Mayr. An algorithm for the general petri net reachability problem. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing, May 11-13, 1981, Milwaukee, Wisconsin, USA*, pages 238–246. ACM, 1981. doi:10.1145/800076.802477.

31 Filip Mazowiecki and Cristian Riveros. Maximal partition logic: Towards a logical characterization of copyless cost register automata. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, volume 41 of *LIPIcs*, pages 144–159. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPICS.CSL.2015.144.

32 Filip Mazowiecki and Cristian Riveros. Copyless cost-register automata: Structure, expressiveness, and closure properties. *Journal of Computer and System Sciences*, 100:1–29, 2019. doi:10.1016/j.jcss.2018.07.002.

33 Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, USA, 1967.

34 Mehryar Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997. URL: https://aclanthology.org/J97-2003.

35 Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., USA, 1986.

36 Marcel-Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2):245–270, 1961. doi:10.1016/S0019-9958(61)80020-X.

37 Imre Simon. On semigroups of matrices over the tropical semiring. *RAIRO Theor. Informatics Appl.*, 28(3-4):277–294, 1994. doi:10.1051/ITA/1994283-402771.

38 Michael Sipser. *Introduction to the Theory of Computation*. Course Technology, Boston, MA, third edition, 2013.

39 Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *26th Annual Symposium on Foundations of Computer Science (FOCS 1985)*, pages 327–338, 1985. doi:10.1109/SFCS.1985.12.

40 Andreas Weber. Finite-valued distance automata. *Theoretical Computer Science*, 134(1):225–251, 1994. doi:10.1016/0304-3975(94)90287-9.

The Algebras for Automatic Relations

Rémi Morvan
LaBRI, Univ. Bordeaux, CNRS & Bordeaux INP, France

Abstract

We introduce "synchronous algebras", an algebraic structure tailored to recognize automatic relations (*a.k.a.* synchronous relations, or regular relations). They are the equivalent of monoids for regular languages, however they conceptually differ in two points: first, they are typed and second, they are equipped with a dependency relation expressing constraints between elements of different types.

The interest of the proposed definition is that it allows to lift, in an effective way, pseudovarieties of regular languages to that of synchronous relations, and we show how algebraic characterizations of pseudovarieties of regular languages can be lifted to the pseudovarieties of synchronous relations that they induce. Since this construction is effective, this implies that the membership problem is decidable for (infinitely) many natural classes of automatic relations. A typical example of such a pseudovariety is the class of "group relations", defined as the relations recognized by finite-state synchronous permutation automata.

In order to prove this result, we adapt two pillars of algebraic language theory to synchronous algebras: (a) any relation admits a syntactic synchronous algebra recognizing it, and moreover, the relation is synchronous if, and only if, its syntactic algebra is finite and (b) classes of synchronous relations with desirable closure properties (*i.e.* pseudovarieties) correspond to pseudovarieties of synchronous algebras.

2012 ACM Subject Classification Theory of computation → Algebraic language theory

Keywords and phrases synchronous automata, automatic relations, regular relations, transductions, synchronous algebras, Eilenberg correspondence, pseudovarieties, algebraic characterizations

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.21

Related Version *Full Version*: `https://arxiv.org/abs/2404.15496`

Acknowledgements We thank Pablo Barceló, Mikołaj Bojańczyk, and Diego Figueira for helpful discussions, and some anonymous reviewers for valuable feedback.

This pdf contains internal links: clicking on a notion leads to its *definition*.

1 Introduction

1.1 Background

The landscape of rationality for k-ary relations of finite words ($k \geq 2$) is far more complex than for languages – recall that languages can be seen as unary relations of finite words – as depicted in Figure 4 on page 20. Perhaps the most natural class is that of *rational relations*, defined as relations accepted by non-deterministic two-tape automata – an input (u, v) is described by writing u on the first tape and v and the second tape – that can move its two heads independently, from left to right – see [13, §2.1] for a formal definition. For instance, the suffix relation is rational.

Our paper focuses on synchronous relations, *a.k.a.* automatic relations or regular relations, defined as the rational relations that can be recognized by synchronous automata, a subclass of the machines described above obtained by keeping a single head that moves synchronously from left to right, reading one pair of letters after the other; we add padding symbols $\_$ at the end of the shorter word – see Figure 1. While the suffix relation is not synchronous, typical examples include the prefix relation, the same-length relation, etc. Synchronous relations play

a central role in the definitions of automatic structures – introduced by Hodgson [23, 24, 25] and rediscovered by Khoussainov & Nerode [26], see [7, §XI, pp. 627–762]. They also have been studied in the context of graph databases [5, Definition 3.1, p.7 & Theorem 6.3, p. 13], see [18, §8, p. 17] for more context & results on *extended* conjunctive regular path queries.

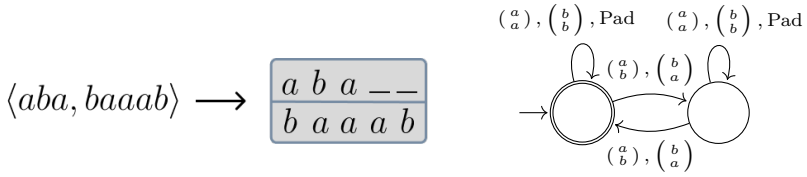


Figure 1 Encoding a pair of words of $\Sigma^* \times \Sigma^*$ into an element of $(\Sigma^2_\_)^*$ where $\Sigma^2_\_ \triangleq (\Sigma \times \Sigma) \cup (\Sigma \times \{\_\}) \cup (\{\_\} \times \Sigma)$ (left) and a deterministic complete synchronous automaton (right) over $\Sigma = \{a, b\}$ accepting the binary relation of pairs (u, v) such that the number of a's in $u_1 \dots u_k$ and in $v_1 \dots v_k$ are the same mod 2, where $k = \min(|u|, |v|)$. Pad denotes the set of transitions $\{\binom{a}{\_}, \binom{b}{\_}, \binom{\_}{a}, \binom{\_}{b}\}$.

▶ Remark 1.1. All our results are described for binary relations, but can be extended to k-ary synchronous relations, see Section 5.

Synchronous relations stand at the frontier between expressiveness and undecidability: for instance, Carton, Choffrut and Grigorieff showed that it is decidable whether an automatic relation is *recognizable* [13, Proposition 3.9, p. 265], meaning that it can be written as a finite union of Cartesian products of regular languages.[1][2] Synchronous relations are effectively closed under Boolean operations – see *e.g.* [7, Lemma XI.1.3, p. 627], and moreover, inclusion (and subsequent problems: universality, emptiness, equivalence...) is decidable for them, by reduction to classical automata, contrary to the equivalence problem over rational relations which is undecidable [6, Theorem 8.4, p. 81].

However, some seemingly easy problems are undecidable: Köcher showed that it is undecidable if the (infinite) graph defined by a synchronous relation is 2-colourable – [28, Proposition 6.5, p. 43], and Barceló, Figueira and Morvan showed that undecidability also holds for regular 2-colourability [3, Theorem 4.4, p. 8]. On the other hand, one can decide if said graph contains an infinite clique, see [27, Corollary 5.5, p. 32]: this is a consequence of [35, Theorem 3.20, p. 185].

1.2 Motivation

Any synchronous relation can be seen as a regular language over the alphabet $\Sigma^2_\_ \triangleq (\Sigma \times \Sigma) \cup (\Sigma \times \{\_\}) \cup (\{\_\} \times \Sigma)$ of pairs. On the other hand any regular language L over $\Sigma^2_\_$ produces a synchronous relation when intersected with the language of all well-formed words – namely words where the padding symbols are consistently placed; see Section 2 for precise definitions. In fact, the semantics of synchronous automata such as the one in Figure 1 is precisely defined this way: it is the intersection of the "classical semantic" of the automaton, seen as an NFA, intersected with well-formed words.

[1] For instance, the relation "having the same length modulo 2" is recognizable, since it can be written as $(aa)^* \times (aa)^* \cup a(aa)^* \times a(aa)^*$.

[2] The problem was latter shown to be NL-complete and PSpace-complete depending on whether the input automaton is deterministic or not in [4, Theorem 1, p. 3].

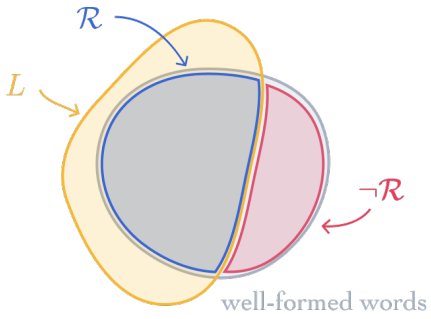


Figure 2 Drawing in $(\Sigma^2_{\shortmid})^*$ of a $\mathcal{V}$-relation $\mathcal{R}$ and $\neg\mathcal{R} \mathrel{\hat{=}} \{(u,v) \in \Sigma^* \times \Sigma^* \mid (u,v) \notin \mathcal{R}\}$, where $\mathcal{R}$ is defined as $L \cap \mathsf{WellFormed}_\Sigma$ with $L \in \mathcal{V}$.

In particular, a class $\mathcal{V}$ of regular languages over $\Sigma^2_{\shortmid}$ (*e.g.* first-order definable languages, group languages, etc.) induces a class of so-called $\mathcal{V}$-relations, defined as the relations over Σ obtained as the intersection of some language of $\mathcal{V}$ with well-formed words, see Figure 2. For instance, the relation of Figure 1 is a $\mathcal{V}$-relation where $\mathcal{V}$ is the class of all group languages – these relations can be alternatively described as those recognized by a deterministic complete synchronous automaton whose transitions functions are permutations of states.

▶ **Question 1.2.** Given a class $\mathcal{V}$ of languages, can we characterize and decide the class of $\mathcal{V}$-relations?

As we will see in Example 2.4, for a relation to be $\mathcal{V}_{\Sigma^2_{\shortmid}}$ is not necessary for it to be a $\mathcal{V}$-relation.

1.3 Contributions

We answer positively to this question. For this we first need to develop an algebraic theory of synchronous relations, which enables us to prove the lifting theorem. In short, the lifting theorem states that algebraic characterizations of classes of word languages can be lifted in a canonical way to algebraic characterizations of classes of word relations.

The algebraic approach usually provides more than decidability: it attaches canonical algebras to languages/relations (*e.g.* monoids for languages of finite words), and often simple ways to characterize complex properties (*e.g.* first-order definability, see *e.g.* [10, Theorem 2.6, p. 40]). Our synchronous algebras differ from monoids in two points:

- they are typed – a quite common feature in algebraic language theory, shared *e.g.* by ω-semigroups [29, §4.1, p. 91];
- they are equipped with a dependency relation, which expresses constraints between elements of different types – to our knowledge, this feature is entirely novel.[3]

Importantly, some variations are possible on the definition of synchronous algebras: in particular, one could get rid of the notion of dependency relation and Lemmas 3.11 and 4.7 would still hold. However, we show in the full version that these simplified synchronous algebras cannot characterize the property of being a $\mathcal{V}$-relation. Therefore, the notion of

[3] Note that algebras equipped with binary relations have been studied before, *e.g.* Pin's ordered ω-semigroups – see [30, §2.4, p. 7] – but the constraints (here the orderings) are always defined between elements of the *same type*.

dependency seems necessary to tackle Question 1.2. Moreover, we show that these algebras arise from a monad, but to our knowledge none of the meta-theorems developing algebraic language theories over monads apply to it, see the full version for more details.

We show that assuming that $\mathcal{V}$ is a $*$-pseudovariety of regular languages – in short, a class of regular languages with desirable closure properties – , then the algebraic characterization of $\mathcal{V}$ can be easily lifted to characterize $\mathcal{V}$-relations.

▶ **Theorem 4.2** (Lifting theorem: Elementary Formulation). *Given a relation $\mathcal{R}$ and a $*$-pseudovariety of regular languages $\mathcal{V}$ corresponding to a pseudovariety of monoids $\mathbb{V}$, the following are equivalent:*

1. *$\mathcal{R}$ is a $\mathcal{V}$-relation,*
2. *$\mathcal{R}$ is recognized by a finite synchronous algebra $\mathbf{A}$ whose underlying monoids are all in $\mathbb{V}$,*
3. *all underlying monoids of the syntactic synchronous algebras $\mathbf{A}_{\mathcal{R}}$ of $\mathcal{R}$ are in $\mathbb{V}$.*

This theorem rests on a solid algebraic theory. First, we show the existence of syntactic algebras (Lemma 3.11): each relation $\mathcal{R}$ admits a unique canonical and minimal algebra $\mathbf{A}_{\mathcal{R}}$, which is finite *iff* the relation is synchronous, and then, we exhibit a correspondence between classes of finite algebras and classes of synchronous relations (Lemma 4.7) – we assume suitable closure properties; these classes are called "pseudovarieties". While the proof structures of Lemmas 3.11 and 4.7 follow the classic proofs, see *e.g.* [31], the dependency relation has to be taken into account quite carefully, leading for instance to a surprising definition of residuals, see Definition 4.5.

Organization. After giving preliminary results in Section 2, we introduce the synchronous algebras in Section 3 and show the existence of syntactic algebras. We then proceed to prove the lifting theorem for $*$-pseudovarieties in Section 4, and after introducing $*$-pseudovarieties of synchronous relations, we provide a more algebraic reformulation of the lifting theorem (Theorem 4.9). We conclude the paper with a short discussion in Section 5.

1.4 Related Work

The algebraic framework has been extended far beyond languages of finite words: let us cite amongst other Reutenauer's "algèbre associative syntactique" for weighted languages [33, Théorème I.2.1, p. 451] and their associated Eilenberg theorem [33, Théorème III.1.1, p. 469]; for languages of ω-words, Wilke's algebras and ω-semigroups, see [29, §II, pp. 75–131 & §VI, pp. 265–306]; more generally, for languages over countable linear orderings, see Carton, Colcombet & Puppis' "$\circledast$-monoids" and "$\circledast$-algebras" [14, §3, p. 7]. A systemic approach has been recently developed using monads, see the full version. Non-linear structures are also suited to such an approach, see *e.g.* Bojańczyk & Walukiewicz's forest algebras [11, §1.3, p. 4] [10, §5, p. 159], or Engelfriet's hyperedge replacement algebras for graph languages [15, §2.3, p. 100] [9, §6.2, p. 194]. For relations over words (*a.k.a.* transductions), recognizable relations are exactly the ones recognized by monoid morphisms $\Sigma^* \times \Sigma^* \to M$ where M is finite. This can be trivially generalized to show that a relation $\mathcal{R}$ is a finite union of Cartesian products of languages in $\mathcal{V}$ if, and only if, it is recognized by a monoid from $\mathbb{V}$, the pseudovariety of monoids corresponding to $\mathcal{V}$, see the full version. In 2023, Bojańczyk & Nguyễn managed to develop an algebraic structure called "transducer semigroups" for "regular functions" [8, Theorem 3.2, p. 6], an orthogonal class of relations to ours – see Figure 4.

The counterpart of $\mathcal{V}$-relations for rational relations – that we call here $\mathcal{V}$-rational relations – was studied by Filiot, Gauwin & Lhote [20]: they show that if $\mathcal{V}$ has decidable membership, then "$\mathcal{V}$-rational transductions" also have decidable membership [20, Theorem 4.10, p. 26]. "Rational transductions" correspond in Figure 4 to the intersection of functional relations with rational relations: this class is orthogonal to synchronous relations, but is included in the class of "regular functions". A different problem – focussing more on the semantics of the transduction – , called "$\mathcal{V}$-continuity" was studied by Cadilhac, Carton & Paperman [12, Theorem 1.3, p. 3], although it has to be noted that their results only concern a finite number of pseudovarieties.

2 Preliminaries

2.1 Automata & Relations

We assume familiarity with basic algebraic language theory over finite words, see [10, §1, 2, 4, pp. 3–66 & pp. 107–156] for a succinct and monad-driven approach, or [31, §I–XIV, pp. 3–247] for a more detailed presentation of the domain. We also refer to [36] for a presentation on pseudovarieties.[4] More precise pointers are given in the full version.

A *relation* is a subset of $\Sigma^* \times \Sigma^*$, where Σ is an alphabet – *i.e.* a non-empty finite set. We define its *complement* $\neg\mathcal{R}$ as the relation $\{(u,v) \in \Sigma^* \times \Sigma^* \mid (u,v) \notin \mathcal{R}\}$. Letting $\Sigma^2_{\_} \triangleq (\Sigma \times \Sigma) \cup (\Sigma \times \{\_\}) \cup (\{\_\} \times \Sigma)$, a *synchronous automaton* is a finite-state machine with initial states, final states, and non-deterministic transitions labelled by elements of $\Sigma^2_{\_}$. We denote by $\mathsf{WellFormed}_\Sigma$ the set of *well-formed* words over $\Sigma^2_{\_}$ where the padding symbols are placed consistently, namely: if some padding symbol occurs on a tape/component, then the following symbols of this tape/component must all be padding symbols. From this constraint, and since $\binom{\_}{\_} \notin \Sigma^2_{\_}$, there can never be padding symbols on both tapes.

Note that elements of $\mathsf{WellFormed}_\Sigma$ are in natural bijection with $\Sigma^* \times \Sigma^*$ – see Figure 1. The relation recognized by a synchronous automaton is the set of pairs $(u,v) \in \Sigma^* \times \Sigma^*$ such that their corresponding element in $\mathsf{WellFormed}_\Sigma$ is the label of an accepting run of the automaton. We say that a relation is *synchronous* if it is recognized by such a machine.

▶ Remark 2.1. Crucially, in the semantics of synchronous automata we *never* try to feed them inputs where the padding symbols are not consistent: for instance, while

$$\binom{aab}{b\_a}, \text{ or } \binom{aba\_}{a\_\_b}$$

are sequences in $(\Sigma^2_{\_})^*$, the behaviour of a synchronous automaton on such sequences is completely disregarded to define the relation it recognizes.

We can then reformulate the definition of the semantics of a synchronous automaton, to make the connection with $\mathcal{V}$-relations – see the next subsection – explicit.

▶ **Fact 2.2.** *Given a synchronous automaton, its semantics as a synchronous automaton can be written as the intersection of its semantics as a classical automaton over $\Sigma^2_{\_}$ with $\mathsf{WellFormed}_\Sigma$.*

In particular a relation $\mathcal{R}$ is synchronous if, and only if, it is a regular language when seen as a subset of $(\Sigma^2_{\_})^*$.

[4] "Pseudovarieties of *foo*" and "varieties of finite *foo*" – where *foo* is *e.g.* "groups" or "semigroups" – are used interchangeably in the literature.

2.2 Induced Relations

Given a class $\mathcal{V}$ of regular languages, the class of *$\mathcal{V}$-relations* over Σ consists of all relations of the form $L \cap \mathsf{WellFormed}_\Sigma$ for some $L \in \mathcal{V}_{\Sigma^2_\_}$ – see Figure 2.[5]

For instance, if $\mathcal{V}$ is the class of all regular languages, then by Fact 2.2, $\mathcal{V}$-relations are exactly the regular relations, *a.k.a.* synchronous relations! However, because of Remark 2.1, the minimal automaton for a relation, seen as a language over $\Sigma^2_\_$, can be significantly more complex than a deterministic complete synchronous automaton recognizing it, see Figure 3 in page 19 – while the size blow-up is only polynomial, it breaks many of the structural properties of the automaton, such as the property of being a permutation automaton.

Note that if $\mathcal{R}$ belongs to $\mathcal{V}$ when $\mathcal{R}$ is seen as a language over $\Sigma^2_\_$, then $\mathcal{R}$ is a $\mathcal{V}$-relation. The converse implication holds under some strong assumption on $\mathcal{V}$ (Fact 2.3), but is not true in general (Example 2.4).

▶ **Fact 2.3.** *If $\mathcal{V}$ is a class of languages closed under intersection and that contains $\mathsf{WellFormed}_\Sigma$, then a relation $\mathcal{R}$ is a $\mathcal{V}$-relation if, and only if, it belongs to $\mathcal{V}$ when seen as a language over $\Sigma^2_\_$.*

Classes of languages $\mathcal{V}$ satisfying the previous assumption (*e.g.* first-order definable languages, piecewise-testable languages, etc.) are easy to capture when it comes to $\mathcal{V}$-relations since this class reduces to $\mathcal{V}$-languages. So, in the remaining of the paper, we will focus on classes $\mathcal{V}$ which do not satisfy the assumptions of Fact 2.3, such as group languages.

▶ **Example 2.4** (Group relations)**.** If $\mathcal{V}$ is the class of group languages, namely languages recognized by permutation automata[6] or equivalently by a finite group, then we call $\mathcal{V}$-relations "*group relations*". They can be characterized as relations recognized by permutation synchronous automata. For instance, the relation of Figure 1 is a group relation as witnessed by the permutation synchronous automaton of Figure 1. Note however that it is not a group language, when seen as a language over $\Sigma^2_\_$, since its minimal automaton over $\Sigma^2_\_$ is not a permutation automaton, see Figure 3 on Page 19.

▶ **Fact 2.5.** *Given a relation $\mathcal{R}$ and a class $\mathcal{V}$ of languages, the following are equivalent:*

1. *$\mathcal{R}$ is a $\mathcal{V}$-relation;*
2. *$\mathcal{R}$ and $\neg\mathcal{R}$ are $\mathcal{V}$-separable as languages over $\Sigma^2_\_$, i.e. there is a language in $\mathcal{V}$ which contains $\mathcal{R}$ and does not intersect $\neg\mathcal{R}$.*

Proof. By definition, see Figure 2, on page 3. ◀

And so, if the $\mathcal{V}$-separability problem is decidable, then the class of $\mathcal{V}$-relations is decidable. However, there are pseudovarieties $\mathcal{V}$ with decidable membership but undecidable separability problem [34, Corollary 1.6, p. 478].[7] Moreover, some of these classes do not contain $\mathsf{WellFormed}_\Sigma$ [34, Corollary 1.7, p. 478]. But beyond this, even when a separation algorithm exists, it can be conceptually much harder than its membership counterpart: for

[5] The notation $L \in \mathcal{V}_{\Sigma^2_\_}$ means that L is a language over the alphabet $\Sigma^2_\_$. See [31, introduction of §XIII.1] for why classes of regular languages are defined in such a way.

[6] A permutation automaton is a finite-state deterministic complete automaton whose transition functions are all permutations of states.

[7] The paper cited only claims undecidability of pointlikes, but it was noted in [21, §1, pp. 1–2] that undecidability of the 2-pointlikes also holds, which is a problem equivalent to separability by [1, Proposition 3.4, p. 6].

instance, deciding membership for group languages is trivial – it boils down to checking if a monoid is a group – , yet the decidability of the separation problem for group languages is considered to be one of the major results in semigroup theory: it follows from Ash's infamous type II theorem [2, Theorem 2.1, p. 129], see [22, Theorem 1.1, p. 3] for a presentation of the result in terms of pointlike sets, see also [32, §III, Theorem 8, p. 5] for an elegant automata-theoretic reformulation.

3 Synchronous Algebras

In this section, we introduce and study the "elementary" properties of synchronous algebras.

3.1 Types & dependent Sets

Motivation. The axiomatization of a semigroup reflects the algebraic structure of finite words: these objects can be concatenated, in an associative way – reflecting the linearity of words. Now observe that elements of $\mathsf{WellFormed}_\Sigma$ are still linear, but not all words can be concatenated together: for instance, $\binom{a}{\_}$ cannot be followed by $\binom{a}{b}$. Formally, given two words $u, v \in \mathsf{WellFormed}_\Sigma$, to decide if $uv \in \mathsf{WellFormed}_\Sigma$ it is necessary and sufficient to know if the last pair of u and first pair of v consists of a pair of proper letters (denoted by $^{\mathrm{L}/\mathrm{L}}$), a pair of a proper letter and a blank/padding symbol ($^{\mathrm{L}/\mathrm{B}}$) or a pair of a blank/padding symbol and a proper letter ($^{\mathrm{B}/\mathrm{L}}$). This information is called the *letter-type* of an element of $\Sigma^2_\_$.

We then define the *type* of a word of $(\Sigma^2_\_)^+$ as the pair (α, β), usually written $\alpha \to \beta$, of the letter-types of its first and last letters. It is then routine to check that the possible types of well-formed words are

$$\mathcal{T} \mathrel{\hat{=}} \left\{ {}^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{L}/\mathrm{L}},\ {}^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{L}/\mathrm{B}},\ {}^{\mathrm{L}/\mathrm{B}} \to {}^{\mathrm{L}/\mathrm{B}},\ {}^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{B}/\mathrm{L}},\ {}^{\mathrm{B}/\mathrm{L}} \to {}^{\mathrm{B}/\mathrm{L}} \right\}.$$

For the sake of readability, we will write α instead of $\alpha \to \alpha$ for $\alpha \in \{{}^{\mathrm{L}/\mathrm{L}}, {}^{\mathrm{L}/\mathrm{B}}, {}^{\mathrm{B}/\mathrm{L}}\}$.

One non-trivial point lies in the following innocuous question: what is the type of the empty word? Any type of $\mathcal{T}$ sounds like an acceptable answer. But then it would be natural to say that the concatenation of $\binom{aaa}{aaa}$ of type $^{\mathrm{L}/\mathrm{L}}$ with the empty word of type $^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{L}/\mathrm{B}}$ should be $\binom{aaa}{aaa}$ of type $^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{L}/\mathrm{B}}$. Automata-wise, this would represent a sequence of transitions $\binom{a}{a}, \binom{a}{a}, \binom{a}{a}$ together with the promise that the next transition would have a padding symbol on its second tape. But then, one has to formalize the idea that the two elements $\binom{aaa}{aaa}$ of type $^{\mathrm{L}/\mathrm{L}}$ and $^{\mathrm{L}/\mathrm{L}} \to {}^{\mathrm{L}/\mathrm{B}}$ represent the same underlying pair of words of $\Sigma^* \times \Sigma^*$: this idea will be captured by what we call a dependency relation. A more natural solution would be to simply introduce a new type for the empty word (or to forbid it), but we show in the full version that the resulting notion of algebras cannot capture the property of being a $\mathcal{V}$-relation.

A *$\mathcal{T}$-typed set* (or *typed set* for short) consists of a tuple $\mathbf{X} = (X_\tau)_{\tau \in \mathcal{T}}$, where each X_τ is a set. Instead of $x \in X_\tau$, we will often write $x_\tau \in \mathbf{X}$. A *map between typed sets* $\mathbf{X}$ and $\mathbf{Y}$ is a collection of functions $X_\tau \to Y_\tau$ for each type τ. Similarly, a subset of $\mathbf{X}$ is a tuple of subsets of X_τ for each type τ. To make the notations less heavy, we will often think of typed sets as sets with type annotations rather than tuples, and ask that all operators/constructions should preserve this type.

▶ **Definition 3.1.** *A dependency relation over a typed set* $\mathbf{X}$ *consists of a reflexive and symmetric relation* $\asymp$ *over* $\biguplus \mathbf{X} \mathrel{\hat{=}} \bigcup_{\tau \in \mathcal{T}} X_\tau \times \{\tau\}$*, such that for all* $x_\sigma, y_\sigma \in \mathbf{X}$*, if* $x_\sigma \asymp y_\sigma$*, then* $x_\sigma = y_\tau$*.*

Crucially, we do not ask for this relation to be transitive – in some examples the dependency relation will be an equivalence relation, but not always (see the full version), and this non-transitivity is actually an important feature, motivated amongst other by the syntactic congruence and Corollary 3.14.

A dependent set is a $\mathcal{T}$*-typed set together with a dependency relation over it. A closed subset of a dependent set* $\langle \mathbf{X}, \asymp \rangle$ *is a subset* $C \subseteq \mathbf{X}$ *such that for all* $x, x' \in \mathbf{X}$*, if* $x \asymp x'$ *then* $x \in C \iff x' \in C$*.*[8]

▶ **Example 3.2.** Given a finite alphabet Σ, let $\mathbf{S}_2\Sigma$ be[9] the dependent set of *synchronous words* defined by:

- $(\mathbf{S}_2\Sigma)_{\mathrm{L/L}} \mathrel{\hat{=}} (\Sigma \times \Sigma)^*$,
- $(\mathbf{S}_2\Sigma)_{\mathrm{L/L} \to \mathrm{L/B}} \mathrel{\hat{=}} (\Sigma \times \Sigma)^*(\Sigma \times \_)^*$,
- $(\mathbf{S}_2\Sigma)_{\mathrm{L/B}} \mathrel{\hat{=}} (\Sigma \times \_)^*$,
- $(\mathbf{S}_2\Sigma)_{\mathrm{L/L} \to \mathrm{B/L}} \mathrel{\hat{=}} (\Sigma \times \Sigma)^*(\_ \times \Sigma)^*$,
- $(\mathbf{S}_2\Sigma)_{\mathrm{B/L}} \mathrel{\hat{=}} (\_ \times \Sigma)^*$.

Moreover, $\asymp$ is the reflexive and symmetric closure of the relation that identifies $u_{\mathrm{L/L}}$ with $u_{\mathrm{L/L} \to \beta}$ for all $u \in (\Sigma \times \Sigma^*)$ and $\beta \in \{\mathrm{L/B}, \mathrm{B/L}\}$, and $u_{\mathrm{L/L} \to \mathrm{L/B}}$ with $u_{\mathrm{L/B}}$ for $u \in (\Sigma \times \_)^*$, and $u_{\mathrm{L/L} \to \mathrm{B/L}}$ with $u_{\mathrm{B/L}}$ for $u \in (\_ \times \Sigma)^*$. This structure is depicted in Figure 5.

Given a relation $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$, we denote by $\underline{\mathcal{R}} = \{(u, v)_\tau \mid (u, v)_\tau \in \mathbf{S}_2\Sigma \text{ and } (u, v) \in \mathcal{R}\}$ the closed subset of $\mathbf{S}_2\Sigma$ induced by $\mathcal{R}$.

▶ **Fact 3.3.** *The map* $\mathcal{R} \mapsto \underline{\mathcal{R}}$ *is a bijection between relations and closed subsets of* $\mathbf{S}_2\Sigma$*.*

Proof. Let f be the function which maps a closed subset C of $\mathbf{S}_2\Sigma$ to $\{(u, v) \in \Sigma^* \times \Sigma^* \mid (u, v)_\tau \in C \text{ for some } \tau \in \mathcal{T}\}$. It then follows that $f \circ \underline{\_}$ (resp. $\underline{f(-)}$) is the identity on subsets of $\Sigma^* \times \Sigma^*$ (resp. closed subsets of $\mathbf{S}_2\Sigma$). ◀

3.2 Synchronous Algebras

One key property of types is that some of them can be concatenated to produce other types. We say that two types $\sigma, \tau \in \mathcal{T}$ are *compatible* when there exists non-empty words $u, v \in \mathsf{WellFormed}_\Sigma$ of type σ and τ, respectively, such that uv is well-formed. Said otherwise, $\alpha \to \beta$ is compatible with $\beta' \to \gamma$ if either $\beta = \beta'$ or $\beta = {}^{\mathrm{L}}/_{\mathrm{L}}$ – indeed, for this last case note that *e.g.* the concatenation of $\left(\begin{smallmatrix} aaa \\ aaa \end{smallmatrix}\right)$ of type ${}^{\mathrm{L}}/_{\mathrm{L}}$ with $\left(\begin{smallmatrix} - \\ a \end{smallmatrix}\right)$ of type ${}^{\mathrm{B}}/_{\mathrm{L}}$ is well-formed. Lastly, if $\alpha \to \beta$ is compatible with $\beta' \to \gamma$, we define their product as $(\alpha \to \beta) \cdot (\beta' \to \gamma) \mathrel{\hat{=}} \alpha \to \gamma$. Note that this partial operation is associative, in the following sense: for $\rho, \sigma, \tau \in \mathcal{T}$, $(\rho \cdot \sigma) \cdot \tau$ is well-defined if and only if $\rho \cdot (\sigma \cdot \tau)$ is well-defined, in which case both types are equal. This implies that the notion of compatibility of types can be unambiguously lifted to finite lists of types $\tau_1, \ldots, \tau_n$.

[8] In other words, C is a union of equivalence classes of the transitive closure of $\asymp$.

[9] The index refers to the arity of the relations we are considering: here we focus on binary relations, but all constructions can be generalized to higher arities.

▶ **Definition 3.4.** *A* synchronous algebra $\langle \mathbf{A}, \cdot, \asymp \rangle$ *consists of a* dependent set $\langle \mathbf{A}, \asymp \rangle$ *together with a partial binary operation* $\cdot$ *on* $\mathbf{A}$*, called* product *such that:*

- *for* $x_\sigma, y_\tau \in \mathbf{A}$*,* $x_\sigma \cdot y_\tau$ *is defined iff* σ *and* τ *are compatible,*
- associativity: *for all* $x_\rho, y_\sigma, z_\tau \in \mathbf{A}$*, if* ρ, σ, τ *are compatible:*

$$(x_\rho \cdot y_\sigma) \cdot z_\tau = x_\rho \cdot (y_\sigma \cdot z_\tau),$$

- "monotonicity": *for all* $x_\sigma, x'_{\sigma'}, y_\tau \in \mathbf{A}$*, if* $x_\sigma \asymp x'_{\sigma'}$ *and both* σ, τ *and* σ', τ *are compatible, then* $x_\sigma \cdot y_\tau \asymp x'_{\sigma'} \cdot y_\tau$*, and dually if* τ, σ *and* τ, σ' *are compatible, then* $y_\tau \cdot x_\sigma \asymp y_\tau \cdot x'_{\sigma'}$*,*
- units: *for each type* τ *there is an element* $1_\tau \in \mathbf{A}$ *such that for any* $x_\sigma \in \mathbf{A}$*, then* $1_\tau \cdot x_\sigma \asymp x_\sigma$ *if* τ *and* σ *are compatible, and* $x_\sigma \cdot 1_\tau \asymp x_\sigma$ *if* σ *and* τ *are compatible, and moreover,* $1_{\mathrm{L}/\mathrm{L} \to \beta} = 1_{\mathrm{L}/\mathrm{L}} \cdot 1_\beta$ *for* $\beta \in \{{}^{\mathrm{L}}/_{\mathrm{B}}, {}^{\mathrm{B}}/_{\mathrm{L}}\}$*.*

Note in particular that for any type $\tau \in \{{}^{\mathrm{L}}/_{\mathrm{L}}, {}^{\mathrm{L}}/_{\mathrm{B}}, {}^{\mathrm{B}}/_{\mathrm{L}}\}$, then $1_\tau \cdot x_\tau \asymp x_\tau$ but since $1_\tau \cdot x_\tau$ has type τ and $\asymp$ is a dependency relation, then $1_\tau \cdot x_\tau = x_\tau$. This implies in particular that restricting $\langle \mathbf{A}, \cdot \rangle$ to a type ${}^{\mathrm{L}}/_{\mathrm{L}}$, ${}^{\mathrm{L}}/_{\mathrm{B}}$ or ${}^{\mathrm{B}}/_{\mathrm{L}}$ yields a monoid. These are called the three *underlying monoids* of $\mathbf{A}$. The canonical example of synchronous algebras is synchronous words $\mathbf{S}_2\Sigma$ under concatenation. Its underlying monoids are $(\Sigma \times \Sigma)^*$, $(\Sigma \times \{\_\})^*$ and $(\{\_\} \times \Sigma)^*$.

▶ **Fact 3.5.** *Any closed subset of* $\mathbf{A}$ *either contains all units, or none of them.*

Proof. From $1_{\mathrm{L}/\mathrm{L} \to \mathrm{L}/\mathrm{B}} = 1_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{B}}$ we have $1_{\mathrm{L}/\mathrm{L}} \asymp 1_{\mathrm{L}/\mathrm{L} \to \mathrm{L}/\mathrm{B}}$ and $1_{\mathrm{L}/\mathrm{L} \to \mathrm{L}/\mathrm{B}} \asymp 1_{\mathrm{L}/\mathrm{B}}$. By symmetry between ${}^{\mathrm{L}}/_{\mathrm{B}}$ and ${}^{\mathrm{B}}/_{\mathrm{L}}$, we also have $1_{\mathrm{L}/\mathrm{L}} \asymp 1_{\mathrm{L}/\mathrm{L} \to \mathrm{B}/\mathrm{L}}$ and $1_{\mathrm{L}/\mathrm{L} \to \mathrm{B}/\mathrm{L}} \asymp 1_{\mathrm{B}/\mathrm{L}}$. Hence, if a closed subset of $\mathbf{A}$ contains at least one unit, then it must contain them all. ◀

Note that the product induces a monoid left (resp. right) action of the underlying monoid $\mathbf{A}_{\mathrm{L}/\mathrm{L}}$ (resp. $\mathbf{A}_{\mathrm{L}/\mathrm{B}}$) on the set $\mathbf{A}_{\mathrm{L}/\mathrm{L} \to \mathrm{L}/\mathrm{B}}$. Moreover, $x_{\mathrm{L}/\mathrm{L}} \mapsto x_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{B}}$ identifies any element of type ${}^{\mathrm{L}}/_{\mathrm{L}}$ with an element of type ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{L}}/_{\mathrm{B}}$. Over $\mathbf{S}_2\Sigma$, these identifications are injective, but it need not be the case in general. Note also that in general, $x_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{L} \to \mathrm{L}/\mathrm{B}} = x_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{B}} = x_{\mathrm{L}/\mathrm{L}} \cdot 1_{\mathrm{L}/\mathrm{B}}$.

▶ Remark 3.6. There exists a monad over the category of dependent sets whose Eilenberg-Moore algebras exactly correspond to synchronous algebras, see the full version.

Morphisms of synchronous algebras are defined naturally as maps that preserve the type, units, the product and the dependency relation.

Free algebras. $\mathbf{S}_2\Sigma$ is free in the sense that for any synchronous algebra $\mathbf{A}$, there is a natural bijection between synchronous algebra morphisms $\mathbf{S}_2\Sigma \to \mathbf{A}$ and maps of typed sets $\Sigma^2_{\_} \to \mathbf{A}$. Said otherwise, synchronous algebra morphisms are uniquely defined by their value on $\Sigma^2_{\_}$.

3.3 Recognizability

Given a synchronous algebra $\mathbf{A}$, a morphism $\varphi\colon \mathbf{S}_2\Sigma \to \mathbf{A}$ and a closed subset $\mathrm{Acc} \subseteq \mathbf{A}$ called "accepting set", we say that $\langle \varphi, \mathbf{A}, \mathrm{Acc} \rangle$ *recognizes* a relation $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$ when $\underline{\mathcal{R}} = \varphi^{-1}[\mathrm{Acc}]$. We extend the notion of recognizability to $\langle \varphi, \mathbf{A} \rangle$ or to simply $\mathbf{A}$ by existential quantification over the missing elements in the tuple $\langle \varphi, \mathbf{A}, \mathrm{Acc} \rangle$.

Synchronous algebra induced by a monoid. A monoid morphism $\varphi\colon (\Sigma^2_{\_})^* \to M$ naturally *induces* a synchronous algebra morphism $\tilde{\varphi}\colon \mathbf{S}_2\Sigma \to \mathbf{A}_M$, where:

- $\mathbf{A}_M$ has for every type τ a copy of M, and $\asymp$ is $\{(x_\sigma, x_\tau) \mid x \in M, \sigma, \tau \in \mathcal{T}\}$,
- for all $x_\sigma, y_\tau \in \mathbf{A}_M$ with compatible type, $x_\sigma \cdot y_\tau \mathrel{\hat{=}} (x \cdot y)_{\sigma\cdot\tau}$,
- $\tilde{\varphi}\left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right) \mathrel{\hat{=}} \left(\varphi\left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right)\right)_{\mathrm{L/L}}$, $\tilde{\varphi}\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right) \mathrel{\hat{=}} \left(\varphi\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)\right)_{\mathrm{L/B}}$, and $\tilde{\varphi}\left(\begin{smallmatrix} \_ \\ a \end{smallmatrix}\right) \mathrel{\hat{=}} \left(\varphi\left(\begin{smallmatrix} \_ \\ a \end{smallmatrix}\right)\right)_{\mathrm{B/L}}$.

The algebra simply duplicates M as many times as needed and identifies two elements together when they originated from the same element of M.

▶ **Fact 3.7.** *If φ recognizes $\mathcal{R}$ for some relation $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$ seen as a language over $\Sigma^2_{\_}$, then $\tilde{\varphi}$ recognizes $\mathcal{R}$.*

Consolidation of a synchronous algebra. Given a synchronous algebra morphism $\varphi\colon \mathbf{S}_2\Sigma \to \mathbf{A}$, define its *consolidation*[10] as the semigroup morphism $\varphi^0\colon (\Sigma^2_{\_})^* \to \mathbf{A}^0$, where $\mathbf{A}^0$ is the monoid obtained from $\biguplus \mathbf{A}$ by first merging units, by adding a zero (denoted by 0), and extending $\cdot$ to be a total function by letting all missing products equal 0, and φ^0 sends a word $u \in (\Sigma^2_{\_})^*$ to

- 0 if u is not well-formed,
- $\varphi(u_{\mathrm{L/L}})$ if $u \in (\Sigma \times \Sigma)^*$,
- $\varphi(u_{\mathrm{L/B}})$ if $u \in (\Sigma \times \_)^+$,
- $\varphi(u_{\mathrm{B/L}})$ if $u \in (\_ \times \Sigma)^+$,
- $\varphi(u_{\mathrm{L/L}\to\mathrm{L/B}})$ if $u \in (\Sigma \times \Sigma)^+(\Sigma \times \_)^+$,
- $\varphi(u_{\mathrm{L/L}\to\mathrm{B/L}})$ if $u \in (\Sigma \times \Sigma)^+(\_ \times \Sigma)^+$.

Note that this operation disregards the dependency relation of $\mathbf{A}$.

▶ **Fact 3.8.** *If φ recognizes some relation $\underline{\mathcal{R}}$, then φ^0 recognizes $\mathcal{R}$, when seen as a language over $\Sigma^2_{\_}$.*

The following result follows from Facts 2.2, 3.7, and 3.8.

▶ **Proposition 3.9.** *A relation is synchronous if and only if it is recognized by a finite synchronous algebra.*

Let us continue with a slightly less trivial example of algebra.

▶ **Example 3.10** (Group relations: Example 2.4, cont'd.)**.** Fix $p, q \in \mathbb{N}_{>0}$. Let $\mathbf{Z}_{p,q}$ denote the algebra whose underlying monoids are:

- the trivial monoid $(0, +)$ for type $^{\mathrm{L}}/_{\mathrm{L}}$,
- the cyclic monoid $(\mathbb{Z}/p\mathbb{Z}, +)$ for type $^{\mathrm{L}}/_{\mathrm{B}}$,
- the cyclic monoid $(\mathbb{Z}/q\mathbb{Z}, +)$ for type $^{\mathrm{B}}/_{\mathrm{L}}$.

Moreover, the sets $Z_{\mathrm{L/L}\to\mathrm{L/B}}$ and $Z_{\mathrm{L/L}\to\mathrm{B/L}}$ are defined as $\mathbb{Z}/p\mathbb{Z}$ and $\mathbb{Z}/q\mathbb{Z}$, respectively. The product is addition – we identify $0_{\mathrm{L/L}}$ with the zero of $\mathbb{Z}/p\mathbb{Z}$ and of $\mathbb{Z}/q\mathbb{Z}$. We denote by $\bar{k}$ the equivalence class of $k \in \mathbb{Z}$ in $\mathbb{Z}/n\mathbb{Z}$ when n is clear from context. The dependency relation identifies (1) all units together and (2) x_σ with $1_\tau \cdot x_\sigma$ and $x_\sigma \cdot 1_\tau$ when the types are compatible.

Let $\varphi\colon \mathbf{S}_2\Sigma \to \mathbf{Z}_{p,q}$ be the synchronous algebra morphism defined by

$$\varphi\left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right) \mathrel{\hat{=}} \bar{0}_{\mathrm{L/L}}, \quad \varphi\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right) \mathrel{\hat{=}} \bar{1}_{\mathrm{L/B}}, \quad \varphi\left(\begin{smallmatrix} \_ \\ a \end{smallmatrix}\right) \mathrel{\hat{=}} \bar{1}_{\mathrm{B/L}} \quad \text{and} \quad \varphi(\varepsilon_\tau) \mathrel{\hat{=}} \bar{0}_\tau \text{ for } \tau \in \mathcal{T}.$$

[10] Named by analogy with Tilson's construction [37, §3, p. 102].

This morphism recognizes any relation of the form

$$\mathcal{R}^{I,J} \mathrel{\hat{=}} \big\{(u,v) \;\big|\; |u| > |v| \text{ and } (|u| - |v| \bmod p) \in I, \text{ or} \\ |u| < |v| \text{ and } (|v| - |u| \bmod q) \in J. \quad \big\},$$

where $I \subseteq \mathbb{Z}/p\mathbb{Z}$ and $J \subseteq \mathbb{Z}/q\mathbb{Z}$ are such that $\bar{0} \notin I$ and $\bar{0} \notin J$. This last condition is necessary because the accepting set has to be a closed subset of $\mathbf{Z}_{p,q}$: if $\bar{0}$ was in I, then we would need $\bar{0} \in J$, but also to add $\bar{0}_{\mathrm{L}/\mathrm{L}}$ to the accepting set: this would recognize

$$\big\{(u,v) \;\big|\; |u| > |v| \text{ and } (|u| - |v| \bmod p) \in I, \text{ or} \\ |u| < |v| \text{ and } (|v| - |u| \bmod q) \in J, \text{ or } |u| = |v|\big\}.$$

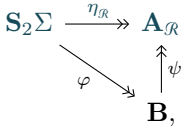
Note also that all relations $\mathcal{R}^{I,J}$ with $\bar{0} \notin I$ and $\bar{0} \notin J$ are group relations: letting G be the group $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$, $\mathcal{R}$ can be written as $\mathsf{WellFormed}_\Sigma \cap \psi^{-1}[I \times \{0\} \cup \{0\} \times J]$ where $\psi\colon (\Sigma^2_{\text{-}})^* \to G$ is the monoid morphism defined by $\psi\left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right) \mathrel{\hat{=}} (\bar{0}, \bar{0})$, $\psi\left(\begin{smallmatrix} a \\ \text{-} \end{smallmatrix}\right) \mathrel{\hat{=}} (\bar{1}, \bar{0})$ and $\psi\left(\begin{smallmatrix} \text{-} \\ a \end{smallmatrix}\right) \mathrel{\hat{=}} (\bar{0}, \bar{1})$.

3.4 Syntactic Morphisms & Algebras

▶ **Lemma 3.11** (Syntactic morphism theorem). *For each relation $\mathcal{R}$, there exists a surjective synchronous algebra morphism*

$$\eta_{\mathcal{R}}\colon \mathbf{S}_2\Sigma \twoheadrightarrow \mathbf{A}_{\mathcal{R}}$$

that recognizes $\mathcal{R}$ and is such that for any other surjective synchronous algebra morphism $\varphi\colon \mathbf{S}_2\Sigma \twoheadrightarrow \mathbf{B}$ recognizing $\mathcal{R}$, there exists a synchronous algebra morphism $\psi\colon \mathbf{B} \twoheadrightarrow \mathbf{A}_{\mathcal{R}}$ such that the diagram



commutes. The objects $\eta_{\mathcal{R}}$ and $\mathbf{A}_{\mathcal{R}}$ are called the syntactic synchronous algebra morphism *and* syntactic synchronous algebra *of $\mathcal{R}$, respectively. Moreover, these objects are unique up to isomorphisms of the algebra.*

▶ **Corollary 3.12** (of Proposition 3.9 and Lemma 3.11). *A relation is synchronous if and only if its syntactic synchronous algebra is finite.*

The proof of Lemma 3.11 – see the full version – relies, as in the case of monoids, on the notion of congruence.

Given a synchronous algebra $\langle \mathbf{A}, \asymp, \cdot \rangle$, a *congruence* is any reflexive, symmetric relation $\approx$ over $\mathbf{A}$ which is coarser than $\asymp$, and which is *locally transitive*, meaning that for all $x_\sigma, x'_\sigma, y_\tau, y'_\tau \in \mathbf{X}$, if $x'_\sigma \approx x_\sigma$, $x_\sigma \approx y_\tau$ and $y_\tau \approx y'_\tau$, then $x'_\sigma \approx y'_\tau$.[11]

The *quotient structure* $\mathbf{A}/\!\approx$ of $\mathbf{A}$ by a congruence $\approx$ is defined as follows:

- its underlying typed set consists of the equivalence classes of $\mathbf{A}$ under the equivalence relation $\{(x_\sigma, y_\sigma) \mid x_\sigma \approx y_\sigma\}$, such a class being abusively denoted by $[x]^{\approx}$,
- its product is the product induced by $\mathbf{A}$, in the sense that $[x]^{\approx} \cdot [y]^{\approx} \mathrel{\hat{=}} [xy]^{\approx}$, and

[11] In particular, it implies that $\approx$ is transitive when restricted to elements of the same type.

- its dependency relation is the relation induced by $\asymp$, *i.e.* $[x]^{\asymp} \preceq [y]^{\asymp}$ whenever $x \asymp y$,
- its units are defined as the equivalence classes of the units of $\mathbf{A}$.

Moreover, $x \mapsto [x]^{\asymp}$ defines a surjective morphism of synchronous algebras from $\mathbf{A}$ to $\mathbf{A}/_{\asymp}$.

Given a synchronous algebra $\langle \mathbf{A}, \preceq, \cdot \rangle$ and a closed subset $C \subseteq \mathbf{A}$, we define a congruence $\asymp_C$, called *syntactic congruence* of C over $\mathbf{A}$ by letting $a_\sigma \asymp_C b_\tau$ when for all $x, y \in \mathbf{A}$

- if both $xa_\sigma y$ and $xb_\tau y$ are defined, then $xa_\sigma y \in C$ *iff* $xb_\tau y \in C$, and
- if both xa_σ and xb_τ are defined, then $xa_\sigma \in C$ *iff* $xb_\tau \in C$, and
- if both $a_\sigma y$ and $b_\tau y$ are defined, then $a_\sigma y \in C$ *iff* $b_\tau y \in C$.

It is routine to check that the syntactic congruence is indeed a congruence. For instance, to prove that $\asymp_C$ is coarser than $\preceq$, observe that if $a_\sigma \preceq b_\tau$, then for all x, y *s.t.* both $xa_\sigma y$ and $xb_\tau y$ are defined, then $xa_\sigma y \preceq xb_\tau y$, and since C is a closed subset of $\mathbf{A}$, $xa_\sigma y \in C$ *iff* $xb_\tau y \in C$. The other two conditions are proven in the same fashion. Note however that while the relation is locally transitive, it is not transitive in general.

When $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$ is a relation, we abuse the notation and write $\asymp_{\mathcal{R}}$ to denote the syntactic congruence $\asymp_{\underline{\mathcal{R}}}$ of $\underline{\mathcal{R}}$ in $\mathbf{S}_2\Sigma$. The existence of the syntactic morphism then follows from the next proposition, proven in the full version.

▶ **Proposition 3.13.** *Let $\varphi \colon \mathbf{S}_2\Sigma \twoheadrightarrow \mathbf{A}$ be a surjective synchronous algebra morphism that recognizes $\mathcal{R}$, say $\underline{\mathcal{R}} = \varphi^{-1}[Acc]$ for some closed subset $Acc \subseteq \mathbf{A}$, then*

$$\begin{array}{rccc} \varphi/_{\asymp_{Acc}} \colon & \mathbf{S}_2\Sigma & \twoheadrightarrow & \mathbf{A}/_{\asymp_{Acc}} \\ & u & \mapsto & [\varphi(u)]^{\asymp_{Acc}} \end{array}$$

is the syntactic morphism of $\mathcal{R}$.

▶ **Corollary 3.14.** *In the syntactic synchronous algebra $\mathbf{A}_{\mathcal{R}}$, the syntactic congruence $\asymp_{Acc}$ and the dependency relation $\preceq$ coincide.*

Proof. By Proposition 3.13 applied to the syntactic morphism, $x \mapsto [x]^{\asymp_{\mathrm{Acc}}}$ is an isomorphism from $\mathbf{A}_{\mathcal{R}}$ to $\mathbf{A}_{\mathcal{R}}/_{\asymp_{\mathrm{Acc}}}$. Hence, $[x]^{\asymp_{\mathrm{Acc}}} \preceq [y]^{\asymp_{\mathrm{Acc}}}$ in $\mathbf{A}_{\mathcal{R}}/_{\asymp_{\mathrm{Acc}}}$ *iff* $x \preceq y$ in $\mathbf{A}_{\mathcal{R}}$, for all $x, y \in \mathbf{A}_{\mathcal{R}}$. But then, the dependency relation $\preceq$ of $\mathbf{A}_{\mathcal{R}}/_{\asymp_{\mathrm{Acc}}}$ is, by definition, such that $[x]^{\asymp_{\mathrm{Acc}}} \preceq [y]^{\asymp_{\mathrm{Acc}}}$ *iff* $x \asymp_{\mathrm{Acc}} y$. Putting both equivalences together, we get that $x \asymp_{\mathrm{Acc}} y$ *iff* $x \preceq y$ for all $x, y \in \mathbf{A}_{\mathcal{R}}$. ◀

We provide in the full version a simple example of syntactic synchronous algebra whose dependency relation is not an equivalence relation.

Boolean operations. Given two synchronous algebras $\mathbf{A}$ and $\mathbf{B}$, define their *Cartesian product* $\mathbf{A} \times \mathbf{B}$ by taking, for each type τ, the Cartesian product $A_\tau \times B_\tau$. Units, product are defined naturally, and the dependency relation is defined by taking the conjunction over each component. Then $\neg\mathcal{R}$ is recognized by $\mathbf{A}$, and $\mathcal{R} \cup \mathcal{S}$ and $\mathcal{R} \cap \mathcal{S}$ are recognized by $\mathbf{A} \times \mathbf{B}$.

4 The Lifting Theorem & Pseudovarieties

4.1 Elementary Formulation

▶ **Example 4.1** (Group relations: Example 3.10 cont'd)**.** We want to decide when the relation

$$\mathcal{R}^{I,J} \triangleq \big\{ (u, v) \;\big|\; |u| > |v| \text{ and } (|u| - |v| \bmod p) \in I, \text{ or} \\ |u| < |v| \text{ and } (|v| - |u| \bmod q) \in J. \quad \big\}$$

from Example 3.10 is a group relation. By definition this happens if and only if there exists a finite group G, together with a monoid morphism $\varphi \colon (\Sigma^2_{\square})^* \to G$ and a subset $\mathrm{Acc} \subseteq G$ *s.t.* $\forall u \in \mathsf{WellFormed}_\Sigma$, $u \in \mathcal{R}^{I,J}$ *iff* $\varphi(u) \in \mathrm{Acc}$. We claim:

$$\mathcal{R}^{I,J} \text{ is a group relation} \quad \textit{iff} \quad \big(\bar{0} \notin I \text{ and } \bar{0} \notin J\big). \tag{$\circledast$}$$

The right-to-left implication was shown in Example 3.10. We prove the implication from left to right: let n be the order of G so that $x^n = 1$ for all $x \in G$. In particular, we have: $\varphi\big(\binom{a}{\_}^{pqn}\big) = 1 = \varphi\big(\binom{a}{a}^{pqn}\big)$. Since $\varphi\big(\binom{a}{a}^{pqn}\big) \notin \mathcal{R}^{I,J}$, it follows that $\binom{a}{\_}^{pqn} \notin \mathcal{R}^{I,J}$ *i.e.* $\bar{0} \notin I$. Also, $\bar{0} \notin J$ by symmetry, which concludes the proof.

Even more generally, we can decide if a relation $\mathcal{R}$ is a group relation by simply looking at the syntactic synchronous algebra of $\mathcal{R}$.

▶ **Theorem 4.2** (Lifting theorem: Elementary Formulation)**.** *Given a relation $\mathcal{R}$ and a $*$-pseudovariety of regular languages $\mathcal{V}$ corresponding to a pseudovariety of monoids $\mathbb{V}$, the following are equivalent:*

1. *$\mathcal{R}$ is a $\mathcal{V}$-relation,*
2. *$\mathcal{R}$ is recognized by a finite synchronous algebra $\mathbf{A}$ whose underlying monoids are all in $\mathbb{V}$,*
3. *all underlying monoids of the syntactic synchronous algebras $\mathbf{A}_{\mathcal{R}}$ of $\mathcal{R}$ are in $\mathbb{V}$.*

See the proof in the full version.

▶ Remark 4.3. In light of Theorem 4.2, one can wonder whether the notion of synchronous algebra is necessary to characterize $\mathcal{V}$-relations, or if it is enough to look at the languages corresponding to the underlying monoids. Said otherwise, is the membership of $\mathcal{R}$ in the class of $\mathcal{V}$-relations uniquely determined by the regular languages $\mathcal{R} \cap (\Sigma \times \Sigma)^*$, $\mathcal{R} \cap (\Sigma \times \{\_\})^*$ and $\mathcal{R} \cap (\{\_\} \times \Sigma)^*$? Unsurprisingly, synchronous algebras are indeed necessary, as there are relations $\mathcal{R}$ such that:

$$\underline{\mathcal{R}} \cap (\Sigma \times \Sigma)^* \in \mathcal{V}_{\Sigma\times\Sigma}, \quad \underline{\mathcal{R}} \cap (\Sigma \times \_)^* \in \mathcal{V}_{\Sigma\times\_} \quad \text{and} \quad \underline{\mathcal{R}} \cap (\_ \times \Sigma)^* \in \mathcal{V}_{\_\times\Sigma}, \tag{$\maltese$}$$

but $\mathcal{R}$ is *not* a $\mathcal{V}$-relation. This can happen even if $\mathcal{V}$ is the $*$-pseudovariety of all regular languages: for instance for the relation

$$\mathcal{R} \mathrel{\hat{=}} \{(u, v) \mid |u| > |v| > 0 \text{ and } |u| - |v| \text{ is prime}\}.$$

Notice that there is a subtle but crucially important difference between ($\maltese$) and the second item of the Lifting Theorem: while the underlying monoids of a synchronous algebra $\mathbf{A}$ recognizing $\mathcal{R}$ only accept words of the form $(\Sigma \times \Sigma)^*$, $(\Sigma \times \_)^*$ or $(\_ \times \Sigma)^*$, elements of $(\Sigma \times \Sigma)^+(\Sigma \times \_)^+$ or $(\Sigma \times \Sigma)^+(\_ \times \Sigma)^+$ influence the underlying monoids of $\mathbf{A}$ via the axioms of synchronous algebras.

Also, note that the existence the Lifting Theorem follows from the careful definition of synchronous algebras: more naive definitions of these algebras simply cannot characterize $\mathcal{V}$-relations, see the full version.

From Theorem 4.2 and the implicit fact that all our constructions are effective, we obtain a decidability (meta-)result for $\mathcal{V}$-relations.

▶ **Corollary 4.4.** *The class of $\mathcal{V}$-relations has decidable membership if, and only if, $\mathcal{V}$ has decidable membership.*

For instance, a relation is a group relation if, and only if, all underlying monoids of its syntactic synchronous algebra are groups.

4.2 Pseudovarieties of Synchronous Relations

We introduce the notion of pseudovariety of synchronous algebras and $*$-pseudovariety of synchronous relations. We show an Eilenberg correspondence between these two notions. We then reformulate the Lifting Theorem to show that any Eilenberg correspondence between monoids and regular languages lifts to an Eilenberg correspondence between synchronous algebras and synchronous relations.

Say that a synchronous algebra $\mathbf{A}$ is a *quotient* of $\mathbf{B}$ when there exists a surjective synchronous algebra morphism from $\mathbf{B}$ to $\mathbf{A}$. A *subalgebra* of $\mathbf{B}$ is any closed subset of $\mathbf{B}$ closed under product and containing the units. We then say that synchronous algebra $\mathbf{A}$ *divides* $\mathbf{B}$ when $\mathbf{A}$ is a quotient of a subalgebra of $\mathbf{B}$.

Observe that $\mathbf{S}_2\Sigma$ admits the following property: elements of type ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{L}}/_{\mathrm{B}}$ and ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{B}}/_{\mathrm{L}}$ are generated by the underlying monoids. Since syntactic synchronous algebras are homomorphic images of $\mathbf{S}_2\Sigma$, they also satisfy this property. In general, we say that a synchronous algebra $\mathbf{A}$ is *locally generated* if every element of type ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{L}}/_{\mathrm{B}}$ (resp. ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{B}}/_{\mathrm{L}}$) can be written as the product of an element of type ${}^{\mathrm{L}}/_{\mathrm{L}}$ with an element of type ${}^{\mathrm{L}}/_{\mathrm{B}}$ (resp. ${}^{\mathrm{B}}/_{\mathrm{L}}$).

A *pseudovariety of synchronous algebras* is any class $\mathbb{V}$ of locally generated finite synchronous algebras closed under

- *finite product:* if $\mathbf{A}, \mathbf{B} \in \mathbb{V}$ then $\mathbf{A} \times \mathbf{B} \in \mathbb{V}$,
- *division:* if some finite locally generated algebra $\mathbf{A}$ divides $\mathbf{B}$ for some $\mathbf{B} \in \mathbb{V}$, then $\mathbf{A} \in \mathbb{V}$.

Because of Lemma 3.11, a synchronous relation is recognized by a finite synchronous algebra of a pseudovariety $\mathbb{V}$ *iff* its syntactic synchronous algebra belongs to $\mathbb{V}$.

A *$*$-pseudovariety of synchronous relations* is a function $\mathcal{V} \colon \Sigma \mapsto \mathcal{V}_\Sigma$ such that for any finite alphabet Σ, $\mathcal{V}_\Sigma$ is a set of synchronous relations over Σ such that $\mathcal{V}$ is closed under

- *Boolean combinations:* if $\mathcal{R}, \mathcal{S} \in \mathcal{V}_\Sigma$, then $\neg\mathcal{R}$, $\mathcal{R} \cup \mathcal{S}$ and $\mathcal{R} \cap \mathcal{S}$ belong to $\mathcal{V}_\Sigma$ too,
- *Syntactic derivatives:* if $\mathcal{R} \in \mathcal{V}_\Sigma$, then any relation recognized by the syntactic synchronous algebra morphism of $\mathcal{R}$ also belongs to $\mathcal{V}_\Sigma$.
- *Inverse morphisms:* if $\varphi \colon \mathbf{S}_2\Gamma \to \mathbf{S}_2\Sigma$ is a synchronous algebra morphism and $\mathcal{R} \in \mathcal{V}_\Sigma$ then $\varphi^{-1}[\mathcal{R}] \in \mathcal{V}_\Gamma$.

To recover a more traditional definition (of the form "closure under Boolean operations, residuals[12] and inverse morphisms"), we need to properly define what are the residuals of a relation. It turns out that the answer is quite surprising and less trivial than what one would expect.

▶ **Definition 4.5** (Residuals). *Let $\mathbf{A}$ be a synchronous algebra, $x_\sigma \in \mathbf{A}$, and $C \subseteq \mathbf{A}$ be a closed subset. The* left residual *and* right residual *of C by x_σ are defined by*

$$x_\sigma^{-1}C \mathrel{\hat{=}} \{y_\tau \in \mathbf{A} \mid \exists y'_{\tau'} \eqsim_C y_\tau,\ x_\sigma y'_{\tau'} \in C\}, \text{ and}$$
$$Cx_\sigma^{-1} \mathrel{\hat{=}} \{y_\tau \in \mathbf{A} \mid \exists y'_{\tau'} \eqsim_C y_\tau,\ y'_{\tau'} x_\sigma \in C\},$$

respectively. We refer indiscriminately to both these notions as residuals. *We extend these notions to sets, by letting $X^{-1}C \mathrel{\hat{=}} \bigcup_{x\in X} x^{-1}C$ and $CX^{-1} \mathrel{\hat{=}} \bigcup_{x\in X} Cx^{-1}$.*

[12] Also called "quotient" *e.g.* in [31, §III.1.3, p. 39], or "polynomial derivative" in [9, §4, p. 19].

For the sake of readability, we will sometimes drop the type of elements when dealing with residuals. It is routine to check that residuals are always closed subsets (since $\eqsim_C$ is coarser than the dependency relation), or that $(x^{-1}C)y^{-1} = x^{-1}(Cy^{-1})$. Equivalently, Cx_σ^{-1} can be defined as the smallest closed subset containing the "naive residual" $\{y_\tau \in \mathbf{A} \mid y_\tau x_\sigma \in C\}$. This latter set is always contained in Cx_σ^{-1} (by reflexivity of $\eqsim_C$), and moreover, if it is empty, then so is Cx_σ^{-1}.

As an example, consider the relation $\mathcal{R}$ from the full version. Then the "naive right residual" of $\underline{\mathcal{R}}$ by $\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)_{\mathrm{L/B}}$ consists of $\varepsilon_{\mathrm{L/L}}$ and all elements of type ${}^{\mathrm{L}}/_{\mathrm{B}}$ and ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{L}}/_{\mathrm{B}}$. But it does not contain any element of type ${}^{\mathrm{B}}/_{\mathrm{L}}$ or ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{B}}/_{\mathrm{L}}$ because such elements cannot be concatenated with $\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)_{\mathrm{L/B}}$ on the right. Yet, the residual $\underline{\mathcal{R}}\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)^{-1}_{\mathrm{L/B}}$ contains all elements of type ${}^{\mathrm{B}}/_{\mathrm{L}}$ (and also ${}^{\mathrm{L}}/_{\mathrm{L}} \to {}^{\mathrm{B}}/_{\mathrm{L}}$): for instance, $\left(\begin{smallmatrix} \_ \\ a \end{smallmatrix}\right)_{\mathrm{B/L}} \in \underline{\mathcal{R}}\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)^{-1}_{\mathrm{L/B}}$ since $\left(\begin{smallmatrix} \_ \\ a \end{smallmatrix}\right)_{\mathrm{B/L}} \eqsim_{\mathcal{R}} \left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)_{\mathrm{L/B}}$ and $\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)_{\mathrm{L/B}}\left(\begin{smallmatrix} a \\ \_ \end{smallmatrix}\right)_{\mathrm{L/B}} \in \mathcal{R}$.

On the other hand, in the algebra $\mathbf{S}_2 a$ consider the relation $\mathcal{S} = (aa)^* \times a(aa)^*$. Then $\underline{\mathcal{S}}\left(\begin{smallmatrix} a \\ a \end{smallmatrix}\right)^{-1}_{\mathrm{L/L}}$ is empty since its "naive residual" $\{y_\tau \in \mathbf{S}_2 a \mid y_\tau \cdot \left(\begin{smallmatrix} a \\ a \end{smallmatrix}\right) \in \mathcal{S}\}$ is empty. Indeed, for $y_\tau \cdot \left(\begin{smallmatrix} a \\ a \end{smallmatrix}\right)_{\mathrm{L/L}}$ to be well-defined, one needs τ to be ${}^{\mathrm{L}}/_{\mathrm{L}}$, *i.e.* y encodes a pair of two words (u, v) of the same length. But then $(ua, va) \notin \mathcal{S}$.

▶ **Lemma 4.6.** *A class $\mathcal{V} \colon \Sigma \mapsto \mathcal{V}_\Sigma$ is a $*$-pseudovariety of synchronous relations if, and only if, it is closed under Boolean combinations, residuals and inverse morphisms.*

See the proof in the full version.

Let $\mathbb{V} \to \mathcal{V}$ denote the map (called *correspondence*) that takes a pseudovariety of synchronous algebras and maps it to

$$\mathcal{V} \colon \Sigma \mapsto \{\mathcal{R} \subseteq \Sigma^* \times \Sigma^* \mid \mathbf{A}_{\mathcal{R}} \in \mathbb{V}\}.$$

Dually, let $\mathcal{V} \to \mathbb{V}$ denote the *correspondence* that takes a $*$-pseudovariety of synchronous relations $\mathcal{V}$ and maps it to the pseudovariety of synchronous algebras generated by all $\mathbf{A}_{\mathcal{R}}$ for some $\mathcal{R} \in \mathcal{V}_\Sigma$. Here, the *pseudovariety generated* by a class C of finite locally generated synchronous algebras is the smallest pseudovariety containing all finite locally generated algebras of C, or equivalently,[13] the class of all finite locally generated synchronous algebras that divide a finite product of algebras of C.[14]

▶ **Lemma 4.7** (An Eilenberg theorem for synchronous relations). *The correspondences $\mathbb{V} \to \mathcal{V}$ and $\mathcal{V} \to \mathbb{V}$ define mutually inverse bijections between pseudovarieties of synchronous algebras and $*$-pseudovarieties of synchronous relations.*

See the proof in the full version.

As consequence of Lemma 4.7, if $\mathcal{V}$ is a $*$-pseudovariety of synchronous relations and $\mathbb{V}$ is a pseudovariety of synchronous algebras, we write $\mathcal{V} \leftrightarrow \mathbb{V}$ to mean that either $\mathcal{V} \to \mathbb{V}$ or, equivalently, $\mathbb{V} \to \mathcal{V}$. This relation is called an *Eilenberg-Schützenberger correspondence*.

[13] The proof is straightforward, see *e.g.* [31, Proposition XI.1.1, p. 190] for a proof in the context of semigroups.

[14] Note that "being locally generated" is not preserved by taking subalgebras, but this is not an issue: we restrict the construction to (finite) locally generated algebras.

▶ **Proposition 4.8.** *If* $\mathbb{V}$ *is a pseudovariety of monoids, then*

$$\mathbb{V}^{sync} \mathrel{\hat{=}} \{\mathbf{A} \text{ locally generated finite synchronous algebra s.t. all underlying monoids of } \mathbf{A} \text{ are in } \mathbb{V}\}$$

is a pseudovariety of synchronous algebras. Similarly, if $\mathcal{V}$ *is an* $*$*-pseudovariety of regular languages, then the class of* $\mathcal{V}$*-relations, namely*

$$\mathcal{V}^{sync} \colon \Sigma \mapsto \{\mathcal{R} \subseteq \Sigma^* \times \Sigma^* \mid \exists L \in \mathcal{V}_{\Sigma^2}\,,\ \underline{\mathcal{R}} = L \cap \mathsf{WellFormed}_\Sigma\},$$

is a $*$*-pseudovariety of synchronous relations.*

Proof. The first point is straightforward. The second one follows from it and Lemma 4.7 and Theorem 4.2. ◀

Finally, Theorem 4.2 can be elegantly rephrased by saying that correspondences between pseudovarieties of monoids and $*$-pseudovarieties of regular languages lift to correspondences between pseudovarieties of synchronous algebras and $*$-pseudovarieties of synchronous relations.

▶ **Theorem 4.9** (Lifting Theorem: Pseudovariety Formulation)**.** *If, in the Eilenberg correspondence between pseudovarieties of monoids and* $*$*-pseudovarieties of regular languages we have* $\mathcal{V} \leftrightarrow \mathbb{V}$*, then in the Eilenberg correspondence between the pseudovariety of synchronous algebras* $\mathbb{V}^{sync}$ *and the* $*$*-pseudovariety of synchronous relations* $\mathcal{V}^{sync}$*, we have* $\mathcal{V}^{sync} \leftrightarrow \mathbb{V}^{sync}$*.*

5 Discussion

A natural next step is to generalize Question 1.2 by replacing $\mathsf{WellFormed}_\Sigma$ by a fixed regular language Ω.

▶ **Question 5.1.** Given a class of regular languages $\mathcal{V}$, can we characterize (and decide) all languages of the form $L \cap \Omega$ for some $L \in \mathcal{V}$?

We claim that the construction of synchronous algebras can be generalized for any Ω, giving rise to the notion of "path algebras". The lifting theorem for monoids can be shown to hold for some Ω, including well-formed words for n-ary relations with $n \geq 3$, and that it cannot effectively hold for all Ω.

A natural next step would then be to study the relationship between "path algebras" and Figueira & Libkin's L-controlled relations [19, §3], see also [16].

Lastly, it would be interesting to extend the results on algebras to automata: for instance, can we adapt our proof to show the existence of a minimal *synchronous* automaton for each relation?

References

1 Jorge Almeida. Some algorithmic problems for pseudovarieties. *Publ. Math. Debrecen*, 52(1):531–552, 1999. Consulted version: `https://www.researchgate.net/profile/Jorge-Almeida-14/publication/2510507_Some_Algorithmic_Problems_for_Pseudovarieties/links/02e7e531d968b4fe8f000000/Some-Algorithmic-Problems-for-Pseudovarieties.pdf`.

2 C. J. Ash. Inevitable graphs: a proof of the type II conjecture and some related decision procedures. *International Journal of Algebra and Computation*, 01(01):127–146, March 1991. `doi:10.1142/S0218196791000079`.

3 Pablo Barceló, Diego Figueira, and Rémi Morvan. Separating Automatic Relations. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023)*, volume 272 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. Consulted version: `https://arxiv.org/abs/2305.08727v2`. `doi:10.4230/LIPIcs.MFCS.2023.17`.

4 Pablo Barceló, Chih-Duo Hong, Xuan-Bach Le, Anthony W. Lin, and Reino Niskanen. Monadic Decomposability of Regular Relations. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 103:1–103:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. Consulted version: `https://arxiv.org/abs/1903.00728v1`. `doi:10.4230/LIPIcs.ICALP.2019.103`.

5 Pablo Barceló, Leonid Libkin, Anthony W. Lin, and Peter T. Wood. Expressive languages for path queries over graph-structured data. *ACM Trans. Database Syst.*, 37(4), December 2012. Consulted version: `https://homepages.inf.ed.ac.uk/libkin/papers/pods10-tods.pdf` (saved on `http://web.archive.org`). `doi:10.1145/2389241.2389250`.

6 Jean Berstel. *Transductions and Context-Free Languages.* Vieweg+Teubner Verlag, Wiesbaden, 1979. Consulted version: `http://www-igm.univ-mlv.fr/~berstel/LivreTransductions/LivreTransductions.pdf` (saved on `http://web.archive.org/`). URL: `http://link.springer.com/10.1007/978-3-663-09367-1`.

7 Achim Blumensath. Monadic Second-Order Model Theory. Version of 2023-12-19 (saved on `http://web.archive.org/`), 2023. URL: `https://www.fi.muni.cz/~blumens/MSO.pdf`.

8 Mikołaj Bojańczyk and Lê Thành Dũng (Tito) Nguyễn. Algebraic Recognition of Regular Functions. 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 117:1–117:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. Consulted version: `https://hal.science/hal-03985883v2`. `doi:10.4230/LIPIcs.ICALP.2023.117`.

9 Mikołaj Bojańczyk. Recognisable Languages over Monads. In Igor Potapov, editor, *Developments in Language Theory*, Lecture Notes in Computer Science, pages 1–13. Springer International Publishing, 2015. Consulted version: `https://arxiv.org/abs/1502.04898v1`. `doi:10.1007/978-3-319-21500-6_1`.

10 Mikołaj Bojańczyk. Languages recognised by finite semigroups, and their generalisations to objects such as trees and graphs, with an emphasis on definability in monadic second-order logic, August 2020. Lecture notes. `arXiv:2008.11635`, `doi:10.48550/arXiv.2008.11635`.

11 Mikołaj Bojańczyk and Igor Walukiewicz. Forest algebras. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, volume 2 of *Texts in Logic and Games*, pages 107–132. Amsterdam University Press, 2008. Consulted version: `https://hal.science/hal-00105796v1`.

12 Michaël Cadilhac, Olivier Carton, and Charles Paperman. Continuity of Functional Transducers: A Profinite Study of Rational Functions. *Logical Methods in Computer Science*, Volume 16, Issue 1, February 2020. `doi:10.23638/LMCS-16(1:24)2020`.

13 Olivier Carton, Christian Choffrut, and Serge Grigorieff. Decision problems among the main subfamilies of rational relations. *RAIRO - Theoretical Informatics and Applications*, 40(2):255–275, April 2006. Consulted version: `http://www.numdam.org/item/10.1051/ita:2006005.pdf`. `doi:10.1051/ita:2006005`.

14 Olivier Carton, Thomas Colcombet, and Gabriele Puppis. An algebraic approach to MSO-definability on countable linear orderings. *The Journal of Symbolic Logic*, 83(3):1147–1189, September 2018. Consulted version: `https://arxiv.org/abs/1702.05342v2`. `doi:10.1017/jsl.2018.7`.

15 Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 2012. Consulted version: https://hal.science/hal-00646514v1. doi:10.1017/CBO9780511977619.

16 María Emilia Descotte, Diego Figueira, and Gabriele Puppis. Resynchronizing Classes of Word Relations. 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 123:1–123:13, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. Consulted version: https://hal.science/hal-01721046v2. doi:10.4230/LIPIcs.ICALP.2018.123.

17 Joost Engelfriet and Hendrik Jan Hoogeboom. Mso definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Logic*, 2(2):216–254, April 2001. Consulted version: SciHub. doi:10.1145/371316.371512.

18 Diego Figueira. Foundations of Graph Path Query Languages (Course Notes). In *Reasoning Web Summer School 2021*, volume 13100 of *Reasoning Web. Declarative Artificial Intelligence - 17th International Summer School 2021, Leuven, Belgium, September 8-15, 2021, Tutorial Lectures*, pages 1–21, Leuven, Belgium, September 2021. Springer. Consulted version: https://hal.science/hal-03349901. doi:10.1007/978-3-030-95481-9_1.

19 Diego Figueira and Leonid Libkin. Synchronizing Relations on Words. *Theory of Computing Systems*, 57(2):287–318, August 2015. Consulted version: https://hal.science/hal-01793633v1/. doi:10.1007/s00224-014-9584-2.

20 Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. Logical and Algebraic Characterizations of Rational Transductions. *Logical Methods in Computer Science*, Volume 15, Issue 4, December 2019. doi:10.23638/LMCS-15(4:16)2019.

21 S. J. v. Gool and B. Steinberg. Pointlike sets for varieties determined by groups. *Advances in Mathematics*, 348:18–50, May 2019. Consulted version: https://arxiv.org/abs/1801.04638v1. doi:10.1016/j.aim.2019.03.020.

22 Karsten Henckell, Stuart W. Margolis, Jean-Éric Pin, and John Rhodes. Ash's type II theorem, profinite topology and Malcev products: part I. *International Journal of Algebra and Computation*, 01(04):411–436, December 1991. Consulted version: https://www.irif.fr/~jep/PDF/HMPR.pdf (saved on http://web.archive.org/). doi:10.1142/S0218196791000298.

23 Bernard R. Hodgson. *Théories décidables par automate fini*. PhD thesis, Université de Montréal, 1976. Not available online.

24 Bernard R. Hodgson. On direct products of automaton decidable theories. *Theoretical Computer Science*, 19(3):331–335, September 1982. doi:10.1016/0304-3975(82)90042-1.

25 Bernard R. Hodgson. Décidabilité par automate fini. *Annales des Sciences Mathématiques du Québec*, 7(1):39–57, 1983. Consulted version: https://www.mat.ulaval.ca/fileadmin/mat/documents/bhodgson/Hodgson_ASMQ_1983.pdf (saved on http://web.archive.org/).

26 Bakhadyr Khoussainov and Anil Nerode. Automatic presentations of structures. In Gerhard Goos, Juris Hartmanis, Jan Leeuwen, and Daniel Leivant, editors, *Logic and Computational Complexity*, volume 960, pages 367–392. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995. doi:10.1007/3-540-60178-3_93.

27 Dietrich Kuske and Markus Lohrey. Some natural decision problems in automatic graphs. *The Journal of Symbolic Logic*, 75(2):678–710, June 2010. Consulted version: https://www.eti.uni-siegen.de/ti/veroeffentlichungen/08-euler-hamilton.pdf (saved on http://web.archive.org/). doi:10.2178/jsl/1268917499.

28 Chris Köcher. *Analyse der Entscheidbarkeit diverser Probleme in automatischen Graphen*. PhD thesis, Technische Universität Ilmenau, Ilmenau, 2014. (Saved on http://web.archive.org/). URL: https://people.mpi-sws.org/~ckoecher/files/theses/bsc-thesis.pdf.

29 Dominique Perrin and Jean-Éric Pin. *Infinite Words, Automata, Semigroups, Logic and Games*, volume 141. Elsevier, 2004. Consulted version: Libgen.

30 Jean-Éric Pin. Positive varieties and infinite words. In Cláudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN'98: Theoretical Informatics*, Lecture Notes in Computer Science, pages 76–87, Berlin, Heidelberg, 1998. Springer. Consulted version: `https://hal.science/hal-00113768v1`. `doi:10.1007/BFb0054312`.

31 Jean-Éric Pin. Mathematical Foundations of Automata Theory, 2022. Version of February 18, 2022 (saved on `http://web.archive.org/`); MPRI lecture notes. URL: `https://www.irif.fr/~jep/PDF/MPRI/MPRI.pdf`.

32 Thomas Place and Marc Zeitoun. Group separation strikes back. In *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, 2023. Consulted version: `https://arxiv.org/abs/2205.01632v2`. `doi:10.1109/LICS56636.2023.10175683`.

33 Christophe Reutenauer. Séries formelles et algèbres syntactiques. *Journal of Algebra*, 66(2):448–483, October 1980. `doi:10.1016/0021-8693(80)90097-6`.

34 John Rhodes and Benjamin Steinberg. Pointlike sets, hyperdecidability and the identity problem for finite semigroups. *International Journal of Algebra and Computation*, November 2011. Consulted version: SciHub. `doi:10.1142/S021819679900028X`.

35 Sasha Rubin. Automata presenting structures: A survey of the finite string case. *Bulletin of Symbolic Logic*, 14(2):169–209, 2008. Consulted version: SciHub. `doi:10.2178/bsl/1208442827`.

36 Howard Straubing and Pascal Weil. Varieties. In Jean Éric Pin, editor, *Handbook of Automata Theory, volume I: Theoretical Foundations*, pages Chapter 16, pp. 569–614. European Mathematical Society Publishing House, September 2021. `doi:10.4171/Automata`.

37 Bret Tilson. Categories as algebra: An essential ingredient in the theory of monoids. *Journal of Pure and Applied Algebra*, 48(1):83–198, 1987. `doi:10.1016/0022-4049(87)90108-3`.

Appendix

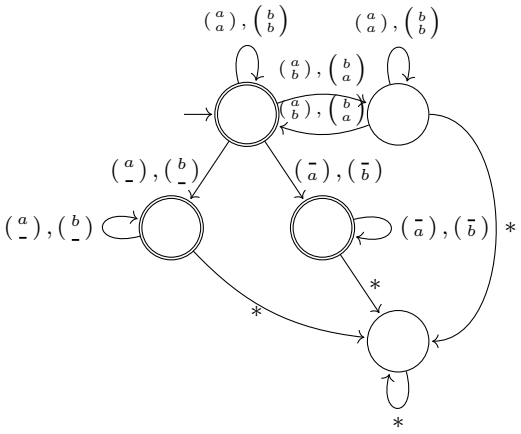


Figure 3 Minimal (deterministic complete) "classical" automaton for the binary relation of pairs (u, v) such that the number of a's in $u_1 \dots u_k$ and in $v_1 \dots v_k$ are the same mod 2, where $k = \min(|u|, |v|)$, seen as a language over $\Sigma^2_{\_}$. Said otherwise, this is automaton rejects exactly all words in $(\Sigma^2_{\_})^*$ which (1) are not the valid encoding of a pair of words and (2) are the encoding of a pair which does not satisfy the property above. Each label $*$ is defined so that the automaton is deterministic and complete.

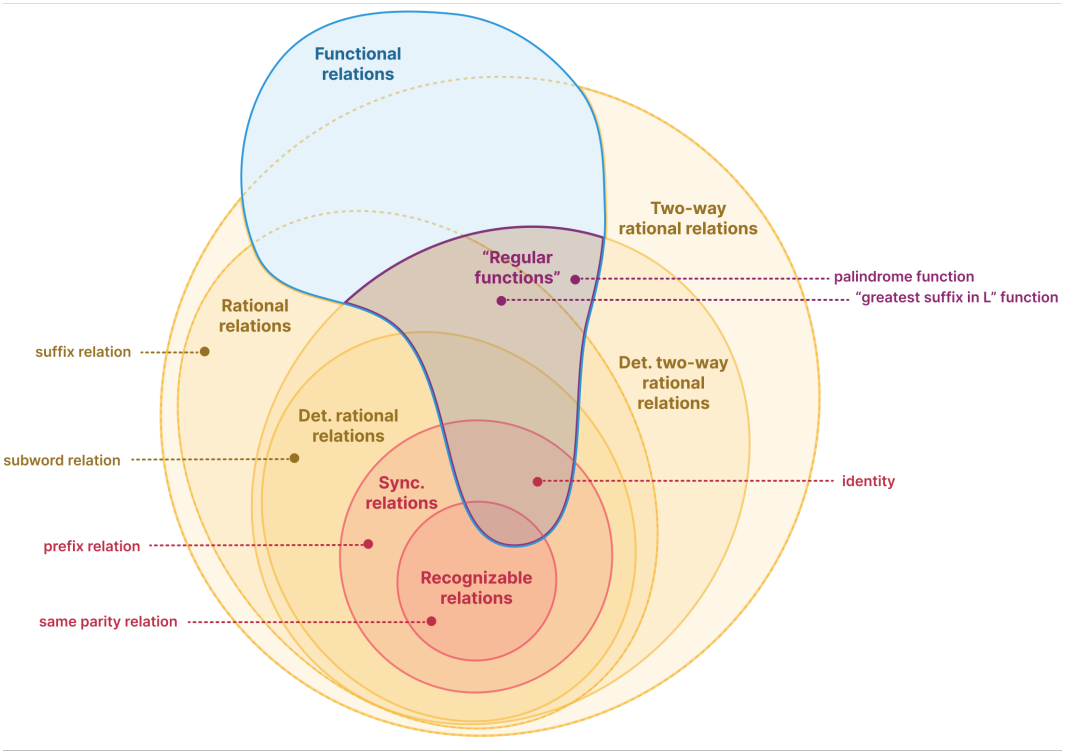


Figure 4 The landscape of rationality for binary relations. Dashed regions are empty: the intersection of functional relations and two-way rational relations collapses to regular functions by [17, Theorem 22, p. 243].

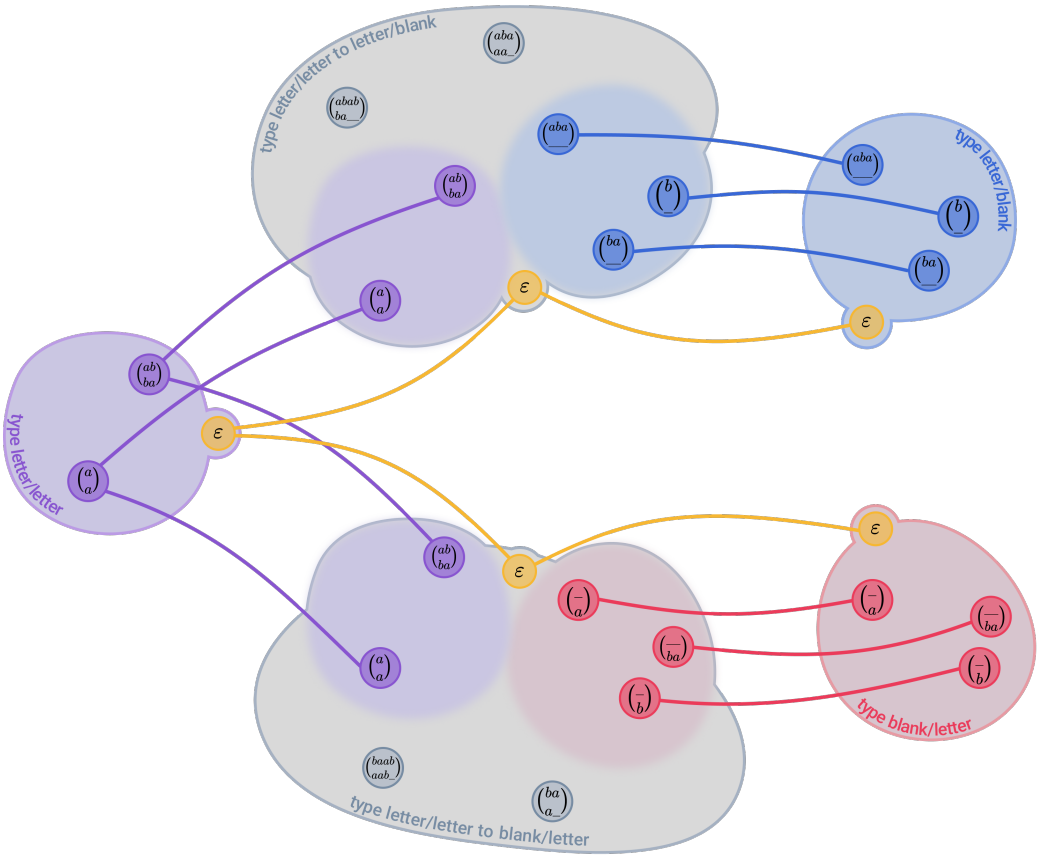


Figure 5 Representation of the dependent set $\mathbf{S}_2\Sigma$ of synchronous words. Coloured edges represent the dependency relation, and self-loops are not drawn.

On the Minimisation of Deterministic and History-Deterministic Generalised (Co)Büchi Automata

Antonio Casares
University of Warsaw, Poland

Olivier Idir
IRIF, Université Paris-Cité, France

Denis Kuperberg
LIP, CNRS, ENS Lyon, France

Corto Mascle
LaBRI, Université de Bordeaux, France
Max Planck Institute for Software Systems (MPI-SWS), Kaiserslautern, Germany

Aditya Prakash
University of Warwick, UK

Abstract

We present a polynomial-time algorithm minimising the number of states of history-deterministic generalised coBüchi automata, building on the work of Abu Radi and Kupferman on coBüchi automata. On the other hand, we establish that the minimisation problem for both deterministic and history-deterministic generalised Büchi automata is NP-complete, as well as the problem of minimising at the same time the number of states and colours of history-deterministic generalised coBüchi automata.

2012 ACM Subject Classification Theory of computation → Verification by model checking

Keywords and phrases Automata minimisation, omega-regular languages, good-for-games automata

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.22

Related Version *Long version with appendices*: `https://arxiv.org/abs/2407.18090`

Funding *Antonio Casares*: Supported by the Polish National Science Centre (NCN) grant "Polynomial finite state computation" (2022/46/A/ST6/00072).

This document contains hyperlinks. On an electronic device, the reader can click on words or symbols (or just hover over them on some PDF readers) to see their definition.

1 Introduction

First introduced by Büchi to obtain the decidability of monadic second order logic over $(\mathbb{N}, \mathsf{succ})$ [14], automata over infinite words (also called ω-automata) have become a well-established area of study in Theoretical Computer Science. Part of its success is due to its applications to model checking (verify whether a system satisfies some given specifications) [5, 42, 24] and synthesis (given a set of specifications, automatically construct a system satisfying them) [13, 35]. In many of these applications, mainly in problems related to synthesis, non-deterministic models of automata are not well-suited, and costly determinisation procedures are usually needed [38].

In 2006, Henzinger and Piterman [26] proposed[1] a model of automata, called *history-deterministic* (HD)[2], presenting a restricted amount of non-determinism so that they exactly satisfy the properties that are needed for applications in synthesis. Namely, these automata do not need to guess the future: an automaton is history-deterministic if it admits a strategy resolving the non-determinism on the fly, in such a way that the run built by the strategy is accepting whenever the input word belongs to the language of the automaton. Since their introduction, several lines of research have focused on questions such as the succinctness of history-deterministic automata [30, 18], the problem of recognising them [4, 8], or extensions to other settings [32, 9, 10].

Minimisation of automata stands as one of the most fundamental problems in automata theory, for various reasons. Firstly, for its applications: when employing algorithms that rely on automata, having the smallest possible ones is crucial for efficiency. Secondly, beneath the problem of minimisation lies a profoundly fundamental question: What is the essential information needed to represent a formal language? A cornerstone result about automata over finite words is that each regular language admits a unique minimal deterministic automaton, in which states corresponds to the residuals of the language (the equivalence classes of the Myhill-Nerode congruence). Moreover, this minimal automaton can be obtained from an equivalent deterministic automaton with n states in time $\mathcal{O}(n \log n)$ [27].

However, the situation is quite different in the case of ω-automata. Contrary to the case of finite words, the residuals of a language are not sufficient to construct a correct deterministic automaton in general. In 2010, Schewe proved that the minimisation of deterministic Büchi automata is NP-complete [39]. That appeared to be a conclusion to the minimisation problem, but a crucial aspect of his proof was that the NP-completeness is established for automata with the acceptance condition *over states*, and this proof does not generalise to *transition-based* automata. A surprising positive result was obtained in 2019 by Abu Radi and Kupferman: we can minimise history-deterministic coBüchi automata using transition-based acceptance in polynomial time [1]. One year later, Schewe showed that the very same problem becomes NP-complete if state-based acceptance is used [40]. Multiple other results have backed the idea that transition-based acceptance is a better-suited model; we refer to [16, Chapter VI] for a detailed discussion. The work of Abu Radi and Kupferman raised the question of what is the complexity of the minimisation problem for other classes of transition-based automata such as (history-)deterministic Büchi automata. Since then, to the best of our knowledge, the only further result concerning minimisation of transition-based automata is Casares' NP-completeness proof for the problem of minimising deterministic Rabin automata [15].

In this paper, we focus our attention on generalised Büchi and generalised coBüchi automata, in which the acceptance condition is given, respectively, by conjunctions of clauses "see colour c infinitely often", and by disjunctions of "eventually avoid colour d". Generalised (co)Büchi automata are as expressive as (co)Büchi automata, but they can be more succinct, due to their more complex acceptance condition. These automata appear naturally in the model-checking and synthesis of temporal properties [21, 25, 41]; for instance, `SPOT`'s LTL-synthesis tool transforms a given LTL formula into a generalised Büchi automaton [22, 33]. Also, many efficient algorithms for their emptiness check have been developed [36, 37, 6].

[1] Similar ideas had been previously investigated by Kupferman, Safra and Vardi [31], and Colcombet studied history-determinism in the context of cost functions [20].

[2] These automata were first introduced under the name *good-for-games* (GFG). Currently, these two notions are no longer used interchangeably, although they coincide in the case of ω-automata. We refer to the survey [11] for further discussions.

Several works have approached the problem of reducing the state-space of generalised Büchi automata, which is usually done either by the use of simulations [41, 29] (which do not yield minimal automata), or by the application of SAT solvers [23, 3]. However, to the best of our knowledge, no theoretical result about the exact complexity of this minimisation problem appears in the literature.

Contributions

We provide a polynomial-time minimisation algorithm for history-deterministic generalised coBüchi automata (Theorem 11). Our algorithm uses Abu Radi-Kupferman's minimal history-deterministic coBüchi automaton as a starting point, and reduces the state-space of this automaton in an optimal way to use a generalised coBüchi condition.

We prove that the minimisation problem is NP-complete for history-deterministic generalised Büchi automata (Theorem 28), as well as for deterministic generalised Büchi and generalised coBüchi automata (Theorem 29). We remark that both the NP-hardness and the NP-upper bound are challenging. Indeed, to obtain that the problem is in NP, we first need to prove that a minimal HD generalised Büchi automaton only uses a polynomial number of output colours. Additionally, we adapt a proof from [19] to show that minimising at the same time the number of states and colours is NP-complete for all the previous models, including history-deterministic generalised coBüchi automata (Theorem 30).

We summarise the results about the state-minimisation of transition-based automata in Table 1.

Table 1 Complexity of the minimisation problem for different types of transition-based automata.

Model \ Condition	coBüchi	Büchi	generalised coBüchi	generalised Büchi
Deterministic	Unknown	Unknown	NP-complete (Theorem 29)	NP-complete (Theorem 29)
History-deterministic	PTIME [2]	Unknown	PTIME (Theorem 11)	NP-complete (Theorem 28)

We note that the PTIME complexity of recognising HD automata can be lifted from Büchi and coBüchi conditions to their generalised versions (Corollary 10). This result can be considered folklore, although we have not find it explicitly in the literature. In the long version, we also lift the characterisation based on the G_2 game from (co)Büchi automata to generalised ones (a similar remark was suggested in the conclusion of [8]).

State-minimality. In this paper, we primarily focus our attention on the minimisation of the number of *states* of the automata. In Theorem 30 we also consider the minimisation of both the number of states and colours of the acceptance condition. We highlight that the decision on how we measure the size of the automata is orthogonal to putting the acceptance condition over transitions.

The reader may wonder why we focus on these quantities and not, e.g., on the number of transitions. This choice, which is standard in the literature ([2, 39]), is justified by various reasons. First, the number of transitions of an automaton is polynomial in the number of states. Indeed, we can assume that there are no two transitions between two states over the

same input letter (see [18, Prop.18]), therefore, $|\Delta| \leq |Q|^2|\Sigma|$. Maybe more importantly, in the case of automata over finite words, each state of the minimal automaton carry a precise information about the language it recognises: a residual of it. Ideally, a construction for a state-minimal automaton for an ω-regular language should lead to an understanding of the essential information necessary to represent it.

The interest of minimising both the number of states and the number of colours comes from the fact that the number of colours can be exponential on the number of states, but the size of the representation of the automaton is polynomial in the sum of these quantities.

2 Preliminaries

The disjoint union of two sets A, B is written $A \uplus B$. The *empty word* is denoted ε. A *factor* of a word w is a word u such that there exist words x, y with $w = xuy$. For an infinite word $w \in \Sigma^\omega$, we denote $\mathsf{Inf}(w)$ the set of letters occurring infinitely often in w.

2.1 Automata

We let Σ be a finite alphabet. An *automaton* is a tuple $\mathcal{A} = (Q, \Sigma, q_{\text{init}}, \Delta, \Gamma, \mathsf{col}, W)$, where Q is its set of states, q_{init} its *initial state*, $\Delta \subseteq Q \times \Sigma \times Q$ its set of transitions, Γ its *output alphabet*, $\mathsf{col}\colon \Delta \to \Gamma$ a *labelling with colours*, and $W \subseteq \Gamma^\omega$ its *acceptance condition*. A state q is called *reachable* if there exists a path from q_{init} to q. The *size* of an automaton is its number of states, written $|Q|$. We write $p \xrightarrow{a:c} q$ if $(p, a, q) \in \Delta$ and $\mathsf{col}((p, a, q)) = c$.

A *run* ρ on an infinite word $w = a_1 a_2 \cdots \in \Sigma^\omega$ is an infinite sequence of transitions $\rho = (q_0, a_1, q_1)(q_1, a_2, q_2)(q_2, a_3, q_3), \cdots \in \Delta^\omega$ with $q_0 = q_{\text{init}}$. It is *accepting* if the infinite word $c_1 c_2 \cdots \in \Gamma^\omega$ defined by $c_i = \mathsf{col}(q_{i-1}, a_i, q_i)$, called the *output* of ρ, belongs to W.

The *language of an automaton* $\mathcal{A}$, denoted $\mathcal{L}(\mathcal{A})$, is the set of words that admit an accepting run. We say that two automata $\mathcal{A}$ and $\mathcal{B}$ over the same alphabet are *equivalent* if $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$.

An automaton $\mathcal{A}$ is *deterministic* (resp. *complete*) if, for all $(p, a) \in Q \times \Sigma$, there exists at most (resp. at least) one $q \in Q$ such that $(p, a, q) \in \Delta$. We note that if $\mathcal{A}$ is deterministic, a word $w \in \Sigma^\omega$ admits at most one run in $\mathcal{A}$.

2.2 Acceptance conditions

In this paper we will focus on automata using generalised Büchi and generalised coBüchi acceptance conditions. A generalised Büchi condition can be seen as a conjunction of Büchi conditions, while a generalised coBüchi condition can be seen as a disjunction of coBüchi conditions.

A *generalised Büchi* condition with k colours is defined over the output alphabet $\Gamma = 2^C$, with C a set of k *output colours*, as

$$\mathsf{genB}_C = \{w \in \Gamma^\omega \mid \bigcup \mathsf{Inf}(w) = C\}.$$

It contains sequences of sets of colours such that every colour is seen infinitely often. Usually, we take $C = [k] = \{1, 2, \ldots, k\}$.

The dual condition is the *generalised coBüchi* condition with k colours. That is, we define:

$$\mathsf{genCoB}_C = \{w \in \Gamma^\omega \mid \bigcup \mathsf{Inf}(w) \neq C\}.$$

It contains sequences of sets of colours such that at least one colour is seen finitely often.

The *size of the representation* of an automaton using a generalised (co)Büchi condition with k colours is $|Q| + |\Sigma| + k$; such an automaton can be described in polynomial space in this measure.

A *Büchi condition* (resp. *coBüchi condition*) can be defined as a generalised Büchi (resp. generalised coBüchi) condition in which $k = 1$. In this case, we call *Büchi transitions* (resp. *coBüchi transitions*) the transitions $(p, a, q) \in \Delta$ such that $\mathsf{col}((p, a, q)) = \{1\}$. An automaton using an acceptance condition of type X is called an *X-automaton*.

A language $L \subseteq \Sigma^\omega$ is *(co)Büchi recognisable* if there exists a deterministic (co)Büchi automaton $\mathcal{A}$ such that $\mathcal{A}$ recognises L. These coincide with languages recognised by generalised (co)Büchi automata (see Corollary 9). We note that non-deterministic (generalised) Büchi automata are strictly more expressive, while non-deterministic (generalised) coBüchi automata are as expressive as deterministic ones [34].

2.3 History-determinism

An automaton is called history-deterministic (or HD for short), if there exists a function, called a resolver, that resolves the non-determinism of $\mathcal{A}$ depending only on the prefix of the input word read so far. Formally, a *resolver* for an automaton $\mathcal{A}$ is a function $\sigma : \Sigma^+ \to \Delta$ such that for all words $w = a_0 a_1 \cdots \in \Sigma^\omega$, the sequence $\sigma^*(w) = \sigma(a_0)\sigma(a_0a_1)\sigma(a_0a_1a_2)\cdots \in \Delta^\omega$ (called the *run induced by σ over w*) satisfies:

1. $\sigma^*(w)$ is a run on w in $\mathcal{A}$,
2. if $w \in \mathcal{L}(\mathcal{A})$, then $\sigma^*(w)$ is an accepting run.

An automaton is *history-deterministic* if it admits a resolver. We say that a (deterministic/history-deterministic) automaton is *minimal* if it has a minimal number of states amongst equivalent (deterministic/history-deterministic) automata.

▶ Remark 1. Every deterministic automaton is HD. While the converse is false (see Example 2 below), we note that any language $L \subseteq \Sigma^\omega$ recognised by an HD Büchi automaton (resp. coBüchi automaton), can be recognised by a deterministic Büchi automaton (resp. deterministic coBüchi automaton) [31].

▶ **Example 2** (From [17, Ex. 2.3])**.** Let $\Sigma = \{a, b, c\}$ and $L = \{w \in \Sigma^\omega \mid \{b, c\} \not\subseteq \mathsf{Inf}(w)\}$, that is, L is the set of words that contain either b or c only finitely often.

In Figure 1 we show an automaton recognising L that is not *determinisable by prunning*, that is, it cannot be made deterministic just by removing transitions.

We claim that this automaton is history-deterministic. First, we remark that the only non-deterministic choice appears when reading letter a from the state q_1. A resolver can be defined as follows: whenever we have arrived at q_1 from q_0 (by reading letter c), if we are given letter a we go to state q_2; if we have arrived at q_1 from q_2 (by reading letter b), we will go to state q_0. Therefore, if after some point letter b (resp. letter c) does not appear, we will stay forever in state q_2 (resp. state q_1) and accept.

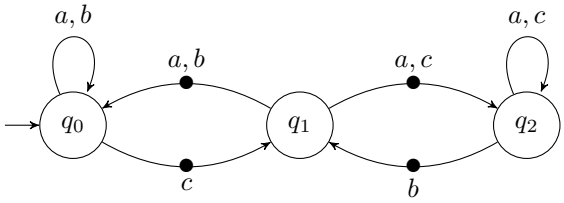


Figure 1 A history-deterministic coBüchi automaton recognising the language $L = \{w \in \Sigma^\omega \mid \{b, c\} \not\subseteq \mathsf{Inf}(w)\}$. CoBüchi transitions are represented with a dot on them.

2.4 Residuals and prefix-independence

Let $L \subseteq \Sigma^\omega$ and $u \in \Sigma^*$. The *residual of L with respect to u* is the language

$$u^{-1}L = \{w \in \Sigma^\omega \mid uw \in L\}.$$

We write $[u]_L = \{v \in \Sigma^* \mid u^{-1}L = v^{-1}L\}$, and $\mathsf{Res}(L)$ for the set of residuals of a language L.

Given an automaton $\mathcal{A}$ and a state q, we denote $\mathcal{A}^q$ the automaton obtained by setting q as initial state, and we refer to $\mathcal{L}(\mathcal{A}^q)$ as the *language recognised by q*. We say that two states q, p are *equivalent*, written $q \sim p$, if they recognise the same language. We note $[q]_\mathcal{A}$ the set of states equivalent to q (we simply write $[q]$ when $\mathcal{A}$ is clear from the context).

We say that an automaton $\mathcal{A}$ is *semantically deterministic* if non-deterministic choices lead to equivalent states, that is, if for every state q and pair of transitions $q \xrightarrow{a} p_1$, $q \xrightarrow{a} p_2$ we have $p_1 \sim p_2$.

If $\mathcal{A}$ is semantically deterministic and $u \in \Sigma^*$ is a word labelling a path from the initial state to q, then $\mathcal{L}(\mathcal{A}^q) = u^{-1}L$. We say then that $u^{-1}L$ is the *residual associated to q*. For a residual $R \in \mathsf{Res}(L)$ we denote Q^R the set of states of $\mathcal{A}$ recognising R. We remark that $Q^R = [q]_\mathcal{A}$ for any state q recognising R.

We say that L is *prefix-independent* if for all $w \in \Sigma^\omega$ and $u \in \Sigma^*$, $w \in L \iff uw \in L$.

▶ Remark 3. A language L is prefix-independent if and only if it has a single residual.

2.5 Morphisms of automaton structures

An *automaton structure* over an alphabet Σ is a tuple $\mathcal{S} = (Q, \Delta)$, where $\Delta \subseteq Q \times \Sigma \times Q$. Let $\mathcal{S}_1 = (Q_1, \Delta_1)$ and $\mathcal{S}_2 = (Q_2, \Delta_2)$ be two automaton structures over the same alphabet. A *morphism of automaton structures* is a mappings $\phi\colon Q_1 \to Q_2$ such that for every $(q, a, q') \in \Delta_1$, $(\phi(q), a, \phi(q')) \in \Delta_2$. We note that such a morphism induces a function $\phi_\Delta\colon \Delta_1 \to \Delta_2$ sending (q, a, q') to $(\phi(q), a, \phi(q'))$. We also denote this function ϕ, whenever no confusion arises, and denote a morphism of automaton structures by $\phi\colon \mathcal{S}_1 \to \mathcal{S}_2$.

3 First properties and examples

We discuss a further example of a history-deterministic automaton and state some well-known facts about these automata that will be relevant for the rest of the paper.

3.1 A central example

The following automata will be used as a running example in Section 4.

▶ **Example 4.** Let Σ_n be an alphabet of size n, and let

$$L_n = \{w \in \Sigma_n^\omega \mid \text{ for some } x \in \Sigma_n \text{ the factor } xx \text{ appears only finitely often in } w\}.$$

On the left of Figure 2 we show a history-deterministic generalised coBüchi automaton recognising L_n with just 2 states (we show it for $\Sigma_3 = \{a, b, c\}$, but the construction clearly generalises to any n). The set of colours is $C = \{1, 2, 3\}$, and we accept if eventually some colour is not produced. A resolver can be defined as follows: in a round-robin fashion, we bet that the factor that does not appear is aa, then bb, then cc. While factor aa is not seen, we will take transition $q_0 \xrightarrow{a} q_1$ whenever letter a is read, to try to avoid colour 1. Whenever factor aa is read, we switch to the corresponding strategy with letter b, trying to avoid colour 2. If eventually factor xx is not produced, for $x \in \{a, b, c\}$, then some colour will forever be avoided.

We show a deterministic coBüchi automaton for L_3 on the right of Figure 2. Applying the characterisation of Abu Radi and Kupferman [2] (see Lemma 15), we can prove that this automaton is minimal amongst HD coBüchi automata. More generally, we can prove that a minimal HD coBüchi automaton for L_n has at least $2n$ states , and in fact, in this case, this optimal bound can be achieved with a deterministic automaton.

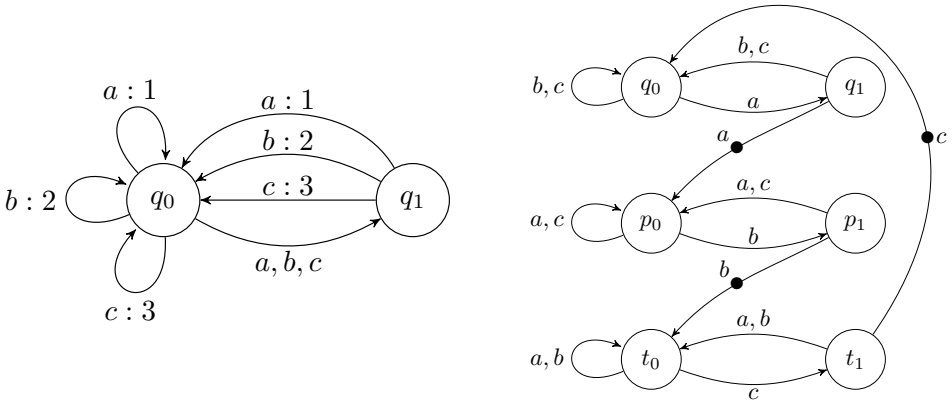


Figure 2 On the left, a history-deterministic generalised coBüchi automaton recognising the language L_3 of words eventually avoiding factor xx for some letter x. On the right, a minimal deterministic coBüchi automaton for the same language (coBüchi transitions have a dot on them). In both cases, the initial state is irrelevant, as the language is prefix-independent.

3.2 Duality Büchi - coBüchi

▶ Remark 5. Let $\mathcal{A}$ be a deterministic generalised Büchi automaton of size n and using k output colours. It suffices to replace the acceptance condition $\mathsf{genB}_{[k]}$ with $\mathsf{genCoB}_{[k]}$ to obtain a deterministic generalised coBüchi automaton of size n and using k output colours recognising the complement language $\Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$. Symmetrically, we can turn any deterministic generalised coBüchi automaton into a deterministic generalised Büchi automaton with the same number of states and colours recognising the complement language. As a consequence, the minimisations of deterministic generalised Büchi automaton and deterministic generalised coBüchi automaton are linear-time-equivalent problems.

We highlight that the hypothesis of determinism in the previous remark is crucial. This duality property no longer holds for non-deterministic (or history-deterministic) automata.

▶ **Lemma 6.** *There exists a history-deterministic generalised coBüchi automaton $\mathcal{A}$ such that any history-deterministic generalised Büchi automaton recognising $\Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$ has strictly more states than $\mathcal{A}$.*

Such an example is provided by the language L_3 from Example 4 (in the long version we prove that any non-deterministic generalised Büchi automaton recognising $\Sigma^\omega \setminus \mathcal{L}(\mathcal{A})$ has at least 3 states). Relatedly, for the non-generalised conditions, Kuperberg and Skrzypczak showed that the gap between an HD coBüchi and an HD Büchi automaton for the complement language can be exponential as well [30], using the link between complementation and determinisation of HD automata from [7, Thm 4].

3.3 From generalised (co)Büchi to (co)Büchi

The idea of this construction is to define a particular deterministic coBüchi automaton recognizing genCoB_C, and to associate it to the input generalised coBüchi automaton via a cascade composition (defined below) in order to obtain the wanted coBüchi automaton. We will now detail these different steps.

Deterministic coBüchi automaton for genCoB_C. Let $C = \{1, 2, \cdots, k\}$ be a set of k colours; for convenience, in the context of colours, we will use the symbol $+$ to denote addition modulo k, in particular, $k + 1 = 1$. We build a deterministic coBüchi automaton $\mathcal{D}_C^{\mathrm{coB}}$ over the alphabet $\Gamma = 2^C$ recognising the language genCoB_C. It has as a state q_i for each colour $i \in C$ and contains the transitions $q_i \xrightarrow{X:\emptyset} q_i$, if $i \notin X$, and $q_i \xrightarrow{X:1} q_{i+1}$, if $i \in X$, for all $X \in \Gamma$. The initial state is arbitrary.

We claim that the automaton $\mathcal{D}_C^{\mathrm{coB}}$ recognises the language genCoB_C. First, we remark that the accepting runs of $\mathcal{D}_C^{\mathrm{coB}}$ are exactly those that eventually remain forever in a state q_i. Let $w = w_1 w_2 \cdots \in 2^C$. If w is accepted by $\mathcal{D}_C^{\mathrm{coB}}$, then the run on w eventually stays in a q_i, so w eventually does not contain colour i, and $w \in \mathsf{genCoB}_C$. Conversely, if w is rejected by $\mathcal{D}_C^{\mathrm{coB}}$, it takes all transitions $q_i \xrightarrow{X:1} q_{i+1}$ infinitely often, so w must contain all colours in C infinitely often.

We define in a similar fashion a deterministic Büchi automaton $\mathcal{D}_C^{\mathrm{B}}$ recognising the language genB_C, simply by changing the acceptance condition of genCoB_C to genB_C.

▶ Remark 7. The automaton $\mathcal{D}_C^{\mathrm{coB}}$ has k states, but exponentially many transitions. This is made possible by the fact that its alphabet Γ is exponential in the number of colours k.

Cascade composition of automata. Let $\mathcal{A} = (Q_\mathcal{A}, \Sigma_\mathcal{A}, q_{\mathrm{init}}^\mathcal{A}, \Delta_\mathcal{A}, \Gamma_\mathcal{A}, \mathsf{col}_\mathcal{A}, W_\mathcal{A})$ and $\mathcal{B} = (Q_\mathcal{B}, \Sigma_\mathcal{B}, q_{\mathrm{init}}^\mathcal{B}, \Delta_\mathcal{B}, \Gamma_\mathcal{B}, \mathsf{col}_B, W_B)$ be two automata such that $\Sigma_\mathcal{B} = \Gamma_\mathcal{A}$ (i.e., $\mathcal{B}$ is an automaton over the set of output colours of $\mathcal{A}$). The *cascade composition* of $\mathcal{A}$ and $\mathcal{B}$ is the automaton over $\Sigma_\mathcal{A}$ defined as:

$$\mathcal{B} \circ \mathcal{A} = (Q_\mathcal{A} \times Q_\mathcal{B}, \Sigma_\mathcal{A}, (q_{\mathrm{init}}^\mathcal{A}, q_{\mathrm{init}}^\mathcal{B}), \Delta', \Gamma_\mathcal{B}, \mathsf{col}', W_\mathcal{B}),$$

with transitions $(p_\mathcal{A}, p_\mathcal{B}) \xrightarrow{a:c} (q_\mathcal{A}, q_\mathcal{B})$ if $p_\mathcal{A} \xrightarrow{a:b} q_\mathcal{A} \in \Delta_\mathcal{A}$ and $p_\mathcal{B} \xrightarrow{b:c} q_\mathcal{B} \in \Delta_\mathcal{B}$.

Intuitively, given a word in $\Sigma_\mathcal{A}^\omega$, we feed the output of $\mathsf{col}_\mathcal{A}$ directly as input to $\mathcal{B}$, while keeping track of the progression in both automata. We accept according to the acceptance condition of $\mathcal{B}$.

▶ **Lemma 8** (Folklore). *Let $\mathcal{A}$ be an automaton with acceptance condition $W \subseteq \Gamma^\omega$, and let $\mathcal{B}$ be a deterministic automaton over Γ recognising W. Then $\mathcal{B} \circ \mathcal{A}$ recognises $\mathcal{L}(\mathcal{A})$, and the automaton $\mathcal{B} \circ \mathcal{A}$ is history-deterministic (resp. deterministic) if and only if $\mathcal{A}$ is.*

Thus, to convert a generalised coBüchi automaton $\mathcal{A}$ to an equivalent coBüchi one, we can just compose it with $\mathcal{D}_C^{\mathrm{coB}}$. The symmetric result holds for generalised Büchi automata.

▶ **Corollary 9** (Folklore). *Let $\mathcal{A}$ be a generalised coBüchi automaton using C as output colours. The automaton $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{A}$ is a coBüchi automaton equivalent to $\mathcal{A}$ which is (history-)deterministic if and only if $\mathcal{A}$ is. Moreover, $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{A}$ can be computed in polynomial time in the size of the representation of $\mathcal{A}$. The same is true for generalised Büchi automata.*

We note that $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{A}$ has $k \cdot |\mathcal{A}|$ states, where $k = |C|$, but it might have exponentially many transitions in k. However, we underline that we can compute $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{A}$ from $\mathcal{A}$ in polynomial time in the size of its representation. Indeed, we just need to compose $\mathcal{A}$ with the restriction of $\mathcal{D}_C^{\mathrm{coB}}$ to transitions whose letters are subsets of colours that appear in $\mathcal{A}$.

Deciding history-determinism. The problem of deciding whether an automaton is HD is known to be in PTIME for Büchi and coBüchi automata [30, 4, 8]. Combining this fact with Corollary 9, we directly obtain:

▶ **Corollary 10.** *Given a generalised Büchi (resp. generalised coBüchi) automaton, it is in* PTIME *to decide whether it is history-deterministic.*

A different proof of Corollary 10, which goes through the G_2 game [4], is presented in the long version.

4 Polynomial-time minimisation of HD generalised coBüchi automata

In this section we present one of the main contributions of the paper (Theorem 11): history-deterministic generalised coBüchi automata can be minimised in polynomial time.

▶ **Theorem 11.** *Given a history-deterministic generalised coBüchi automaton, we can build in polynomial time in its representation an equivalent history-deterministic generalised coBüchi automaton with a minimal number of states.*

The proof of this result strongly relies on the construction of minimal coBüchi automata by Abu Radi and Kupferman [2]. We will show that, for a coBüchi recognisable language L, we can extract a minimal HD generalised coBüchi automaton for L from its minimal HD coBüchi automaton.

In Section 4.1 we introduce some terminology and state the main property satisfied by the minimal HD coBüchi automaton of Abu Radi and Kupferman. We then present our construction, decomposing it in two steps for simplicity: first we construct a minimal HD generalised coBüchi automaton in the case of prefix-independent languages in Section 4.2, and in Section 4.3, we show how to get rid of the prefix-independence assumption.

4.1 Minimisation of HD coBüchi automata

Safe components and safe languages. A path $q \rightsquigarrow q'$ in a coBüchi automaton is *safe* if no coBüchi transition appears on it. Let $\mathcal{A}_{\mathsf{safe}}$ be the automaton obtained by removing from $\mathcal{A}$ all coBüchi transitions. A *safe component* of $\mathcal{A}$ is a strongly connected component (i.e., a maximal set of states which are all reachable from each other) of $\mathcal{A}_{\mathsf{safe}}$; formally, this is an automaton structure $\mathcal{S} = (S, \Delta_{\mathcal{S}})$ with $S \subseteq Q_{\mathcal{A}}$ and $\Delta_{\mathcal{S}} \subseteq \Delta_{\mathcal{A}}$. We let $\mathsf{Safe}(\mathcal{A})$ be the set of safe components of $\mathcal{A}$.

We define the *safe language of a state q* as:

$$\mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^q) = \{w \in \Sigma^\omega \mid \text{ there is a run } q \overset{w}{\rightsquigarrow} \text{ in } \mathcal{A}_{\mathsf{safe}}\}.$$

▶ **Example 12.** The safe components of the automaton on the right of Figure 2 (page 7) have as set of states: $S_1 = \{q_0, q_1\}$, $S_2 = \{p_0, p_1\}$ and $S_3 = \{t_0, t_1\}$. The safe language of q_0 is $\mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^{q_0}) = \{w \in \Sigma^\omega \mid w \text{ does not contain the factor } aa\}$.

Nice coBüchi automata. We say that a coBüchi automaton $\mathcal{A}$ is in *normal form* if all transitions between two different safe components are coBüchi transitions. We note that any coBüchi automaton can be put in normal form without modifying its language by setting all transitions between two different safe components to be coBüchi. We say that $\mathcal{A}$ is *safe deterministic* if $\mathcal{A}_{\mathsf{safe}}$ is a deterministic automaton. That is, if the non-determinism of $\mathcal{A}$ appears exclusively in coBüchi transitions. We say that $\mathcal{A}$ is *nice* if all its states are reachable, it is semantically deterministic, in normal form, and safe deterministic.

It is not difficult to see that any history-deterministic automaton $\mathcal{A}$ can be assumed to be semantically deterministic. This means that different choices made by a resolver from the same state with different histories must be consistent with the residual, i.e. must lead to states accepting the same language, which is the language $u^{-1}L(\mathcal{A})$ after reading a word u. Kuperberg and Skrzypczak showed the more involved result that we can moreover assume safe determinism [30]. All in all, we have:

▶ **Lemma 13** ([30]). *Every history-deterministic coBüchi automaton $\mathcal{A}$ can be turned in polynomial time into an equivalent nice HD coBüchi automaton $\mathcal{A}_{\text{nice}}$ such that:*

- $|\mathcal{A}_{\text{nice}}| \leq |\mathcal{A}|$,
- *For every safe component $\mathcal{S}$ of $\mathcal{A}_{\text{nice}}$, there is some safe component $\mathcal{S}'$ of $\mathcal{A}$ with $|\mathcal{S}| \leq |\mathcal{S}'|$.*

Although the second item of the previous proposition is not explicitly stated in [30], it simply follows from the fact that all the transformations used to turn $\mathcal{A}$ into a nice automaton either add coBüchi transitions to $\mathcal{A}$ or remove transitions from it. These operations can only subdivide safe components.

Minimal HD coBüchi automata. We present the necessary conditions for the minimality of history-deterministic coBüchi automata identified by Abu Radi and Kupferman [2].

We say that a coBüchi automaton $\mathcal{A}$ is *safe centralised* if for all equivalent states $q \sim p$, if the safe languages of q and p are comparable for the inclusion relation ($\mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^q) \subseteq \mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^p)$ or vice versa), then they are in the same safe component of $\mathcal{A}$. It is *safe minimal* if for all states $q \sim p$, the equality $\mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^q) = \mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^p)$ implies $q = p$.

▶ **Example 14.** The automaton on the right of Figure 2 is safe minimal and safe centralised. However, the automaton from Figure 1 is not safe centralised, as $\mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^{q_1}) = \emptyset \subseteq \mathcal{L}_{\mathsf{Safe}}(\mathcal{A}^{q_2})$, but q_1 and q_2 appear in different safe components.

▶ **Lemma 15** ([2, Lemma 3.5]). *Let $\mathcal{A}_{\min}$ be a nice, safe minimal and safe centralised HD coBüchi automaton. Then, for any equivalent nice HD coBüchi automaton $\mathcal{A}$ there is an injection $\eta\colon \mathsf{Safe}(\mathcal{A}_{\min}) \to \mathsf{Safe}(\mathcal{A})$ such that for every safe component $\mathcal{S} \in \mathsf{Safe}(\mathcal{A}_{\min})$, it holds that $|\mathcal{S}| \leq |\eta(\mathcal{S})|$.*

Minimality of such automata directly follows:

▶ **Corollary 16.** *Let $\mathcal{A}_{\min}$ be a nice, safe minimal and safe centralised HD coBüchi automaton. Then, the number of states of $\mathcal{A}_{\min}$ is minimal among all HD coBüchi automata recognising the same language.*

▶ **Theorem 17** ([2, Theorem 3.15]). *Any coBüchi recognisable language L can be recognised by a nice, safe minimal and safe centralised HD coBüchi automaton $\mathcal{A}_L^{\mathsf{coB}}$. Moreover, such an automaton $\mathcal{A}_L^{\mathsf{coB}}$ can be computed in polynomial time from any HD coBüchi automaton recognising L.*

4.2 Minimal HD generalised coBüchi automata: prefix-independent case

In this subsection, we show how to minimise history-deterministic generalised coBüchi automata recognising prefix-independent languages. The prefix-independence hypothesis will be removed in the next subsection.

Let $L \subseteq \Sigma^\omega$ be a prefix-independent coBüchi recognisable language, and let $\mathcal{A}$ be a history-deterministic generalised coBüchi automaton recognising it.

Combining Corollary 9 and Theorem 17, we obtain that we can build in polynomial time the minimal HD coBüchi automaton $\mathcal{A}_L^{\mathsf{coB}}$ for L. Let $\mathsf{Safe}(\mathcal{A}_L^{\mathsf{coB}}) = \{\mathcal{S}_1, \dots, \mathcal{S}_k\}$ be an enumeration of the safe components of $\mathcal{A}_L^{\mathsf{coB}}$, with S_i and Δ_i the sets of states and transitions of each safe component, respectively. We show how to build an HD generalised coBüchi automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ of size $n_{\max} = \max_{1 \le i \le k} |S_i|$ and using k output colours.

Intuitively, $\mathcal{A}_L^{\mathsf{genCoB}}$ will be the full automaton (it contains transitions between all pairs of states, for all input letters). Since $|S_i| \le n_{\max}$, we can map each safe component $\mathcal{S}_i$ to this full automaton via a morphism ϕ_i, and use (the non-appearance of) colour i to accept runs that eventually would have stayed in the safe component $\mathcal{S}_i$ in $\mathcal{A}_L^{\mathsf{coB}}$. That is, the transitions of $\mathcal{A}_L^{\mathsf{genCoB}}$ that are "safe-for-colour i" will be exactly those in $\phi_i(\mathcal{S}_i)$.

Formally, let $\mathcal{A}_L^{\mathsf{genCoB}} = (Q, \Sigma, q_{\text{init}}, \Delta, \Gamma, \mathsf{col}, \mathsf{genCoB})$ with:

- $Q = \{p_1, p_2, \dots, p_{n_{\max}}\}$,
- $q_{\text{init}} = p_1$ (any state can be chosen as initial),
- $\Delta = Q \times \Sigma \times Q$,
- $\Gamma = 2^{\{1,\dots,k\}}$.

Finally, we define the colour labelling $\mathsf{col} \colon \Delta \to \Gamma$. For each $1 \le i \le k$, let $\phi_i \colon \mathcal{S}_i \to (Q, \Delta)$ be any injective morphism of automaton structures (such a morphism exists since $|\mathcal{S}_i| \le n_{\max}$ and (Q, Δ) is the full automaton structure). We put colour i in a transition $e \in \Delta$ if and only if there is no transition $e' \in \Delta_i$ such that $\phi_i(e') = e$. That is, $\mathsf{col}(e) = \{i \mid \phi_i^{-1}(e) = \emptyset\}$.

▶ Remark 18. We remark that this colour labelling uses some arbitrary choices, namely, the way we map the different safe components of $\mathcal{A}_L^{\mathsf{coB}}$ to the full automaton of size $n_{\max}$. In particular, there is no unique minimal HD generalised coBüchi automaton recognising L. By a slight abuse of notation, we denote $\mathcal{A}_L^{\mathsf{genCoB}}$ one automaton originated by this procedure.

▶ **Example 19.** The automaton on the left of Figure 2 (page 7) (almost) corresponds to this construction. Indeed, it has been obtained by assigning a colour to each safe component of $\mathcal{A}_L^{\mathsf{coB}}$ (on the right) and superposing them in a 2-state automaton. To simplify its presentation, we have removed some unnecessary transitions of $\mathcal{A}_L^{\mathsf{genCoB}}$, that is why the automaton displayed is not the full-automaton.

▶ **Proposition 20** (Correctness). *Let L be a prefix-independent language that is coBüchi recognisable. The automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ is history-deterministic and recognises L.*

Proof sketch. If w admits an accepting run ρ in $\mathcal{A}_L^{\mathsf{genCoB}}$, then its run eventually does not produce some colour i in its output. This means that, eventually, such a run is the ϕ_i-projection of a run in a safe component of $\mathcal{A}_L^{\mathsf{coB}}$, so $w \in L$. This is where we use prefix-independence: as soon as there is a witness that a suffix of w is also a suffix of a word in L, then the whole word w is in L as well.

A resolver for $\mathcal{A}_L^{\mathsf{genCoB}}$ can be defined as follows: in a round-robin fashion we follow the different safe components of $\mathcal{A}_L^{\mathsf{coB}}$. If a colour i is produced while we are trying to avoid it, we go back to p_1 and try to avoid colour $i' = (i+1) \mod k$ by following the safe component $\mathcal{S}_{i'}$. If a word w belongs to L, it eventually admits a safe path in $\mathcal{A}_L^{\mathsf{coB}}$, so it will be accepted by this resolver. ◀

▶ **Proposition 21** (Minimality). *Let $\mathcal{B}$ be a history-deterministic generalised coBüchi automaton recognising a prefix-independent language L. Then, $|\mathcal{A}_L^{\mathsf{genCoB}}| \le |\mathcal{B}|$.*

Proof. Let $C = \{1, \dots, k\}$ be the set of output colours used by the acceptance condition of $\mathcal{B}$ and let $\mathcal{D}_C^{\mathrm{coB}}$ be the coBüchi automaton recognising genCoB_C presented in Section 3.3. By Corollary 9, $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{B}$ is a history-deterministic automaton recognising L. Moreover, the states of $\mathcal{D}_C^{\mathrm{coB}} \circ \mathcal{B}$ are a disjoint union $Q_1 \uplus Q_2 \uplus \cdots \uplus Q_k$ such that:

- $|Q_i| = |\mathcal{B}|$,
- all transitions leaving Q_i are coBüchi transitions going to Q_{i+1}, where $Q_{k+1} = Q_1$.

Therefore, each safe component $\mathcal{S}$ of $\mathcal{D}_C^{\mathsf{coB}} \circ \mathcal{B}$ is included in some Q_i, so $|\mathcal{S}| \leq |\mathcal{B}|$. By Lemma 13, we can turn $\mathcal{D}_C^{\mathsf{coB}} \circ \mathcal{B}$ into a nice HD coBüchi automaton $\mathcal{B}'$ satisfying that none of its safe components is larger than $|\mathcal{B}|$.

By Lemma 15 there is an injection $\eta\colon \mathsf{Safe}(\mathcal{A}_L^{\mathsf{coB}}) \to \mathsf{Safe}(\mathcal{B}')$ such that $|\mathcal{S}| \leq |\eta(\mathcal{S})|$ for all safe component $\mathcal{S}$ of $\mathcal{A}_L^{\mathsf{coB}}$. In particular, if $\mathcal{S}_{\max}$ is a safe component of maximal size in $\mathcal{A}_L^{\mathsf{coB}}$, we obtain: $|\mathcal{A}_L^{\mathsf{genCoB}}| = |\mathcal{S}_{\max}| \leq |\eta(\mathcal{S}_{\max})| \leq |\mathcal{B}|$. ◀

4.3 Minimal HD generalised coBüchi automata: general case

We now describe the polynomial-time construction for minimising a given HD generalised coBüchi automaton (without the prefix-independence assumption). For the optimality proof, we can reduce to the simplest prefix-independent case.

We fix an HD generalised coBüchi automaton $\mathcal{A}$ recognising a language L. As before, using Corollary 9 and the minimisation procedure of Abu Radi and Kupferman, we can obtain the minimal HD coBüchi automaton $\mathcal{A}_L^{\mathsf{coB}}$ in polynomial time. We show how to convert it to an equivalent HD generalised coBüchi automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ of minimal size.

Let $R_1, R_2, \cdots, R_m$ be all the distinct residual languages of L, i.e., languages of the form $u^{-1}L$ for some finite word $u \in \Sigma^*$. The case $m = 1$ corresponds to the prefix-independent case treated in Section 4.2. We note that these residuals induce a partition of the states of $\mathcal{A}_L^{\mathsf{coB}}$ into $Q^{R_1}, \ldots, Q^{R_m}$, where the states in Q^{R_j} recognise R_j. We assume that $R_1 = L$ is the residual corresponding to the initial state of $\mathcal{A}_L^{\mathsf{coB}}$. Let $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_k$ be the safe components of $\mathcal{A}_L^{\mathsf{coB}}$, with S_i and Δ_i as sets of states and transitions, respectively. For each residual language R_j, define n_j as the largest number of states recognising R_j appearing in a safe component of $\mathcal{A}_L^{\mathsf{coB}}$. That is,

$$n_j = \max_{1 \leq i \leq k} |S_i \cap Q^{R_j}|.$$

We shall construct a language-equivalent HD generalised coBüchi automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ with $n_1 + n_2 + \cdots + n_m$ states. Towards this, for each residual language R_j, let $P_j = \{p_j^1, p_j^2, \cdots, p_j^{n_j}\}$ be a set of n_j elements. The automaton $\mathcal{A}_L^{\mathsf{genCoB}} = (Q, \Sigma, q_{\text{init}}, \Delta, \Gamma, \mathsf{col}, \mathsf{genB})$ is given by:

- $Q = P_1 \uplus P_2 \uplus \cdots \uplus P_m$.
- $q_{\text{init}} = p_1^1$ (any state corresponding to the residual of the initial state of $\mathcal{A}_L^{\mathsf{coB}}$ would work).
- Let (q, a, q') be a transition in $\mathcal{A}_L^{\mathsf{coB}}$, with $q \in Q^{R_j}$ and $q' \in Q^{R_{j'}}$. Then, $(p, a, p') \in \Delta$ for all $p \in P_j$ and $p' \in P_{j'}$.
- $\Gamma = 2^{\{1,\ldots,k\}}$.

One way of picturing $\mathcal{A}_L^{\mathsf{genCoB}}$ is by taking the automaton of residuals of L and making n_j copies of the state corresponding to each residual R_j (while keeping all transitions).

We now describe the colour labelling $\mathsf{col}\colon \Delta \to \Gamma$. Informally, each safe component $\mathcal{S}_i$ is mapped into $\mathcal{A}_L^{\mathsf{genCoB}}$ so that the states of $S_i \cap R_j$ are mapped into P_j. These safe components are then "superimposed" upon each other and coloured appropriately, so that a run eventually staying in $\mathcal{S}_i$ in $\mathcal{A}_L^{\mathsf{coB}}$ corresponds to a run in $\mathcal{A}_L^{\mathsf{genCoB}}$ that eventually avoids colour i.

More formally, for $i \in [1, k]$, let $\phi_i\colon \mathcal{S}_i \to \mathcal{A}_L^{\mathsf{genCoB}}$ be an injective morphism such that $\phi_i(q) \in P_j$ if $q \in Q^{R_j}$. Such injective morphism does indeed exist, by the choice of n_j and the fact that $\mathcal{A}_L^{\mathsf{genCoB}}$ contains all transitions consistent with the residuals. The transitions of Δ that are i-safe are defined to be exactly those that are the image by ϕ_i of some transition in $\mathcal{S}_i$. That is, for $e \in \Delta$, the labelling $\mathsf{col}(e)$ contains exactly the colours in $\{i \mid \phi_i^{-1}(e) = \emptyset\}$.

▶ Remark 22. We remark that the automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ obtained in this way uses a polynomial number of output colours in the size of the minimal HD coBüchi automaton $\mathcal{A}_L^{\mathsf{coB}}$ (see also long version). More precisely, the number k of colours is the number of safe components of $\mathcal{A}_L^{\mathsf{coB}}$. However, this number of colours is not necessarily optimal (see Theorem 30).

The correctness of our construction, stated below, is proven similarly to Proposition 20.

▶ **Proposition 23** (Correctness). *Let L be a coBüchi recognisable language. The automaton $\mathcal{A}_L^{\mathsf{genCoB}}$ is history-deterministic and recognises L.*

In particular, the resolver for $\mathcal{A}_L^{\mathsf{genCoB}}$ is defined as in Proposition 20: it follows the different safe components of $\mathcal{A}_L^{\mathsf{coB}}$ in a round-robin fashion, by trying to avoid colour some colour i while moving in $\phi_i(S_i)$, then if colour i is seen it switches to $i' = (i+1) \mod k$, etc.

We explain how to obtain the minimality of $\mathcal{A}_L^{\mathsf{genCoB}}$, stated below. We reduce to the prefix-independent case, using a technique from [12].[3] Full proofs can be found in the long version.

▶ **Proposition 24** (Minimality). *Let $\mathcal{B}$ be a history-deterministic generalised coBüchi automaton recognising a language L. Then, $|\mathcal{A}_L^{\mathsf{genCoB}}| \leq |\mathcal{B}|$.*

For each residual $R = u^{-1}L \in \mathsf{Res}(L)$, we define the *local alphabet at R*, as:

$$\Sigma\restriction_R = \{v \in \Sigma^+ \mid [uv] = [u] \text{ and for any proper prefix } v' \text{ of } v,\ [uv'] \neq [v]\}.$$

That is, if $\mathcal{A}$ is a semantically deterministic automaton with states Q, then $\Sigma\restriction_R$ is the set of words that connect states in Q^R. Note that in general $\Sigma\restriction_R$ may be infinite, however this is harmless in this context, and we will freely allow ourselves to talk about automata over infinite alphabets. Also, $\Sigma\restriction_R$ is empty if and only if all the states of Q^R are *transient*, that is, they do not occur in any cycle of the automaton. For simplicity, in the following we assume that no state of $\mathcal{A}$ is transient; the general case is treated in detail in the long version.

We define the *localisation of L to* a residual $R \in \mathsf{Res}(L)$ as the language over the alphabet $\Sigma\restriction_R$ given by: $L\restriction_R = \{w \in {\Sigma\restriction_R}^\omega \mid w \in R\}$.

▶ Remark 25. For every residual R, $L\restriction_R$ is a prefix-independent language. It corresponds to infinite words whose letters are in $\Sigma\restriction_R$ that is, going from Q^R to Q^R, and eventually avoid seeing some colour on such paths. The prefix-independence of $L\restriction_R$ follows from the fact that $L\restriction_R$ has only itself as residual, on alphabet $\Sigma\restriction_R$.

Let $\mathcal{A}$ be a semantically deterministic generalised coBüchi automaton with k colours recognising $L \subseteq \Sigma^\omega$. For each recurrent residual R of L we define $\mathcal{A}\restriction_R$ to be the generalised coBüchi automaton over $\Sigma\restriction_R$ given by:

- the set of states is Q^R, that is, the set of states of $\mathcal{A}$ recognising R.
- the initial state is arbitrary,
- the acceptance condition is genCoB_C (for C the output colours of $\mathcal{A}$),
- there is a transition $q \xrightarrow{w:X} p$, with $w \in \Sigma\restriction_R$, $X \in 2^{\{1,\dots,k\}}$, if there is a path from q to p labelled w and producing the set of colours X in $\mathcal{A}$.

[3] An alternative proof scheme is to extend the proof of Proposition 21 to the general case, by taking into account the residuals of the language. For this, we need a refinement of Lemma 15, stating that the injection η satisfies that, for every residual R and safe component $\mathcal{S}$ of $\mathcal{A}_L^{\mathsf{coB}}$, $|\mathcal{S} \cap R| \leq |\eta(\mathcal{S}) \cap R|$. The proof of Abu Radi and Kupferman [2] does indeed lead to this result, but it is not explicitly stated in this form.

▶ **Lemma 26.** *The automaton $(\mathcal{A}\restriction_R)^q$ recognises $L\restriction_R$ for each $q \in Q^R$. Moreover, if $\mathcal{A}^q$ is history-deterministic, so is $(\mathcal{A}\restriction_R)^q$.*

Using the fact that (safe) paths between states in Q^R are the same in $\mathcal{A}$ or in $\mathcal{A}\restriction_R$, combined with Lemma 15, we can prove:

▶ **Lemma 27.** *$\mathcal{A}_L^{\mathsf{coB}}\restriction_R$ is a minimal HD coBüchi automaton recognising $L\restriction_R$. Moreover, a maximal safe component of this automaton has size n_j.*

To conclude the proof of Proposition 24, we combine Lemma 27 with Proposition 21 to show that $|Q_{\mathcal{B}}^{R_j}| \geq n_j$. This implies: $|\mathcal{B}| \geq n_1 + \cdots + n_m = |\mathcal{A}_L^{\mathsf{genCoB}}|$.

5 NP-completeness of minimisation of deterministic and HD generalised Büchi automata

In this section we contrast the polynomial-time complexity previously obtained for minimising history-deterministic generalised coBüchi automata with the NP-hardness of the minimisation of deterministic generalised (co)Büchi and history-deterministic generalised Büchi automata.

▶ **Theorem 28.** *The following problem is NP-complete: Given a history-deterministic generalised Büchi automaton $\mathcal{A}$ and a number n, decide whether there is an equivalent history-deterministic generalised Büchi automaton with at most n states.*

▶ **Theorem 29.** *The minimisation of the number of states of deterministic generalised Büchi and deterministic generalised coBüchi automata is NP-complete.*

We show NP-hardness of the minimisation problems in the deterministic and history-deterministic Büchi cases simultaneously. The NP-hardness for the deterministic coBüchi case follows directly. Our reduction is from a suitable version of the 3-colouring problem.

We further consider the problem of minimising both colours and states simultaneously for generalised (co)Büchi automata. For the automata classes appearing in the previous theorems (deterministic and history-deterministic generalised Büchi automata), it easily follows that this problem is NP-complete. We focus therefore in the case of history-deterministic generalised coBüchi, for which the minimisation of states has proven to be polynomial (Theorem 11). We show that, even in this case, minimising both states and colours is NP-complete.

▶ **Theorem 30.** *The following problem is NP-complete: Given a history-deterministic generalised coBüchi automaton $\mathcal{A}$, and numbers n and k, decide whether there is an equivalent history-deterministic generalised coBüchi automaton with at most n states and k colours.*

We obtain the lower bound by adapting the proof of NP-hardness of the minimising of Rabin pairs, given in [19], itself inspired from [28]. Details can be found in the long version.

5.1 Containment in NP and bounds on the necessary number of colours

In this section we address an important subtlety of our minimisation problems: We minimise the number of states, but the number of colours used by a generalised Büchi or generalised coBüchi automaton may be exponential in its number of states.

In order to show that the problems at hand are in NP, we need to show that the minimal deterministic and history-deterministic automata require a number of colours that is polynomial in the size of the input (that is, the number of states and colours of the input automaton). This turns out to be true, although not trivial. Full proofs are in the long version.

▶ **Lemma 31.** *Let $\mathcal{A}$ be a deterministic generalised Büchi automaton with n states and k colours. Then there is an equivalent deterministic generalised Büchi automaton with a minimal number of states and using $O(n^2k)$ colours.*

▶ **Lemma 32.** *Let $\mathcal{A}$ be a history-deterministic generalised Büchi automaton with n states and k colours. Then there is an equivalent history-deterministic generalised Büchi automaton with a minimal number of states and using $O(n^3k^2)$ colours.*

This allows us to obtain an NP algorithm as we only need to guess an automaton with polynomially many states and colours, and check equivalence with the input automaton. The latter test can be done in polynomial time (see for instance [40, Thm. 4]).

As an additional result, we show that the previous lemmas do not hold for general non-deterministic automata: minimising an automaton may blow up its number of colours.

▶ **Proposition 33.** *There exists a family of non-deterministic generalised Büchi automata $(\mathcal{A}_n)_{n\in\mathbb{N}}$ such that for all n, $\mathcal{A}_n$ uses $n+1$ states and 2 colours and a minimal automaton equivalent to $\mathcal{A}_n$ requires 2^n colours.*

5.2 Hardness of state minimisation

We provide a reduction from the 3-colouring problem. We construct from a given graph G a deterministic automaton $\mathcal{A}_G$ such that:

- If G is 3-colourable then there is a 3-state deterministic automaton equivalent to $\mathcal{A}_G$, and
- if G is not 3-colourable then there is no automaton $\mathcal{B}$ (deterministic or not) with 3 states equivalent to $\mathcal{A}_G$.

This establishes the hardness of state-minimisation for deterministic and history-deterministic automata simultaneously. The full proof is in the long version. We only present here the languages we use and a sketch of the first item. The second item is obtained by a refined case analysis over the cycles of generalised Büchi automata with three states.

Given an undirected graph $G = (V, E)$, we define the *neighbourhood* of a vertex v as the set $N[v] = \{v' \in V \mid \{v, v'\} \in E\}$, and its *strict neighbourhood* as $n(v) = N[v] \setminus \{v\}$.

We consider the alphabet $\Sigma = V$. For each $v \in V$, we define the language:

$$L_v = (V^*vv)^\omega \cup (V^*(V \setminus N[v]))^\omega \quad \text{and we let} \quad L_G = \bigcap_{v\in V} L_v.$$

In words, a sequence of nodes is in L_G if for all $v \in V$ it either has infinitely many factors vv or sees a vertex that is not a neighbour of v infinitely many times.

The first item is proven by the following more general lemma. We actually show that from a k-colouring of G we can build a deterministic automaton with k states for L_G. This also allows us to construct the automaton $\mathcal{A}_G$ by applying this lemma on a trivial $|V|$-colouring.

▶ **Lemma 34.** *For all graph $G = (V, E)$ and $k \in \mathbb{N}$, if G is k-colourable then there exists a complete deterministic generalised Büchi automaton $\mathcal{B}$ with k states which recognises L_G.*

Proof sketch. Suppose G is k-colourable, let $c : V \to \{1, \dots, k\}$ be a k-colouring of G. We define the deterministic generalised Büchi automaton $\mathcal{B}$ as follows:

- The set of states is $Q = \{1, \dots, k\}$, we pick any state as the initial one.
- For $q \in Q$ and $v \in \Sigma = V$, the v-transition from q is $q \xrightarrow{v} c(v)$.
- The set of output colours is V, hence the output alphabet is $\Gamma = 2^V$.
- If $q \neq c(v)$, the transition $q \xrightarrow{v} c(v)$ is coloured with $V \setminus N[v]$. Transitions of the form $c(v) \xrightarrow{v} c(v)$ are coloured with $V \setminus n(v)$.

It is then quite straightforward to show that a word is accepted by $\mathcal{B}$ if and only if for each v it goes infinitely many times through the v-loop on $c(v)$ or sees infinitely many times vertices outside of $N[v]$. The structure of the automaton ensures that those words are exactly the ones in L_v. In particular, the fact that $c(u) \neq c(v)$ for all neighbours u and v implies that we cannot go through the v loop on $c(v)$ without reading a v or a non-neighbour of v just before. Figure 3 shows an example of this construction. ◀

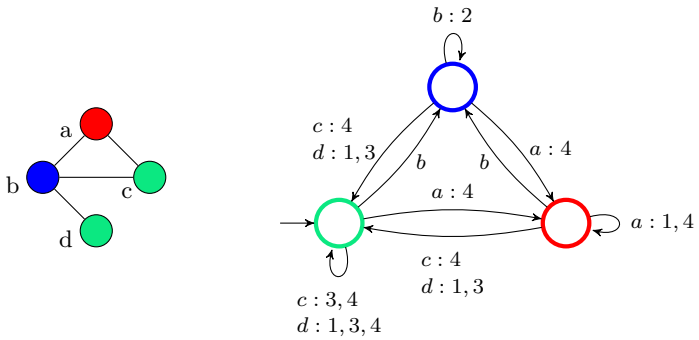


Figure 3 A graph with a 3-colouring, and the corresponding automaton as defined in Lemma 34. The output colours a, b, c, d have been replaced by $1, 2, 3, 4$ for readability.

6 Conclusion

We believe that one of the key novel insights of this work is to compare the complexity of the minimisation of HD generalised coBüchi automata (polynomial) with both HD generalised Büchi automata and deterministic models (NP-complete). For history-deterministic and deterministic Büchi automata the minimisation problem is still open; our results are an important step in this direction, and seem to indicate that the polynomial-time minimisation algorithm for the HD coBüchi case will not extend to the Büchi or the deterministic case.

References

1 Bader Abu Radi and Orna Kupferman. Minimizing GFG transition-based automata. In *ICALP*, volume 132 of *LIPIcs*, pages 100:1–100:16, 2019. doi:10.4230/LIPIcs.ICALP.2019.100.

2 Bader Abu Radi and Orna Kupferman. Minimization and canonization of GFG transition-based automata. *Log. Methods Comput. Sci.*, 18(3), 2022. doi:10.46298/lmcs-18(3:16)2022.

3 Souheib Baarir and Alexandre Duret-Lutz. Mechanizing the minimization of deterministic generalized Büchi automata. In *FORTE*, volume 8461 of *Lecture Notes in Computer Science*, pages 266–283, 2014. doi:10.1007/978-3-662-43613-4_17.

4 Marc Bagnol and Denis Kuperberg. Büchi good-for-games automata are efficiently recognizable. In *FSTTCS*, page 16, 2018. doi:10.4230/LIPICS.FSTTCS.2018.16.

5 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

6 Frantisek Blahoudek, Alexandre Duret-Lutz, and Jan Strejcek. Seminator 2 can complement generalized Büchi automata via improved semi-determinization. In *CAV*, volume 12225 of *Lecture Notes in Computer Science*, pages 15–27, 2020. doi:10.1007/978-3-030-53291-8_2.

7 Udi Boker, Denis Kuperberg, Orna Kupferman, and Michal Skrzypczak. Nondeterminism in the presence of a diverse or unknown future. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013*, volume 7966 of *Lecture Notes in Computer Science*, pages 89–100. Springer, 2013. doi:10.1007/978-3-642-39212-2_11.

8 Udi Boker, Denis Kuperberg, Karoliina Lehtinen, and Michał Skrzypczak. On succinctness and recognisability of alternating good-for-games automata. *CoRR*, abs/2002.07278, 2020. arXiv:2002.07278.

9 Udi Boker and Karoliina Lehtinen. Good for games automata: From nondeterminism to alternation. In *CONCUR*, volume 140, pages 19:1–19:16, 2019. doi:10.4230/LIPIcs.CONCUR.2019.19.

10 Udi Boker and Karoliina Lehtinen. History determinism vs. good for gameness in quantitative automata. In *FSTTCS*, volume 213, pages 38:1–38:20, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.38.

11 Udi Boker and Karoliina Lehtinen. When a little nondeterminism goes a long way: An introduction to history-determinism. *ACM SIGLOG News*, 10(1):24–51, 2023. doi:10.1145/3584676.3584682.

12 Patricia Bouyer, Antonio Casares, Mickael Randour, and Pierre Vandenhove. Half-positional objectives recognized by deterministic Büchi automata. In *CONCUR*, volume 243, pages 20:1–20:18, 2022. doi:10.4230/LIPIcs.CONCUR.2022.20.

13 J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969. URL: http://www.jstor.org/stable/1994916.

14 J. Richard Büchi. On a decision method in restricted second order arithmetic. *Proc. Internat. Congr. on Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

15 Antonio Casares. On the minimisation of transition-based Rabin automata and the chromatic memory requirements of Muller conditions. In *CSL*, volume 216, pages 12:1–12:17, 2022. doi:10.4230/LIPIcs.CSL.2022.12.

16 Antonio Casares. *Structural properties of automata over infinite words and memory for games (Propriétés structurelles des automates sur les mots infinis et mémoire pour les jeux)*. PhD thesis, Université de Bordeaux, France, 2023. URL: https://theses.hal.science/tel-04314678.

17 Antonio Casares, Thomas Colcombet, Nathanaël Fijalkow, and Karoliina Lehtinen. From Muller to Parity and Rabin Automata: Optimal Transformations Preserving (History) Determinism. *TheoretiCS*, Volume 3, April 2024. doi:10.46298/theoretics.24.12.

18 Antonio Casares, Thomas Colcombet, and Karoliina Lehtinen. On the size of good-for-games Rabin automata and its link with the memory in Muller games. In *ICALP*, volume 229, pages 117:1–117:20, 2022. doi:10.4230/LIPIcs.ICALP.2022.117.

19 Antonio Casares and Corto Mascle. The complexity of simplifying ω-automata through the alternating cycle decomposition. *CoRR*, abs/2401.03811, 2024. arXiv:2401.03811, doi:10.48550/arXiv.2401.03811.

20 Thomas Colcombet. The theory of stabilisation monoids and regular cost functions. In *ICALP*, pages 139–150, 2009. doi:10.1007/978-3-642-02930-1_12.

21 Costas Courcoubetis, Moshe Y. Vardi, Pierre Wolper, and Mihalis Yannakakis. Memory-efficient algorithms for the verification of temporal properties. *Formal Methods Syst. Des.*, 1(2/3):275–288, 1992. doi:10.1007/BF00121128.

22 Alexandre Duret-Lutz, Alexandre Lewkowicz, Amaury Fauchille, Thibaud Michaud, Etienne Renault, and Laurent Xu. Spot 2.0 - A framework for LTL and ω-automata manipulation. In *ATVA*, volume 9938 of *Lecture Notes in Computer Science*, pages 122–129, 2016. doi:10.1007/978-3-319-46520-3_8.

23 Rüdiger Ehlers. Minimising deterministic Büchi automata precisely using SAT solving. In *Theory and Applications of Satisfiability Testing - SAT*, volume 6175 of *Lecture Notes in Computer Science*, pages 326–332, 2010. doi:10.1007/978-3-642-14186-7_28.

24 Javier Esparza, Orna Kupferman, and Moshe Y. Vardi. Verification. In Jean-Éric Pin, editor, *Handbook of Automata Theory*, pages 1415–1456. European Mathematical Society Publishing House, Zürich, Switzerland, 2021. doi:10.4171/AUTOMATA-2/16.

25 Rob Gerth, Doron A. Peled, Moshe Y. Vardi, and Pierre Wolper. Simple on-the-fly automatic verification of linear temporal logic. In *IFIP*, volume 38, pages 3–18, 1995.

26 Thomas A. Henzinger and Nir Piterman. Solving games without determinization. In *Computer Science Logic*, pages 395–410, 2006. doi:10.1007/11874683_26.

27 John E. Hopcroft. An n log n algorithm for minimizing states in a finite automaton. Technical report, Stanford University, 1971. doi:10.5555/891883.

28 Christopher Hugenroth. Zielonka DAG acceptance, regular languages over infinite words. In *DLT*, 2023.

29 Sudeep Juvekar and Nir Piterman. Minimizing generalized Büchi automata. In *CAV*, pages 45–58, 2006. doi:10.1007/11817963_7.

30 Denis Kuperberg and Michał Skrzypczak. On determinisation of good-for-games automata. In *ICALP*, pages 299–310, 2015. doi:10.1007/978-3-662-47666-6_24.

31 Orna Kupferman, Shmuel Safra, and Moshe Y. Vardi. Relating word and tree automata. In *LICS*, pages 322–332, 1996. doi:10.1109/LICS.1996.561360.

32 Karoliina Lehtinen and Martin Zimmermann. Good-for-games ω-pushdown automata. In *LICS*, pages 689–702, 2020. doi:10.1145/3373718.3394737.

33 Thibaud Michaud and Maximilien Colange. Reactive synthesis from LTL specification with Spot. In *SYNT@CAV*, Electronic Proceedings in Theoretical Computer Science, 2018.

34 Satoru Miyano and Takeshi Hayashi. Alternating finite automata on omega-words. *Theor. Comput. Sci.*, 32:321–330, 1984. doi:10.1016/0304-3975(84)90049-5.

35 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *POPL*, pages 179–190, 1989. doi:10.1145/75277.75293.

36 Etienne Renault, Alexandre Duret-Lutz, Fabrice Kordon, and Denis Poitrenaud. Three SCC-based emptiness checks for generalized Büchi automata. In *LPAR*, volume 8312 of *Lecture Notes in Computer Science*, pages 668–682, 2013. doi:10.1007/978-3-642-45221-5_44.

37 Etienne Renault, Alexandre Duret-Lutz, Fabrice Kordon, and Denis Poitrenaud. Variations on parallel explicit emptiness checks for generalized Büchi automata. *Int. J. Softw. Tools Technol. Transf.*, 19(6):653–673, 2017. doi:10.1007/S10009-016-0422-5.

38 Schmuel Safra. On the complexity of ω-automata. In *FOCS*, pages 319–327, 1988. doi:10.1109/SFCS.1988.21948.

39 Sven Schewe. Beyond hyper-minimisation—minimising DBAs and DPAs is NP-complete. In *FSTTCS*, volume 8, pages 400–411, 2010. doi:10.4230/LIPIcs.FSTTCS.2010.400.

40 Sven Schewe. Minimising Good-For-Games automata is NP-complete. In *FSTTCS*, volume 182, pages 56:1–56:13, 2020. doi:10.4230/LIPIcs.FSTTCS.2020.56.

41 Fabio Somenzi and Roderick Bloem. Efficient Büchi automata from LTL formulae. In *CAV*, volume 1855 of *Lecture Notes in Computer Science*, pages 248–263, 2000. doi:10.1007/10722167_21.

42 M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994. doi:10.1006/inco.1994.1092.

Permissive Equilibria in Multiplayer Reachability Games

Aline Goeminne
F.R.S.-FNRS & UMONS – Université de Mons, Belgium

Benjamin Monmege
Aix-Marseille Univ, CNRS, LIS, Marseille, France

Abstract

We study multi-strategies in multiplayer reachability games played on finite graphs. A multi-strategy prescribes a set of possible actions, instead of a single action as usual strategies: it represents a set of all strategies that are consistent with it. We aim for profiles of multi-strategies (a multi-strategy per player), where each profile of consistent strategies is a Nash equilibrium, or a subgame perfect equilibrium. The permissiveness of two multi-strategies can be compared with penalties, as already used in the two-player zero-sum setting by Bouyer, Duflot, Markey and Renault [3]. We show that we can decide the existence of a multi-strategy profile that is a Nash equilibrium or a subgame perfect equilibrium, while satisfying some upper-bound constraints on the penalties in PSPACE, if the upper-bound penalties are given in unary. The same holds when we search for multi-strategies where certain players are asked to win in at least one play or in all plays.

2012 ACM Subject Classification Software and its engineering → Formal methods; Theory of computation → Logic and verification; Theory of computation → Solution concepts in game theory

Keywords and phrases multiplayer reachability games, penalties, permissive equilibria

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.23

Related Version *Full Version*: `https://arxiv.org/abs/2411.13296` [14]

Funding *Aline Goeminne*: Postdoctoral Researcher of the Fonds de la Recherche Scientifique – FNRS.
Benjamin Monmege: This author was partially funded by ANR JCJC Quasy ANR-23-CE48-0008.

1 Introduction

Nowadays, computer systems are ubiquitous and increasingly complex. Errors in such systems can have dramatic consequences. This is why *model checking* provides a formal tool to ensure these systems are *correct* and meet certain *specifications*. *Synthesis*, on the other hand, allows for the construction of a correct-by-construction system model: concepts from *game theory* can be used for this purpose.

Two-player zero-sum games are commonly used to model a system interacting with its environment. In this model, the system aims to achieve a goal while the environment acts antagonistically to prevent it. This situation can be abstracted as a *game played on a graph* involving two players (the system and the environment). The graph represents the different possible configurations of the system, and an infinite path in this graph is a sequence of interactions between the system and the environment. In this model, building a correct system amounts to synthesizing a winning strategy, that is, a way for the system to play that ensures its goal is met regardless of the environment's behavior.

Unlike the purely antagonistic view of two-player zero-sum games, *multiplayer games* allow for modeling situations where the environment may have its own goals, or where the system consists of different interacting components, each with its own specification. In this context, the notion of a winning strategy is no longer appropriate, hence notions of equilibria

are studied: Nash equilibria or subgame perfect equilibria, which more adequately account for the sequential aspect of games played on graphs (avoiding non-credible threats). Intuitively, an equilibrium can be seen as a contract among players such that no player has an incentive to unilaterally change his strategy.

It is well known that different equilibria can coexist in the same game. In particular, a game may include an equilibrium where no player achieves his goal and an equilibrium where all players achieve their goals. The latter equilibrium is more *relevant* than the former. Therefore, it seems appropriate to focus on the existence and synthesis of relevant equilibria (according to certain relevance criteria).

Even if the synthesis process provides an equilibrium, its implementation may fail. This can be due to the occurrence of errors; for example, the action prescribed by the equilibrium may be unavailable. Synthesizing *robust equilibria* against such perturbations is therefore essential. To address these robustness issues, the classic notion of a player's strategy can be replaced by the notion of a *multi-strategy*: unlike a classic strategy that provides a single action at each decision point, a multi-strategy provides a subset of possible actions (see, for example, [2, 3]).

Intuitively, a multi-strategy is more *permissive* than another if the first allows more behaviors than the second. There are different ways to express this permissiveness. A qualitative view of permissiveness is studied in [2], where a multi-strategy is more permissive than another if the set of resulting plays includes those of the second multi-strategy. A quantitative view is addressed in [3] via the notion of *penalty* of multi-strategies, where a cost is associated with each edge not chosen by the multi-strategy. Thus, the penalty of a multi-strategy is the highest sum of blocked edges along a play consistent with the multi-strategy.

Related works. In [2], permissiveness in parity games (a highly expressive winning condition) is studied: considering the qualitative view of permissiveness, there does not necessarily exist a most permissive strategy. However, one exists when restricted to memoryless strategies (which always make the same decision in any given vertex of the game). By reducing to safety games, the authors show that it is possible to compute the most permissive strategy. In [3], the above-mentioned quantitative view of permissiveness is implemented. Several penalty measures and games are used, and the complexity of computing the most permissive strategies in this context is given. More general parity objectives are then studied in [5]. Recently, other methods have explored permissiveness in two-player games using templates to concisely represent multiple strategies in graph games [1]. This approach is also used in multiplayer games for the synthesis of secure equilibria [16].

Independently, different equilibria (Nash or subgame perfect) have been studied in multiplayer games to ensure a strategy profile where no player has an incentive to deviate. Several works have characterized such equilibria and studied the complexity of decision problems related to the existence of relevant equilibria. Notably, these works have focused on the study and characterization of (*i*) Nash equilibria in games where players have classical ω-regular objectives [17, 11], (*ii*) weak subgame perfect equilibria (a variant of subgame perfect equilibria) where players have classical ω-regular objectives [9] (this work also characterizes subgame perfect equilibria when the studied objectives are either qualitative reachability or safety objectives); (*iii*) subgame perfect equilibria for games with quantitative reachability objectives [10]; (*iv*) subgame perfect equilibria for games with parity objectives [6] or (*v*) mean-payoff objectives [8, 7].

Contribution. Our goal is to combine these two research directions by studying permissiveness in strategy profiles that describe equilibria (Nash or subgame perfect). In this first work, we focus on reachability games only. We study permissive strategy profiles such that all the fully described strategy profiles they contain are equilibria. The motivation is to allow greater latitude and robustness of equilibrium profiles without losing quality in the final goal of secure synthesis. With the qualitative view, as in the two-player game framework [2], it is not difficult to show that there does not necessarily exist most permissive profiles that are Nash equilibria (or subgame perfect equilibria). We will thus consider a quantitative view of permissiveness similar to the penalty measures introduced in [3] for two-player games. We obtain a characterization based on trees, and decision algorithms with penalties bounded by a given threshold in polynomial space with respect to the size of the game and the maximal penalty bound (if this is encoded in unary). We also solve the problem of synthesis of robust and relevant equilibria, where the relevance is the constraint that all derived equilibria ensure that all players in a fixed subset satisfy their objective (*strongly winning*), or that at least one derived equilibrium ensures this guarantee (*weakly winning*).

All missing proofs can be found in the long version of this article [14].

2 Multiplayer reachability games

A *(multiplayer) reachability games* is a tuple $(\mathrm{N}, V, (V_i)_{i\in\mathrm{N}}, E, (\mathrm{F}_i)_{i\in\mathrm{N}}, v_0)$, that we denote $(\mathcal{G}, v_0)$ to emphasize the v_0 component, where $\mathrm{N} = \{1, \dots, n\}$ is a finite set of n players, (V, E) is a finite directed graph without deadlocks (for all $v \in V$, there exists $v' \in V$ such that $(v, v') \in E$), $(V_i)_{i\in\mathrm{N}}$ is a partition of V between the players, $\mathrm{F}_i \subseteq V$ is the set of target vertices, called *target set*, of player $i \in \mathrm{N}$, and v_0 is an initial vertex. Given a vertex $v \in V$, we let $\mathrm{Succ}(v) = \{v' \in V \mid (v, v') \in E\}$ be the set of all successors of v.

A *play* in $\mathcal{G}$ is an infinite sequence of vertices consistent with the graph structure, *i.e.,* if $\rho = \rho_0\rho_1\cdots$ is a play, then for all $k \in \mathbb{N}$, $\rho_k \in V$ and $(\rho_k, \rho_{k+1}) \in E$. The set of plays is denoted by Plays, while $\mathrm{Plays}(v)$ denotes the set of plays beginning in v. Given a play $\rho = \rho_0\rho_1\cdots$ and $k \in \mathbb{N}$, $\rho_{\geq k}$ is the suffix $\rho_k\rho_{k+1}\cdots$ of ρ.

For each player $i \in \mathrm{N}$, we let Gain_i be the *gain function* that associates with each play the value 1 if the play is winning for player i, 0 if it is losing. For a reachability game as above, we have $\mathrm{Gain}_i(\rho) = 1$ iff player i reaches his target set in ρ, *i.e.,* $\rho = \rho_0\rho_1\cdots$ and there exists $k \in \mathbb{N}$ with $\rho_k \in \mathrm{F}_i$. **In the rest of this article, $(\mathcal{G}, v_0)$ will always denote a reachability game associated with these gain functions.**

A *history* is a finite sequence of vertices $h = h_0h_1\cdots h_k$ with $k \in \mathbb{N}$ defined similarly. The set of histories is denoted by Hist, while $\mathrm{Hist}(v)$ denotes the set of histories beginning in v. For all $i \in \mathrm{N}$, we write Hist_i to denote the set of histories ending in a vertex owned by player i. If $h = h_0\cdots h_k$ with $k \in \mathbb{N}$ is a history, $\mathrm{Last}(h)$ denotes the last vertex h_k, while $|h|$ denotes its length k. Given a history $h = h_0\cdots h_k$, $\mathrm{Visit}(h) = \{i \in \mathrm{N} \mid \exists 1 \leq \ell \leq k\ h_\ell \in F_i\}$ is the set of players who visit their target set along h.

A *strategy* of player i is a function $\sigma_i \colon \mathrm{Hist}_i(v_0) \to V$ that assigns to each history $hv \in \mathrm{Hist}_i(v_0)$ a vertex v' such that $(v, v') \in E$. A play $\rho = \rho_0\rho_1\cdots$ is consistent with a strategy σ_i if for all $\rho_k \in V_i$, $\sigma_i(\rho_0\cdots\rho_k) = \rho_{k+1}$. A *strategy profile* is a tuple $\sigma = (\sigma_i)_{i\in\mathrm{N}}$ of strategies, one per player: there is a unique play from v_0 which is consistent with each strategy σ_i, and we call this play the *outcome* of σ, denoted by $\langle\sigma\rangle_{v_0}$. To highlight the role of player i, we sometimes write $\sigma = (\sigma_i, \sigma_{-i})$ where σ_{-i} denotes the strategy profile of the players other than player i.

The strategy profile σ is a *Nash equilibrium* (NE) in $(\mathcal{G}, v_0)$ if no player has an incentive to deviate unilaterally from his strategy to increase his gain, *i.e.,* if for all players $i \in \mathrm{N}$ and all strategies σ'_i of player i, $\mathrm{Gain}_i(\langle\sigma\rangle_{v_0}) \geq \mathrm{Gain}_i(\langle\sigma'_i, \sigma_{-i}\rangle_{v_0})$.

The concept of *subgame perfect equilibrium* (SPE) takes more into account the sequential nature of games played on graphs by avoiding non-credible threat, a well-known weakness of NEs in this setting. Informally, a strategy profile is an SPE if it is an NE in all subgames. Given a history $hv \in \mathrm{Hist}(v_0)$, the *subgame* $(\mathcal{G}_{\restriction h}, v)$ is obtained from $\mathcal{G}$ by changing the initial vertex to v, and by considering the gain functions $(\mathrm{Gain}_{i\restriction h})_{i\in\mathrm{N}}$ taking into account the players that have won in history h: we thus write, for each $i \in \mathrm{N}$, $\mathrm{Gain}_{i\restriction h}(\rho) = \mathrm{Gain}_i(h\rho)$ for all $\rho \in \mathrm{Plays}(v)$. Moreover, if σ_i is a strategy of player i in $\mathcal{G}$, then $\sigma_{i\restriction h}$ is the strategy of player i in the subgame $(\mathcal{G}_{\restriction h}, v)$ such that for all $h' \in \mathrm{Hist}_i(v)$, $\sigma_{i\restriction h}(h') = \sigma_i(hh')$. In the same way, from a strategy profile σ in $\mathcal{G}$, we can derive a strategy profile $\sigma_{\restriction h}$ in $(\mathcal{G}_{\restriction h}, v)$. We now define formally the concept of SPEs: a strategy profile σ is an SPE in $\mathcal{G}$ if for all $i \in \mathrm{N}$, for all $hv \in \mathrm{Hist}_i(v_0)$, $\sigma_{\restriction h}$ is an NE in $(\mathcal{G}_{\restriction h}, v)$. Notice that an SPE is an NE and that there always exists an SPE (and thus an NE) in a reachability game [17].

3 Permissiveness in strategies

Our goal is to allow for some permissiveness in strategies of all players, *i.e.,* being able to underspecify the strategies of the players, while maintaining that they describe an NE or an SPE.

A *multi-strategy* of player i is a function $\Theta_i \colon \mathrm{Hist}_i(v_0) \to 2^V \setminus \{\emptyset\}$ that assigns to each history $hv \in \mathrm{Hist}_i(v_0)$ a non-empty set of vertices $\mathrm{A} \subseteq V$ such that for all $v' \in \mathrm{A}$, $(v, v') \in E$. Notice that a strategy σ_i can be seen as a multi-strategy Θ_i where, for all $hv \in \mathrm{Hist}_i(v_0)$, $\Theta_i(hv)$ is the singleton $\{\sigma_i(hv)\}$. A *multi-strategy profile* $\Theta = (\Theta_i)_{i\in\mathrm{N}}$ is a tuple of multi-strategies, one per player.

Unlike strategies, when we fix a game $\mathcal{G}$ and a multi-strategy profile Θ, there exist several plays beginning in v_0 that are consistent with all the multi-strategies Θ_i. To describe them, we say that a strategy σ_i is *consistent* with a multi-strategy Θ_i, written $\sigma_i \lesssim \Theta_i$ if for all $hv \in \mathrm{Hist}_i(v_0)$, $\sigma_i(hv) \in \Theta_i(hv)$. We extend this notation to profiles of strategies, as expected. Then, we let $\langle\Theta\rangle_{v_0}$ be the set of plays $\langle\sigma\rangle_{v_0}$ for all profiles σ of strategies consistent with the multi-strategy Θ. We call this set the *outcomes of* Θ. We also let $\langle\Theta\rangle^{\mathrm{H}}_{v_0}$ be the set of histories consistent with the multi-strategy Θ, *i.e.,* the finite prefixes of plays in $\langle\Theta\rangle_{v_0}$.

Our goal is to compute profiles of multi-strategies such that all profiles of consistent strategies are NEs or SPEs: such profiles of multi-strategies are called permissive NEs or permissive SPEs. By the existence of NEs and SPEs in reachability games, we straightforwardly obtain the existence of permissive NEs and permissive SPEs. We thus want to study *most permissive* NEs or SPEs, *i.e.,* profiles of multi-strategies that are permissive NEs or SPEs, and such that no "more permissive" multi-strategies are still permissive NEs or SPEs.

The natural first attempt would be to look for a notion of "more permissive" that is set-theoretic, with respect to a given solution concept. We would thus say that a profile of multi-strategies Θ is at least as permissive as a profile of multi-strategies Θ' if for all $i \in \mathrm{N}$, for all histories $h \in \mathrm{Hist}_i(v_0)$, $\Theta_i(h) \supseteq \Theta'_i(h)$. Then, Θ would be more permissive than Θ' if it is at least as permissive, while being different (for at most one history). Finally, Θ would be a *most permissive* NE or SPE if it is a permissive NE or SPE, respectively, and no permissive NE or SPE, respectively, is more permissive than Θ.

This natural definition is very problematic in the realm of reachability games (as already noticed in the context of winning strategies in parity games by [2]) where no most permissive NE or SPE could exist, as demonstrated by the game in Figure 1.

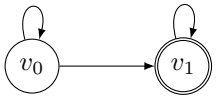


Figure 1 In this game, player 1 owns all vertices and wants to reach v_1. For all $k \in \mathbb{N}$, we define the multi-strategy Θ_1^k such that for all $h \in \mathrm{Hist}(v_0)$, $\Theta_1^k(h) = \{v_0, v_1\}$ if $\mathrm{Last}(h) = v_0$ and $|\{n \in \mathbb{N} \mid h_n = v_0\}| \leq k$, and $\Theta_1^k(h) = \{v_1\}$ otherwise. We have that for all $k \in \mathbb{N}$, for all $\sigma_1 \lesssim \Theta_1^k$, $\mathrm{Gain}_1(\langle\sigma_1\rangle_{v_0}) = 1$ (and thus Θ_1^k is a permissive SPE), but for all $k \in \mathbb{N}$, $\langle\Theta_1^k\rangle_{v_0} \subseteq \langle\Theta_1^{k+1}\rangle_{v_0}$.

We thus propose another way to measure the permissiveness of a multi-strategy, inspired by the definition of penalty used in [3] to describe permissive winning strategies in two-player games. To define the notion of penalty in our context, we equip the game with a function $w\colon E \to \mathbb{N}$ assigning a non-negative weight to each edge: if unspecified, we will consider that every edge has weight 1. The player who owns the vertex at the source of an edge e will pay the penalty $w(e)$ if he decides to not include the edge e in his multi-strategy. All penalties are then counted additively. Formally, for a multi-strategy profile Θ, we first define for each player $i \in \mathrm{N}$ the *penalty of player i w.r.t.* Θ in a play $\rho = \rho_0\rho_1\cdots$ by induction on the length of its prefixes:

- $\mathrm{Penalty}_i^\Theta(\varepsilon) = 0$ where ε denotes the empty prefix;
- for $h = \rho_0\cdots\rho_k$, $\mathrm{Penalty}_i^\Theta(hv) = \begin{cases} \mathrm{Penalty}_i^\Theta(h) + \sum_{v' \in \mathrm{Succ}(v)\setminus\Theta_i(hv)} w(v,v') & \text{if } v \in V_i \\ \mathrm{Penalty}_i^\Theta(h) & \text{otherwise} \end{cases}$;
- $\mathrm{Penalty}_i^\Theta(\rho) = \lim_{k\to+\infty} \mathrm{Penalty}_i^\Theta(\rho_0\cdots\rho_k)$: this limit exists (it may be equal to $+\infty$) since $(\mathrm{Penalty}_i^\Theta(\rho_0\cdots\rho_k))_k$ is a non-decreasing sequence of natural numbers.

There are several ways to associate a penalty with a multi-strategy profile Θ, depending on how we take into account the non-determinism offered in the multi-strategies. A first choice consists in considering a worst-case scenario in the outcomes (without considering the possible deviations). A second choice consists in considering only the deviations of one player, *i.e.,* to consider that the retaliation of other players with respect to the deviation of a player will count in the final penalty. It is then possible to combine both types of penalties, though we will treat them separately in the rest of this article.

▶ **Definition 1** (Penalties). *Let Θ be a multi-strategy profile in $(\mathcal{G}, v_0)$. The* main penalty *and* retaliation penalty *of player i with respect to Θ are defined respectively as*

$$\mathrm{MPenalty}_i(\Theta, v_0) = \sup_{\rho \in \langle\Theta\rangle_{v_0}} \mathrm{Penalty}_i^\Theta(\rho)$$

$$\mathrm{RPenalty}_i(\Theta, v_0) = \sup_{hv \in \mathrm{Hist}_i(v_0)\setminus\langle\Theta\rangle_{v_0}^{\mathrm{H}}} \sup\{\mathrm{Penalty}_i^\Theta(\rho) \mid \rho \in \langle\Theta_{\restriction hv}\rangle_v\}$$

If there are no histories hv in $\mathrm{Hist}_i(v_0) \setminus \langle\Theta\rangle_{v_0}^{\mathrm{H}}$, we let $\mathrm{RPenalty}_i(\Theta, v_0) = 0$.

The existence of a multi-strategy profile which satisfies some upper-bounds on penalties does not provide any certainty about the satisfaction of the reachability objectives of the players. For this reason, we also consider multi-strategy profiles that satisfy some properties on the set of players who satisfy their objective. Let Win be a subset of players and Θ be a multi-strategy profile. Then, Θ is said *weakly winning* if **there exists** a strategy profile σ which is consistent with Θ and such its outcome is winning for all players in Win. Similarly, Θ is said *strongly winning* if **for each** strategy profile σ which is consistent with Θ, its outcome is winning for all players in Win.

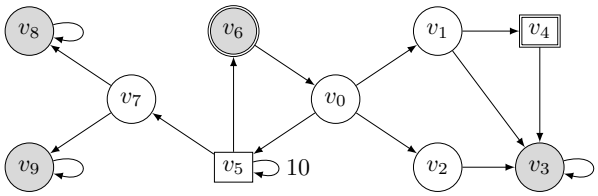


Figure 2 An example of a reachability game where player 1 (resp. player 2) owns circle (resp. rectangle) vertices. The initial vertex is v_0. Target vertices $F_1 = \{v_3, v_6, v_8, v_9\}$ of player 1 and $F_2 = \{v_4, v_6\}$ of player 2 are drawn with gray vertices and double-bordered vertices respectively.

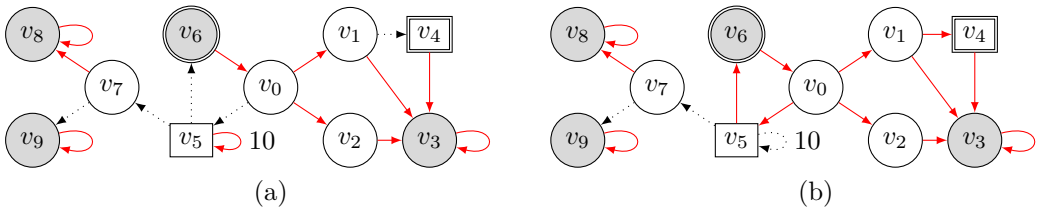


Figure 3 Examples of permissive equilibria: (a) a permissive NE and (b) a permissive SPE.

▶ **Definition 2** (Weakly and strongly winning). *Given a subset of player* Win $\subseteq$ N *and a multi-strategy profile* Θ*,*

- Θ *is said* weakly winning with respect to Win *if there exists a strategy profile* σ *such that* $\sigma \lesssim \Theta$ *and for all* $i \in$ Win, $\mathrm{Gain}_i(\langle\sigma\rangle_{v_0}) = 1$.
- Θ *is said* strongly winning with respect to Win *if for all strategy profiles* σ *such that* $\sigma \lesssim \Theta$*, we have that for all* $i \in$ Win, $\mathrm{Gain}_i(\langle\sigma\rangle_{v_0}) = 1$.

▶ **Example 3.** An example of a reachability game with two players is depicted in Figure 2. The edge labelled with 10 corresponds to the penalty if player 2 decides not to allow this edge: all other penalties are set to 1 by default. A multi-strategy is represented with red edges (black dotted edges are thus the ones that are not selected in the multi-strategy) in Figure 3(a).[1] All strategy profiles that are consistent with this multi-strategy depend on the choice of successor for v_0 among $\{v_1, v_2\}$. It is indeed a permissive NE since the consistent strategies are NEs: player 1 has no interest in deviating from either v_1 or v_2 in v_0, since all strategies lead to plays where he visits his target set, while going to v_5 make him lose. It has a main penalty of 2 for player 1 and 0 for player 2. Player 1 can do slightly better by allowing the edge (v_1, v_4) in the multi-strategy: this remains a permissive NE (now player 2 wins in certain plays, but he is left with no real choices to make), and player 1 now gets a main penalty of 1. This modified permissive NE is strongly winning w.r.t. $\{1\}$, and weakly winning w.r.t. $\{1, 2\}$. It is not a permissive SPE since player 2 has a profitable deviation from v_5 by going to v_6 where he wins. A permissive SPE is depicted in Figure 3(b), that is strongly winning w.r.t. $\{1\}$, but only weakly winning w.r.t $\{1, 2\}$. Player 2 has a main penalty of 11 (because he cuts edges (v_5, v_5) and (v_5, v_7)), while player 1 has a retaliation penalty of 1 (because he cuts edge (v_7, v_9)). If we want a permissive SPE that is strongly winning w.r.t. $\{1, 2\}$, we need to increase the main penalty of player 1 to 2 by removing edges (v_1, v_3) and (v_0, v_2). However, we may decrease to 0 the retaliation penalty of player 1 by adding the edge (v_7, v_9) (since it is equally good to him anyway).

[1] Notice that, in this example, the set of successors prescribed by multi-strategies only depends on the current vertex and not on the past history.

We now define the problems we study in the rest of the article, where we use the word "equilibrium" to either mean NE or SPE, depending on the solution concept we want to check. In all these problems, we give different penalty bounds for the main penalty and the retaliation penalty. Notice though that the bounds can be set to $+\infty$, relaxing the constraints in this case.

▶ **Problem 1** (Constrained penalty problem). *Given a reachability game* $(\mathcal{G}, v_0)$, $m \in (\mathbb{N} \cup \{\infty\})^n$ *and* $r \in (\mathbb{N} \cup \{\infty\})^n$, *does there exist a permissive equilibrium* Θ *in* $(\mathcal{G}, v_0)$ *such that for all* $i \in \mathrm{N}$, $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$ *and* $\mathrm{RPenalty}_i(\Theta, v_0) \leq r_i$*?*

▶ **Problem 2** (Weakly winning with constrained penalty problem). *Given a reachability game* $(\mathcal{G}, v_0)$, $m \in (\mathbb{N} \cup \{\infty\})^n$, $r \in (\mathbb{N} \cup \{\infty\})^n$ *and* $\mathrm{Win} \subseteq \mathrm{N}$, *does there exist a permissive equilibrium* Θ *in* $(\mathcal{G}, v_0)$ *such that* (i) *for all* $i \in \mathrm{N}$, $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$ *and* $\mathrm{RPenalty}_i(\Theta, v_0) \leq r_i$ *and* (ii) Θ *is weakly winning w.r.t.* Win*?*

▶ **Problem 3** (Strongly winning with constrained penalty problem). *Given a reachability game* $(\mathcal{G}, v_0)$, $m \in (\mathbb{N} \cup \{\infty\})^n$, $r \in (\mathbb{N} \cup \{\infty\})^n$ *and* $\mathrm{Win} \subseteq \mathrm{N}$, *does there exist a permissive equilibrium* Θ *in* $(\mathcal{G}, v_0)$ *such that* (i) *for all* $i \in \mathrm{N}$, $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$ *and* $\mathrm{RPenalty}_i(\Theta, v_0) \leq r_i$ *and* (ii) Θ *is strongly winning w.r.t.* Win*?*

We show in the rest of this article that all these problems, for NEs and SPEs, are decidable in PSPACE, if the upper-bound penalties are encoded in unary. To do so, we characterize the permissive equilibria in the various problems in Section 4. In Section 5, we then show that tree-like witnesses can be found if the according permissive equilibria exist. These witnesses have a height bounded by a polynomial depending on the size of the game and the largest upper-bound on penalties. We use these witnesses to obtain the PSPACE decision procedures.

4 Characterizations of permissive equilibria

We now characterize permissive equilibria of the reachability game $(\mathcal{G}, v_0)$. This is a first step towards their computation in the next section. We provide a characterization for permissive NEs in Section 4.1 and one for permissive SPEs in Section 4.2. These characterizations are inspired by existing ones for classical NEs (resp. SPEs) [11, 9]. The latter rely on properties that a play (resp. a set of plays) must satisfy in order to be the outcome of an NE (resp. the set of subgame outcomes of an SPE). However, the outcomes of permissive equilibria are a set of plays and not a simple play. For that reason, the characterizations of permissive equilibria employ trees that we first formally define.

Trees. We call *tree over* $\mathcal{G}$ *rooted at* v (for some $v \in V$) any subset $\mathcal{T}$ of non-empty histories of $\mathcal{G}$ that contains v and such that if $hu \in \mathcal{T}$ then $h \in \mathcal{T}$. All $h \in \mathcal{T}$ are called *nodes* of the tree, the particular node v is called the *root* of the tree, and for all $hu \in \mathcal{T}$, h is called the *parent* of hu, and hu a *child* of h.

As for histories in an arena, for all $hu \in \mathcal{T}$, we let $\mathrm{Last}(hu) = u$. The *depth* of a node $h \in \mathcal{T}$, written $\mathrm{depth}(h)$, is equal to $|h|$ and its *height*, denoted by $\mathrm{height}(h)$, is given by $\sup\{|\,\mathrm{Last}(h)h'|\,\mid h' \in \mathrm{Hist} \text{ and } hh' \in \mathcal{T}\}$. The height of the tree corresponds to the height of its root. A node $h \in \mathcal{T}$ is called a *leaf* if $\mathrm{height}(h) = 0$.

We denote by $\mathcal{T}_{\restriction hu}$, the subtree of $\mathcal{T}$ rooted at u for some $hu \in \mathcal{T}$, that is the set of non-empty histories $h' \in \mathrm{Hist}(u)$ such that $hh' \in \mathcal{T}$.

A (finite or infinite) *branch* of the tree is a maximal (finite or infinite) sequence of nodes $h_0h_1\cdots$ such that for all $k \in \mathbb{N}$, h_k is the parent of h_{k+1}. Finally, we denote by $\mathcal{T}^\infty$ the set of plays in $\mathcal{G}$ represented by infinite branches in $\mathcal{T}$, *i.e.,*

$$\mathcal{T}^\infty = \{\rho_0\rho_1\cdots \in \text{Plays} \mid \text{there exists a branch } h_0h_1\cdots \in \mathcal{T} \text{ st. } \forall k \in \mathbb{N},\ \rho_k = \text{Last}(h_k)\}$$

In what follows, we consider outcomes of multi-strategies as trees. Indeed, given a multi-strategy Θ, $\langle\Theta\rangle^{\text{H}}_{v_0}$ can be seen as a tree $\mathcal{T}$ over $\mathcal{G}$ rooted at v_0 and $\langle\Theta\rangle_{v_0}$ corresponds to $\mathcal{T}^\infty$. In particular, penalties can also be defined on trees, mimicking the definition for profiles of multi-strategies. The penalty of a tree $\mathcal{T}$ for a player i, denoted by $\text{Penalty}_i(\mathcal{T})$, is the maximal penalty of a branch of $\mathcal{T}$, the penalty of a branch being equal to the penalty of the associated play ρ w.r.t. any profile of multi-strategies that is consistent with the choices appearing in $\mathcal{T}$ along the play ρ. Formally, let $\mathcal{T}$ be a tree and $i \in \text{N}$ be a player. For each $hv = v_1\cdots v_kv \in \mathcal{T}$, we define $\text{Blocked}(h) = \{u \in \text{Succ}(v_k) \mid hu \notin \mathcal{T}\}$ as the set of blocked successors of h in $\mathcal{T}$ and

$$\text{Penalty}_i(hv) = \begin{cases} 0 & \text{if } h = \varepsilon \\ \text{Penalty}_i(h) + \sum_{u\in\text{Blocked}(h)} w(v_k,u) & \text{if } v_k \in V_i \\ \text{Penalty}_i(h) & \text{otherwise.} \end{cases}$$

Moreover, for all plays $\rho = \rho_0\rho_1\cdots \in \mathcal{T}^\infty$, we let $\text{Penalty}_i(\rho) = \lim_{k\to+\infty} \text{Penalty}_i(\rho_0\cdots\rho_k)$. Thus, the penalty of a tree $\mathcal{T}$ for a player $i \in \text{N}$ is naturally defined as:

$$\text{Penalty}_i(\mathcal{T}) = \sup\{\text{Penalty}_i(\rho) \mid \rho \in \mathcal{T}^\infty\}.$$

4.1 Characterization of permissive Nash equilibria

In order to characterize permissive Nash equilibria, we start by defining *good* trees, by checking two conditions. The first one, called *resistance to internal deviations*, means that at any node h of the tree such that $\text{Last}(h)$ belongs to player i, if h has at least two children, the plays starting with h are either all losing, or all winning, for player i. The second one, called *resistance to external deviations*, means that at any node hu of the tree with u belonging to player i, if player i has the possibility to play to a successor u' not in the tree from which he has a winning strategy, then all plays in the subtree from hu must be winning for player i.

▶ **Definition 4.** *Let $\mathcal{T}$ be a tree over $(\mathcal{G}, v_0)$.*

1. *Given a subset of players* $\text{D} \subseteq \text{N}$*, the tree $\mathcal{T}$ is* D-resistant to internal deviations *if for all $i \in \text{D}$ and for all $hv \in \mathcal{T}$ such that $v \in V_i$ and $|\{hvv' \in \mathcal{T} \mid v' \in V\}| \geq 2$, we have that for all $\rho, \rho' \in \mathcal{T}^\infty_{\restriction hv}$, $\text{Gain}_i(h\rho) = \text{Gain}_i(h\rho')$. If $\text{D} = \text{N}$, we simply say that $\mathcal{T}$ is* resistant to internal deviations*.*
2. *The tree $\mathcal{T}$ is* resistant to external deviations *if for all $hu \in \mathcal{T}$ with $u \in V_i$ and $i \notin \text{Visit}(hu)$, if there exists $u' \in \text{Succ}(u)$ such that $huu' \notin \mathcal{T}$ and player i has a winning strategy from u' (against the coalition of the other players), then for all plays $\rho \in \mathcal{T}^\infty_{\restriction hu}$, $\text{Gain}_i(\rho) = 1$.*
3. *The tree $\mathcal{T}$ is* good *if it is resistant to internal and external deviations.*

The resistance to internal and external deviations leads to the characterization of outcomes of permissive NEs (Theorem 5): given a good tree $\mathcal{T}$, there exists a permissive NE such that its outcomes are the plays corresponding to the infinite branches of $\mathcal{T}$ iff $\mathcal{T}$ is good.

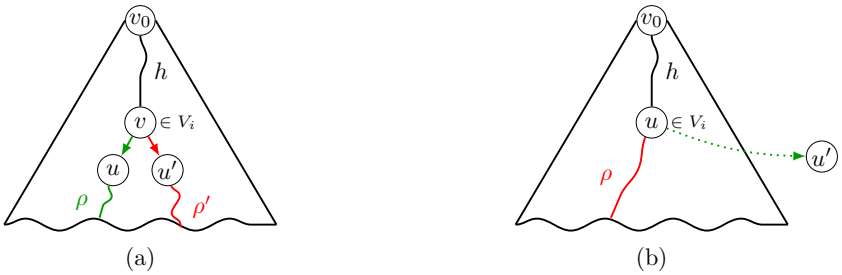


Figure 4 Examples of trees that do not respect: (a) the resistance to internal deviations since $\mathrm{Gain}_i(\rho') = 0$ but $\mathrm{Gain}_i(\rho) = 1$; (b) the resistance to external deviations since $\mathrm{Gain}_i(\rho) = 0$ but Player i can win from u'.

▶ **Theorem 5.** *Let $\mathcal{T}$ be a tree over $(\mathcal{G}, v_0)$ rooted at v_0. The following assertions are equivalent:*

1. *There exists a permissive NE Θ in $(\mathcal{G}, v_0)$ such that $\langle\Theta\rangle^{\mathrm{H}}_{v_0} = \mathcal{T}$;*
2. *The tree $\mathcal{T}$ is good.*

▶ Remark 6. For all multi-strategies Θ, and all players $i \in \mathrm{N}$, the penalty $\mathrm{MPenalty}_i(\Theta, v_0)$ is equal to the penalty of player i in the good tree $\langle\Theta\rangle^{\mathrm{H}}_{v_0}$, *i.e.,* $\mathrm{Penalty}_i(\langle\Theta\rangle^{\mathrm{H}}_{v_0})$. The construction of Theorem 5 thus also preserves the main penalties.

Proof sketch. For $(1 \Rightarrow 2)$ let us assume that Θ is a permissive NE and that $\langle\Theta\rangle^{\mathrm{H}}_{v_0} = \mathcal{T}$. We have to prove that $\mathcal{T}$ is good. If $\mathcal{T}$ is not resistant to internal deviations that means that from some vertex v there exists two plays ρ, crossing u, and ρ', crossing $u' \neq u$, such that: ρ is winning for player i and ρ' is losing for player i, see Figure 4(a). In particular, we can build a strategy profile σ consistent with Θ such that $\langle\sigma_{\restriction h}\rangle_v = \rho'$ and $\langle\sigma_{\restriction hv}\rangle_u = \rho_{\geq 1}$. Meaning that player i should deviate by choosing u instead of u' from v, meaning that σ is not an NE and Θ is not a permissive NE. If $\mathcal{T}$ is not resistant to external deviations, that means that from some vertex u of player i there exists a play ρ such that $\mathrm{Gain}_i(\rho) = 0$ and u' a successor of u outside $\mathcal{T}$ from which player i can win, see Figure 4(b). Thus we can build a strategy profile σ consistent with Θ such that $\langle\sigma\rangle_{v_0} = h\rho$. In this way, player i should choose to go in u' and then follow a winning strategy meaning that σ is not an NE and Θ not a permissive NE.

For $(2 \Rightarrow 1)$, let us assume that $\mathcal{T}$ is a good tree. We build a permissive NE Θ such that its outcomes are the plays corresponding to the infinite branches of $\mathcal{T}$. Additionally, if a player i deviates from $\mathcal{T}$, the coalition of the other players plays its retaliation[2] strategy to prevent player i from deviating. ◀

4.2 Characterization of permissive subgame perfect equilibria

Permissive subgame perfect equilibria are intrinsically more complex than permissive Nash equilibria. Thus their characterization cannot only rely on the outcomes from the initial vertex, it should also take into account the outcomes in all subgames. This is the reason why, in order to deal with a compact representation of outcomes of a permissive SPE and its subgames, we introduce the notion of *forest*. Then, we generalize the definition of good trees to define *good forests* needed to characterize SPEs instead of NEs.

[2] This retaliation strategy corresponds to the winning strategy of player 2 in a two-player zero-sum reachability game in which player 1 is player i and wants to reach F_i and player 2 is the coalition of the other players and wants to avoid visiting F_i [15, Chapter 2].

Forests and penalties of forests. Trees of the forest are indexed by tuples $(i, v, \mathrm{I}) \in \mathrm{N} \times V \times 2^{\mathrm{N}}$. More precisely, we let

$$\mathcal{I} = \{(0, v_0, \mathrm{I}_0)\} \cup \{(i, v, \mathrm{I}) \in \mathrm{N} \times V \times 2^{\mathrm{N}} \mid \exists hv' \in \mathrm{Hist}_i(v_0) \text{ st. } v \in \mathrm{Succ}(v') \wedge \mathrm{I} = \mathrm{Visit}(hv'v)\}$$

where $\mathrm{I}_0 = \{i \in \mathrm{N} \mid v_0 \in F_i\}$. Apart from the special tuple $(0, v_0, \mathrm{I}_0)$, a tuple (i, v, I) represents the fact that v is a vertex played by player i and reachable from v_0, and that all players in I have already seen their target when v is reached. A forest in $(\mathcal{G}, v_0)$ is thus a set of trees $\mathcal{F} = \{\mathcal{T}_{i,v,I} \mid (i, v, \mathrm{I}) \in \mathcal{I}\}$ such that $\mathcal{T}_{i,v,\mathrm{I}}$ is a tree without leaves over $\mathcal{G}$ rooted at v. The intuition behind this object is that the tree $\mathcal{T}_{0,v_0,\mathrm{I}_0}$ represents the outcomes of a multi-strategy Θ and the other trees $\mathcal{T}_{i,v,\mathrm{I}}$ represent the outcomes of $\Theta_{\restriction hv'}$ in the subgames $(\mathcal{G}_{\restriction hv'}, v)$ for all $hv' \in \mathrm{Hist}_i(v_0)$ such that $\mathrm{Visit}(hv'v) = \mathrm{I}$.

Moreover the main (resp. retaliation) penalty of a forest $\mathcal{F}$ for a player $i \in \mathrm{N}$ are respectively given by

$$\mathrm{MPenalty}_i(\mathcal{F}) = \mathrm{Penalty}_i(\mathcal{T}_{0,v_0,\mathrm{I}_0}) \quad \text{and} \quad \mathrm{RPenalty}_i(\mathcal{F}) = \sup_{\substack{\mathcal{T}_{i,v,I} \in \mathcal{F} \\ (i,v,\mathrm{I}) \in \mathrm{Out}}} \mathrm{Penalty}_i(\mathcal{T}_{i,v,I})$$

where $\mathrm{Out} = \{(i, v, I) \in \mathcal{I} \setminus \{(0, v_0, \mathrm{I}_0)\} \mid \exists hv \in \mathrm{Hist}(v_0) \text{ st. } hv \notin \mathcal{T}_{(0,v_0,\mathrm{I}_0)} \wedge \mathrm{Last}(h) \in V_i \wedge \mathrm{I} = \mathrm{Visit}(hv)\}$ described the indices of trees in the forest that are deviations from the main tree $\mathcal{T}_{0,v_0,\mathrm{I}_0}$. If Out is empty, we let $\mathrm{RPenalty}_i(\mathcal{F}) = 0$.

Characterization. Following the same philosophy as for permissive NEs, a forest is *good* if each tree $\mathcal{T}_{i,v,\mathrm{I}}$ of the forest satisfies two properties. The first one is that $\mathcal{T}_{i,v,\mathrm{I}}$ has to be $(\mathrm{N} \setminus \mathrm{I})$-resistant to internal deviations, exactly as for permissive NEs except that we take into account players who have already visited their target set, *i.e.,* players in I. The second one, called *resistance to constrained external deviations*, means that at any node hu of the tree such that u belongs to player j, if player j has the possibility to jump to another tree $\mathcal{T}_{j,u',\mathrm{I}'}$ by playing to a successor u' not in the tree and if there exists a play in this latter tree which is winning for player j, then all plays after hu in $\mathcal{T}_{i,v,\mathrm{I}}$ have to be winning for player j.

▶ **Definition 7** (Good forest). *Let $\mathcal{F}$ be a forest in $(\mathcal{G}, v_0)$.*

1. *A tree $\mathcal{T}_{i,v,\mathrm{I}} \in \mathcal{F}$ is* resistant to constrained external deviations *if it satisfies the following property: for all $hu \in \mathcal{T}_{i,v,\mathrm{I}}$ and $j \in \mathrm{N}$ such that we have that* (i) *$u \in V_j$ and $j \notin I \cup \mathrm{Visit}(hu)$ and* (ii) *there exists $u' \in \mathrm{Succ}(u)$ such that $huu' \notin \mathcal{T}_{i,v,\mathrm{I}}$, if there exists $\rho' \in \mathcal{T}^{\infty}_{j,u',\mathrm{I}'}$, where $\mathrm{I}' = \mathrm{I} \cup \mathrm{Visit}(huu')$, such that $\mathrm{Gain}_j(\rho') = 1$, then for all $\rho \in \mathcal{T}^{\infty}_{i,v,\mathrm{I}\restriction hu}$, $\mathrm{Gain}_j(\rho) = 1$.*
2. *The forest $\mathcal{F}$ is* good *if each tree $\mathcal{T}_{i,v,\mathrm{I}} \in \mathcal{F}$ is $(\mathrm{N} \setminus \mathrm{I})$-resistant to internal deviations (see* (1) *in Definition 4) and resistant to constrained external deviations.*

Thanks to good forests, we are able to characterize the outcomes of permissive SPEs: given a good tree $\mathcal{T}^*$, there exists a permissive SPE such that its outcomes correspond to $\mathcal{T}^*$ iff there exists a good forest whose "main" tree is $\mathcal{T}^*$, *i.e.,* $\mathcal{T}_{0,v_0,\mathrm{I}_0} = \mathcal{T}^*$. With some other constraints, this also preserves strongly (resp. weakly) winning and penalty properties.

▶ **Theorem 8.** *Let $m \in (\mathbb{N} \cup \{\infty\})^n$ and $r \in (\mathbb{N} \cup \{\infty\})^n$ be upper thresholds. Let $\mathcal{T}^*$ be a tree rooted at v_0 and* $\mathrm{Win} \subseteq \mathrm{N}$ *be a set of players. The following assertions are equivalent:*

1. *There exists a permissive SPE Θ in $(\mathcal{G}, v_0)$ such that:*
   - **a.** $\langle \Theta \rangle^{\mathrm{H}}_{v_0} = \mathcal{T}^*$*;*
   - **b.** *Θ is strongly winning w.r.t.* Win*;*
   - **c.** *for all $i \in \mathrm{N}$,* $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$ *and* $\mathrm{RPenalty}_i(\Theta, v_0) \leq r_i$.

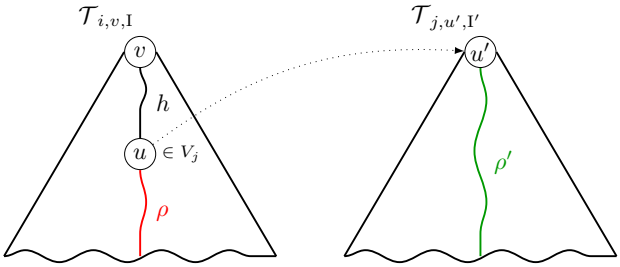


Figure 5 Example of forest that does not respect the resistance to constrained external deviations since $\mathrm{Gain}_i(\rho) = 0$ but $\mathrm{Gain}_i(\rho') = 1$.

2. *There exists a good forest $\mathcal{F}$ in $(\mathcal{G}, v_0)$ such that:*
   a. $\mathcal{T}_{0,v_0,\mathrm{I}_0} = \mathcal{T}^*$;
   b. *for all* $\rho \in \mathcal{T}^\infty_{0,v_0,\mathrm{I}_0}$, *for all* $i \in \mathrm{Win}$, $\mathrm{Gain}_i(\rho) = 1$;
   c. *for all* $i \in \mathrm{N}$, $\mathrm{MPenalty}_i(\mathcal{F}) \leq m_i$ *and* $\mathrm{RPenalty}_i(\mathcal{F}) \leq r_i$.

These assertions are still equivalent by replacing 1b by "Θ is weakly winning w.r.t. Win*" and 2b by "there exists $\rho \in \mathcal{T}^\infty_{0,v_0,\mathrm{I}_0}$ such that for all $i \in \mathrm{Win}$, $\mathrm{Gain}_i(\rho) = 1$".*

Proof sketch. For $(1 \Rightarrow 2)$, let us assume that Θ is a permissive SPE. We build a good forest $\mathcal{F}$ such that $\mathcal{T}_{0,v_0,\mathrm{I}_0}$ is the outcomes of Θ, *i.e.*, $\mathcal{T}_{0,v_0,\mathrm{I}_0} = \langle\Theta\rangle^{\mathrm{H}}_{v_0}$, and a tree $\mathcal{T}_{i,v,\mathrm{I}}$ is a representative of the outcomes of $\Theta_{\restriction hv'}$ in some subgame $(\mathcal{G}_{\restriction hv'}, v)$ such that $v' \in V_i$ and $\mathrm{Visit}(hv'v) = \mathrm{I}$. In order to obtain a good forest and since several $hv'v$ could satisfy those properties, each representative $\mathcal{T}_{i,v,\mathrm{I}}$ has to be chosen in a proper way: it has to minimize the maximal gain of player i for plays in $\mathcal{T}_{i,v,\mathrm{I}}$. More formally, for each $(i, v, \mathrm{I}) \in \mathcal{I}$, we let $\mathcal{O}(i, v, \mathrm{I}) = \{\langle\Theta_{\restriction hv'}\rangle^{\mathrm{H}}_v \mid hv'v \in \mathrm{Hist}(v_0) \wedge v' \in V_i \wedge \mathrm{I} = \mathrm{Visit}(hv'v)\}$ and we choose $\mathcal{T}_{i,v,\mathrm{I}} \in \mathcal{O}(i, v, \mathrm{I})$ such that $\max\{\mathrm{Gain}_i(\rho) \mid \rho \in \mathcal{T}^\infty_{i,v,\mathrm{I}}\} = \min_{\mathcal{T} \in \mathcal{O}(i,v,\mathrm{I})} \max\{\mathrm{Gain}_i(\rho) \mid \rho \in \mathcal{T}^\infty\}$.

Thanks to this latter property, $\mathcal{F}$ is good. Indeed, let $\mathcal{T}_{i,v,\mathrm{I}}$ be a tree of $\mathcal{F}$. Exactly as for permissive NEs, if $\mathcal{T}_{i,v,\mathrm{I}}$ is not $(\mathrm{N} \setminus \mathrm{I})$-resistant to internal deviations, we can build a strategy profile σ consistent with Θ such that the restriction of σ is not an NE in a subgame corresponding to $\mathcal{T}_{i,v,\mathrm{I}}$. If $\mathcal{T}_{i,v,\mathrm{I}}$ is not resistant to constrained external deviations that means that from some node u, owned by player j, there exists a play ρ losing for player j and player j could choose to play outside $\mathcal{T}_{i,v,\mathrm{I}}$ by jumping to a tree $\mathcal{T}_{j,u',\mathrm{I}'}$ in which there exists a play ρ' winning for him, see Figure 5. Let g be the history such that $\mathcal{T}_{i,v,\mathrm{I}}$ represents the outcomes of $\Theta_{\restriction g}$ in $(\mathcal{G}_{\restriction g}, v)$. Notice that $\langle\Theta_{\restriction gh}\rangle_{u'}$ may be different from $\mathcal{T}_{j,u',I'}$. However thanks to the way in which this representative is chosen, we have that there exists a play ρ'' in $\langle\Theta_{\restriction gh}\rangle_{u'}$ with $\mathrm{Gain}_j(\rho'') = 1$. Thus, we can build a strategy σ consistent with Θ such that $\langle\sigma_{\restriction gh}\rangle_u = \rho$ and $\langle\sigma_{\restriction ghu}\rangle_{u'} = \rho''$. This means that player j could deviate by choosing u' instead of u from v in the subgame $(\mathcal{G}_{\restriction g}, v)$, thus σ would not be an SPE and Θ not a permissive SPE.

For $(2 \Rightarrow 1)$, from a good forest $\mathcal{F}$ a multi-strategy is build such that its subgame outcomes are the trees of $\mathcal{F}$. This forms a permissive SPE because $\mathcal{F}$ is good. ◀

For now, good trees and trees in good forests are infinite, but Section 5 will show that we can represent some trees using a finite representation (intuitively, by supposing that every branch ends with a lasso in the game). It is this finite representation of good trees and good forests that will be used to decide the constrained penalty problems for permissive NEs and permissive SPEs, thanks to the characterizations of Theorems 5 and 8.

5 Computation of permissive equilibria

Theorems 5 and 8 characterize permissive NEs and SPEs with respect to infinite tree-shaped objects. In this section, we use these characterizations in order to decide the various penalty problems defined in Section 3: we check the existence of the good infinite tree-shaped objects by checking the existence of finite symbolic representations of such objects. We start by describing for a single tree this symbolic representation, and show that there exists a *polynomial-size* such representation (when the penalty upper-bounds are encoded in unary).

5.1 Symbolic trees and forests

▶ **Definition 9.** *A* symbolic tree *is a pair* $\mathcal{U} = (\mathcal{T}, f)$ *with* $\mathcal{T}$ *a finite tree (*i.e., *a finite subset of non-empty histories of* $\mathcal{G}$*), and* f *a function mapping each leaf* h *of* $\mathcal{U}$ *to a non-empty set of successor nodes* h' *that are ancestors of* h *in* $\mathcal{U}$ *such that* $(\mathrm{Last}(h), \mathrm{Last}(h')) \in E$.

A symbolic tree can be *unfolded* into an infinite tree by repeatedly expanding the leaves of $\mathcal{U}$ using as successors the choice prescribed by f. We denote by $\widetilde{\mathcal{U}}$ the infinite tree obtained by unfolding the symbolic tree $\mathcal{U}$. Similarly, the notions of symbolic forest $\mathcal{F}$, where every tree in it is a symbolic tree, and unfolding of symbolic forest $\widetilde{\mathcal{F}}$ can be defined.

In order to treat simultaneously NEs and SPEs, we introduce a new definition generalizing the resistance to external deviations and constrained external deviations. For a vector $\gamma \in \{0,1\}^{\mathrm{N} \times V \times 2^{\mathrm{N}}}$ of gains, and a subset $D \subseteq \mathrm{N}$ of players (that represent players that did not already win at the beginning of the tree), we say that a tree $\mathcal{T}$ is (γ, D)-resistant if for all $hu \in \mathcal{T}$ with $u \in V_i$ and $u' \in \mathrm{Succ}(u)$ with $huu' \notin \mathcal{T}$, if $\gamma_{i,u',(\mathrm{N}\setminus D)\cup \mathrm{Visit}(huu')} = 1$, if $i \notin (\mathrm{N} \setminus D) \cup \mathrm{Visit}(hu)$, then for all plays $\rho \in \mathcal{T}^{\infty}_{\restriction hu}$, $\mathrm{Gain}_i(\rho) = 1$.

▶ Remark 10. The notion of (γ, D)-resistance is close to the resistance to external deviations and constrained external deviations, so that we directly obtain from Theorems 5 and 8:

- Let $\gamma^{\mathcal{G}}$ be defined as follows: for all (i, u, I), we let $\gamma^{\mathcal{G}}_{i,u,\mathrm{I}}$ equals 1 iff player i belongs to I or can win from u against the coalition of the other players in $\mathcal{G}$. Let $\mathcal{T}$ be a tree. Then, $\mathcal{T}$ is a good tree iff $\mathcal{T}$ is resistant to internal deviations and $(\gamma^{\mathcal{G}}, \mathrm{N})$-resistant.
- Let $\mathcal{F}$ be a forest and let $\gamma^{\mathcal{F}}$ defined as follows: for all (j, u, J), we let $\gamma^{\mathcal{F}}_{j,u,\mathrm{J}}$ equals 1 iff player j belongs to J or the tree $\mathcal{T}_{j,u,\mathrm{J}}$ contains at least one branch with a vertex of F_j. Then, $\mathcal{F}$ is a good forest iff each each tree $\mathcal{T}_{i,v,\mathrm{I}}$ of $\mathcal{F}$ is $(\mathrm{N} \setminus \mathrm{I})$-resistant to internal deviations and $(\gamma^{\mathcal{F}}, \mathrm{N} \setminus \mathrm{I})$-resistant.

The challenge to make this remark a decision procedure is to make the tree and forest finitely representable. We treat each tree independently of each other, thus explaining how to symbolically represent one single tree in the following proposition:

▶ **Proposition 11.** *Let* $\mathcal{T}$ *be a tree that is* D*-resistant to internal deviations, with* $D \subseteq \mathrm{N}$. *We let* $\gamma \in \{0,1\}^{\mathrm{N} \times V \times 2^{\mathrm{N}}}$ *be a vector of gains such that* $\mathcal{T}$ *is* (γ, D)*-resistant, and* $(P_i)_{i \in \mathrm{N}'}$ *be finite constraints on penalties for a subset* $\mathrm{N}' \subseteq \mathrm{N}$ *of players. There exists a symbolic tree* $\mathcal{U}$*, that is a subtree of* $\mathcal{T}$*, of height polynomial in the number of players and vertices of* $\mathcal{G}$*, and in the largest bound on penalty* P_i*, such that the infinite tree* $\widetilde{\mathcal{U}}$ *satisfies the following properties:*

1. $\widetilde{\mathcal{U}}$ *is* D*-resistant to internal deviations;*
2. *in* $\widetilde{\mathcal{U}}$*, every player* $i \in \mathrm{N}'$ *has a penalty at most* P_i*;*
3. $\widetilde{\mathcal{U}}$ *is* (γ, D)*-resistant.*

Moreover, for a subset Win *of players, if we start with* $\mathcal{T}$ *that is strongly (respectively, weakly) winning w.r.t.* Win*, then we can make the above construction so that moreover* $\widetilde{\mathcal{U}}$ *is strongly (respectively, weakly) winning w.r.t.* Win.

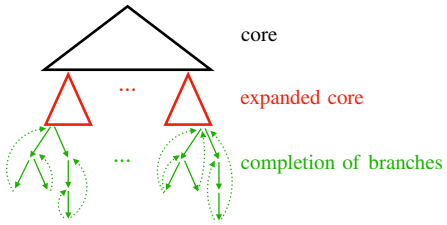


Figure 6 Construction of the symbolic tree.

The proof of this result goes by several steps, that we briefly sketch here only in the case where $\mathcal{T}$ is strongly winning w.r.t. Win. Figure 6 depicts the notions used in the construction of the symbolic tree. First, we consider the smallest subtree of $\mathcal{T}$ where leaves are such that all players of Win have visited their target set: this subtree is finite by König's lemma, since all branches of $\mathcal{T}$ have such a node where all players of Win have won, and the tree is finitely branching. This subtree is called the *core*. We then continue considering the parts of $\mathcal{T}$ outside the core, in order to complete the branches so that: the D-resistance to internal deviations is fulfilled (if a player has won in a certain branch of a subtree, he must win in all of them), the (γ, D)-resistance is fulfilled (if γ gives a constraint in the current node for a player i, all the branches of this subtree should visit a target vertex of i). This extension of the core is cut into two parts: the *expanded core* that ends in places where all the new players that must visit their target because of D-resistance to internal deviations and (γ, D)-resistance have indeed won; the *completion of branches* in order to then find leaves of the symbolic tree where all successors can be replaced (with function f) by similar nodes in the same branch, and the lassos thus formed are such that the penalty of players that have a finite penalty threshold does not increase along them. We show that these completions of branches can be chosen of polynomial length. We then compress the core and expanded core so that they also have polynomial height.

The symbolic tree $\mathcal{U}$ thus built is a subtree of $\mathcal{T}$ (even if its unfolding $\widetilde{\mathcal{U}}$ is not): in particular, as a corollary, if a player j has no winning play in $\mathcal{T}$, he does not have a winning play in $\mathcal{U}$ neither. In particular, when we apply independently this proposition to all the trees of a forest $\mathcal{F}$, to obtain a symbolic forest $\mathcal{H}$, this remark allows us to check that the new vector $\gamma^{\widetilde{\mathcal{H}}}$ has all its components not above the corresponding ones in $\gamma^{\mathcal{F}}$ (if $\gamma^{\mathcal{F}}_{i,v,\mathrm{I}} = 0$ then $\gamma^{\widetilde{\mathcal{H}}}_{i,v,\mathrm{I}} = 0$). In particular, if the tree $\mathcal{T}_{i,v,\mathrm{I}}$ of the forest $\mathcal{F}$ is $(\gamma^{\mathcal{F}}, \mathrm{N} \setminus \mathrm{I})$-resistant, then the tree $\widetilde{\mathcal{U}}_{i,v,\mathrm{I}}$ of the symbolic forest $\mathcal{H}$ is $(\gamma^{\widetilde{\mathcal{H}}}, \mathrm{N} \setminus \mathrm{I})$-resistant.

Finally, by combining this result with Remark 10, we obtain the following corollaries that allow us to obtain the PSPACE decision procedures:

▶ **Corollary 12.** *Let $m \in (\mathbb{N} \cup \{\infty\})^n$ be upper thresholds, and M be the largest such upper threshold. The following assertions are equivalent:*

1. *There exists a permissive NE Θ in $(\mathcal{G}, v_0)$ such that:*
   (a) Θ is strongly winning w.r.t. Win*; (b) for all $i \in \mathrm{N}$, $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$.*
2. *There exists a symbolic tree $\widetilde{\mathcal{T}}$ in $(\mathcal{G}, v_0)$ of height polynomial in the number of players and vertices of $\mathcal{G}$ and in M, such that*
   (a) $\widetilde{\mathcal{T}}$ is resistant to internal deviations, and $(\gamma^{\mathcal{G}}, \mathrm{N})$-resistant, with $\gamma^{\mathcal{G}}$ defined in Remark 10; and (b) for all $\rho \in \widetilde{\mathcal{T}}^{\infty}$ and $i \in$ Win, $\mathrm{Gain}_i(\rho) = 1$; and (c) for all $i \in \mathrm{N}$, $\mathrm{Penalty}_i(\widetilde{\mathcal{T}}) \leq m_i$.

These assertions are still equivalent by replacing 1(a) by "Θ is weakly winning w.r.t. Win*" and 2(b) by "there exists $\rho \in \widetilde{\mathcal{T}}^{\infty}$ such that for all $i \in$ Win, $\mathrm{Gain}_i(\rho) = 1$".*

▶ **Corollary 13.** *Let $m \in (\mathbb{N} \cup \{\infty\})^n$ and $r \in (\mathbb{N} \cup \{\infty\})^n$ be upper thresholds, and M be the largest such upper threshold. The following assertions are equivalent:*

1. *There exists a permissive SPE Θ in $(\mathcal{G}, v_0)$ such that:*
   (a) Θ is strongly winning w.r.t. Win*; and (b) for all $i \in$ N, $\mathrm{MPenalty}_i(\Theta, v_0) \leq m_i$ and $\mathrm{RPenalty}_i(\Theta, v_0) \leq r_i$.*
2. *There exists a symbolic forest $\mathcal{F}$ in $(\mathcal{G}, v_0)$, where each symbolic tree has a height polynomial in the number of players and vertices of $\mathcal{G}$ and in M, such that (a) each tree $\widetilde{\mathcal{T}}_{i,v,\mathrm{I}}$ is $(\mathrm{N} \setminus \mathrm{I})$-resistant to internal deviations, and $(\gamma^{\mathcal{F}}, \mathrm{N})$-resistant, with $\gamma^{\mathcal{F}}$ defined in Remark 10; (b) for all $\rho \in \widetilde{\mathcal{T}}^{\infty}_{0,v_0,\mathrm{I}_0}$ and $i \in$ Win, $\mathrm{Gain}_i(\rho) = 1$; and (c) for all $i \in$ N, $\mathrm{MPenalty}_i(\widetilde{\mathcal{F}}) \leq m_i$ and $\mathrm{RPenalty}_i(\widetilde{\mathcal{F}}) \leq r_i$.*

These assertions are still equivalent by replacing 1(a) by "Θ is weakly winning w.r.t. Win*" and 2(b) by "there exists $\rho \in \widetilde{\mathcal{T}}^{\infty}_{0,v_0,\mathrm{I}_0}$ such that for all $i \in$ Win, $\mathrm{Gain}_i(\rho) = 1$".*

5.2 Decision problems over permissive Nash equilibria

For permissive NEs, it makes little sense to take into consideration the retaliation penalties, since the punishment after a deviation should definitely make the deviator lose whatever the penalty from now on. We thus obtain the following decision result:

▶ **Theorem 14.** *The constrained penalty problem, the weakly winning with constrained penalty problem and the strongly winning with constrained penalty problem, all with infinite (and thus no) constraints on retaliation penalties and for NEs are decidable in* PSPACE *(when the penalty bounds are encoded in unary).*

Proof. We build upon Corollary 12, looking for a finite symbolic tree with the corresponding properties. We first explain how to solve the constrained penalty problem, and explain afterwards the adaptation for the two other problems. The idea is to use an alternating polynomial time Turing machine (since AP = PSPACE [12]) to guess a symbolic tree, checking the various constraints over it by using branch per branch. We describe the construction by supposing that the states of the Turing machine are split between existential states (where the machine accepts if at least one execution accepts) and universal states (where the machine accepts if all the executions accept). Existential states thus allow us to non-deterministically guess the finite symbolic tree node after node. We use a polynomial counter to keep track of the polynomially bounded height of the tree: if the counter goes over the polynomial bound, the execution of the alternating machine fails. At each node, existential states guess non-deterministically the set of successors on the working tape.

Universal states allow us to check several pieces of information on the guessed symbolic tree: the resistance to internal deviations, the constraint on the penalty for each player, and the $(\gamma^{\mathcal{G}}, \mathrm{N})$-resistance, with $\gamma^{\mathcal{G}}$ as in Remark 10. Notice that this vector has exponential size, but the index I in a triple (i, v, I) is useless (apart from knowing if $i \in \mathrm{I}$), and can thus be ignored: moreover, this set I will be maintained along the execution of the algorithm. This vector can thus be precomputed in (deterministic) polynomial time by determining, for each player, their set of winning vertices (against the coalition of the other players) [15].

The various checks can be performed branch per branch by keeping some pieces of information in memory, not only for the current node of the symbolic tree, but also for the whole current branch (this remains in polynomial space). Universal states are thus used to perform the checks on all the branches of the guessed tree.

- *Checking the penalty for player i.* If we have to check that the main penalty of player i is bounded by a threshold m_i (*i.e.,* that the penalty of player i over each branch is bounded by m_i), we keep in memory the current penalty, forbidding for it to go above m_i.

- *Checking the resistance to internal deviations and* $(\gamma^{\mathcal{G}}, \mathrm{N})$*-resistance.* At each node of the guessed tree, if the existential states guessed at least two successors, or depending on the vector $\gamma^{\mathcal{G}}$ (for a vertex v where $\gamma^{\mathcal{G}}$ has value 1, and that has not been chosen among the set of successors), we must remember constraints on the successors: either (a) all plays in their subtrees must be winning for a certain player i, or (b) none. We could add neither constraint (a) nor (b) for a certain player (if only one successor has been chosen, and the $\gamma^{\mathcal{G}}$ value of all the other successors is 0). In the case where only the resistance to internal deviation applies (if at least two successors have been chosen, and the $\gamma^{\mathcal{G}}$ value of all the other successors is 0), the choice of constraint (a) or (b) is guessed non-deterministically. These constraints are kept all along the guessed branch except if a vertex of the target set of player i is visited; in this case the constraint (a) is released. Moreover, the constraint (b) for a player i forbids to select a successor in the future where player i visits one of his target vertices.
- *The end of the branches.* The existential states decide when to stop the branch of the symbolic tree (before the counter runs out of the polynomial bound). Notice that the branch cannot stop if one of the type (a) constraints is not released. Then existential states provide the set of successors taken in the ancestors so that for players that have a finite upper threshold on their penalty, ancestors must have the same current penalty as the leaf (to ensure that their penalty does not raise to $+\infty$ in the long run).

For the strongly winning variants, universal states also check the constraint that every player of Win must win at the end of each branch. For the weakly winning variant, the existential states are also used to propose a branch where all players of Win will win. The universal states moreover check whether this condition is fulfilled for this particular branch. ◀

5.3 Decision problems over permissive subgame perfect equilibria

▶ **Theorem 15.** *The constrained penalty problem, the weakly winning with constrained penalty problem and the strongly winning with constrained penalty problem for SPEs are decidable in* PSPACE *(when the penalty bounds are encoded in unary).*

Proof. The proof is the same as for NEs, instead of the fact that we use Corollary 13, with a vector $\gamma^{\mathcal{F}}$ that is partially guessed non-deterministically when it is needed. When the existential states extend a branch of the tree $\mathcal{T}$ from a vertex of player i, the universal states does not only explore the chosen successors (with constraints (a) or (b) as in the previous proof), but now also explores the other vertices u by starting a fresh exploration of another tree $\mathcal{T}_{i,u,I}$ of the forest. Existential states also non-deterministically guess if player i is weakly winning in $\mathcal{T}_{i,u,I}$. If so, this gives new constraints (a) in the tree $\mathcal{T}$. The guessed weakly winning constraints are then checked in the fresh exploration: if player i must be weakly winning, this is a constraint of the same type as a weakly winning constraint in the "main" tree; if player i must not be weakly winning, this is a constraint of type (b) (none of the play must be winning for player i) that we deal as before.

Main penalties are checked as before. For the retaliation penalties, for each player, we check that the total penalty of all new symbolic trees $\mathcal{T}_{i,u,\mathrm{I}}$ is below the given upper threshold. To ensure polynomial time termination, we maintain a polynomial counter, and the set of trees (more precisely, the set of triples (i, u, I) used to index the trees of the forest) we jumped in so far. The polynomial counter again takes care of the depth of the branch we explore in the current tree (we reset this counter when we jump from a tree to another one). The set of trees we jumped in so far is maintained to forbid several explorations of the same tree of the forest. As for NEs, the exploration is losing if the depth of the current branch is longer than

the polynomial bound. The cardinal of the set of triples (i, u, I) we must maintain is also polynomial (bounded by $|\mathrm{N}| \times |V| \times |\mathrm{N}|$, even though there are exponentially many trees in a forest), since the subset I of winning players does not decrease along the jumps from a tree to the next one. This also implies that the total length of the executions of the Turing machine is indeed polynomial.

Notice that weakly and strongly winning conditions have only to be checked on the "main" tree as for permissive NEs. ◀

6 Conclusion

We studied the permissiveness in Nash, and subgame perfect equilibria over multiplayer reachability games. We showed that several associated problems are decidable in PSPACE: they ask for the existence of such equilibria with various constraints, both on the set of players who reach their target set, and on the penalties that allow us to compare the permissiveness of two equilibria. The polynomial space depends on the size of the game, and the largest upper threshold on the penalties. We were not able to decrease the space dependency to be only polynomial in the logarithm of the penalty thresholds: we leave for future work to investigate if this is possible, or if there is a matching lower bound on complexity.

As other ideas for future works, we would like to extend our study to other objectives than reachability, like more general ω-regular objectives (*e.g.,* parity games), but also weighted games like mean-payoff games, discounted-payoff games, or shortest-path games (where the reachability objective is combined with an objective to reach the target with the smallest possible total weight). An even more challenging problem is to extend this study to the setting of timed games, where the permissiveness is not only on the choice of edges, but also on the choice of delays spent in a given vertex. Work along these lines has been carried out on timed automata and two-player timed games [4, 13].

References

1 Ashwani Anand, Satya Prakash Nayak, and Anne-Kathrin Schmuck. Synthesizing permissive winning strategy templates for parity games. In *CAV 2023*, volume 13964 of *LNCS*, pages 436–458. Springer, 2023. doi:10.1007/978-3-031-37706-8_22.

2 Julien Bernet, David Janin, and Igor Walukiewicz. Permissive strategies: from parity games to safety games. *RAIRO Theor. Informatics Appl.*, 36(3):261–275, 2002. doi:10.1051/ITA:2002013.

3 Patricia Bouyer, Marie Duflot, Nicolas Markey, and Gabriel Renault. Measuring permissivity in finite games. In *CONCUR 2009*, volume 5710 of *LNCS*, pages 196–210. Springer, 2009. doi:10.1007/978-3-642-04081-8_14.

4 Patricia Bouyer, Erwin Fang, and Nicolas Markey. Permissive strategies in timed automata and games. *Electron. Commun. Eur. Assoc. Softw. Sci. Technol.*, 72, 2015. doi:10.14279/TUJ.ECEASST.72.1015.

5 Patricia Bouyer, Nicolas Markey, Jörg Olschewski, and Michael Ummels. Measuring permissiveness in parity games: Mean-payoff parity games revisited. In *ATVA 2011*, volume 6996 of *LNCS*, pages 135–149. Springer, 2011. doi:10.1007/978-3-642-24372-1_11.

6 Léonard Brice, Jean-François Raskin, and Marie van den Bogaard. On the Complexity of SPEs in Parity Games. In *CSL 2022*, volume 216 of *LIPIcs*, pages 10:1–10:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CSL.2022.10.

7 Léonard Brice, Jean-François Raskin, and Marie van den Bogaard. The Complexity of SPEs in Mean-Payoff Games. In *ICALP 2022*, volume 229 of *LIPIcs*, pages 116:1–116:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.116.

8 Léonard Brice, Jean-François Raskin, and Marie van den Bogaard. Subgame-perfect equilibria in mean-payoff games. *Logical Methods in Computer Science*, 19, 2023. doi:10.46298/LMCS-19(4:6)2023.

9 Thomas Brihaye, Véronique Bruyère, Aline Goeminne, and Jean-François Raskin. Constrained existence problem for weak subgame perfect equilibria with ω-regular boolean objectives. In *GandALF 2018*, volume 277 of *EPTCS*, pages 16–29, 2018. doi:10.4204/EPTCS.277.2.

10 Thomas Brihaye, Véronique Bruyère, Aline Goeminne, Jean-François Raskin, and Marie van den Bogaard. The complexity of subgame perfect equilibria in quantitative reachability games. In *CONCUR 2019*, volume 140 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.CONCUR.2019.13.

11 Thomas Brihaye, Véronique Bruyère, Aline Goeminne, and Nathan Thomasset. On relevant equilibria in reachability games. In *RP 2019*, volume 11674 of *LNCS*, pages 48–62. Springer, 2019. doi:10.1007/978-3-030-30806-3_5.

12 Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981. doi:10.1145/322234.322243.

13 Emily Clement, Thierry Jéron, Nicolas Markey, and David Mentré. Computing maximally-permissive strategies in acyclic timed automata. In *FORMATS 2020*, volume 12288 of *LNCS*, pages 111–126. Springer, 2020. doi:10.1007/978-3-030-57628-8_7.

14 Aline Goeminne and Benjamin Monmege. Permissive equilibria in multiplayer reachability games. Technical Report 2411.13296, arXiv, 2024. doi:10.48550/arXiv.2411.13296.

15 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *LNCS*. Springer, 2002. doi:10.1007/3-540-36387-4.

16 Satya Prakash Nayak and Anne-Kathrin Schmuck. Most general winning secure equilibria synthesis in graph games. In *TACAS 2024*, volume 14572 of *LNCS*, pages 173–193. Springer, 2024. doi:10.1007/978-3-031-57256-2_9.

17 Michael Ummels. Rational behaviour and strategy construction in infinite multiplayer games. In *FSTTCS 2006*, volume 4337 of *LNCS*, pages 212–223. Springer, 2006. doi:10.1007/11944836_21.

Propositional Logics of Overwhelming Truth

Thibaut Antoine
Univ Rennes, CNRS, IRISA, France

David Baelde
Univ Rennes, CNRS, IRISA, France

Abstract

Cryptographers consider that *asymptotic security* holds when, for any possible attacker running in polynomial time, the probability that the attack succeeds is *negligible*, i.e. that it tends fast enough to zero with the size of secrets. In order to reason formally about cryptographic truth, one may thus consider logics where a formula is satisfied when it is true with overwhelming probability, i.e. a probability that tends fast enough to one with the size of secrets. In such logics it is not always the case that either φ or $\neg\varphi$ is satisfied by a given model. However, security analyses will inevitably involve specific formulas, which we call *determined*, satisfying this property – typically because they are not probabilistic. The Squirrel proof assistant, which implements a logic of overwhelming truth, features ad-hoc proof rules for this purpose.

In this paper, we study several propositional logics whose semantics rely on overwhelming truth. We first consider a modal logic of overwhelming truth, and show that it coincides with S5. In addition to providing an axiomatization, this brings a well-behaved proof system for our logic in the form of Poggiolesi's hypersequent calculus. Further, we show that this system can be adapted to elegantly incorporate reasoning on determined atoms. We then consider a logic that is closer to Squirrel's language, where the overwhelming truth modality cannot be nested. In that case, we show that a simple proof system, based on regular sequents, is sound and complete. This result justifies the core of Squirrel's proof system.

2012 ACM Subject Classification Security and privacy → Formal methods and theory of security; Theory of computation → Modal and temporal logics; Theory of computation → Proof theory

Keywords and phrases Cryptography, Modal Logic, Sequent Calculus

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.24

Funding This work received funding from the France 2030 program managed by the French National Research Agency under grant agreement No. ANR-22-PECY-0006.

1 Introduction

In modern cryptography, one cannot hope for absolute truth. Considering a signature primitive sign with a freshly generated pair of secret and public keys sk and pk, it is expected that an attacker cannot forge a valid signature $\mathrm{sign}(m, k)$, even if he has had access to honestly generated signatures $\mathrm{sign}(m_i, k)$ for $i \in [1; n]$ – unless of course $m = m_i$ for some i. However, one cannot rule out the possibility that the attacker guesses the secret key: brute force attacks are always possible. Hence, cryptographers must work with a complex notion of truth, restricting attackers to limited resources and admitting a small probability of success. In provable cryptography, a system is said to be *asymptotically secure* when, for any attacker represented as a probabilistic polynomial-time Turing machine, the probability that an attack succeeds is negligible, i.e. asymptotically smaller than the inverse of any positive polynomial in the length of secret keys increases [18]. Dually, we expect that the system remains secure with *overwhelming* probability, i.e. that tends fast enough to one as key lengths increase. In other words, overwhelming truth is the working notion of truth for cryptographers.

In order to ease formal proofs in presence of this complex notion of cryptographic truth, cryptographers have developped specific proof techniques, such as game hopping and reasoning up-to failure [27]. Going further, formal systems have been developed and implemented to mechanize cryptographic proofs [13, 10, 11, 1], successfully bringing together proof techniques coming from both cryptography, program verification and theorem proving – see [9] for a survey. All of the above mentionned systems allow the explicit manipulation of probabilities. This may be desired, e.g. to formalize *concrete security* arguments where precise, explicit bounds are derived for the attacker's advantage. However, this level of detail makes for very tedious proofs, and seems unnecessary to formalize the large body of work on e.g. the asymptotic security of cryptographic protocols. Bana and Comon have proposed to solve this issue with the CCSA approach [7, 8], which builds on the standard framework of first-order logic to provide a formal language that abstracts away probabilities and asymptotic reasoning. After some successful uses on paper [14, 21, 26, 6], the CCSA approach has been mechanized in the Squirrel proof assistant [2, 3, 16], which implements a higher-order version of the CCSA logic [5]. In that logic, one may write e.g. an authentication property as a formula of the form $[\forall \tau.\ \text{happens}(\tau) \Rightarrow \text{condition}@\tau \Rightarrow \exists \tau'.\ \tau' < \tau \wedge \text{event}@\tau']$ which intuitively expresses that, in any execution trace of a protocol, if some participant checks a condition at any time point inside the trace, some event must have happened earlier in the trace. Crucially, the $[\varphi]$ construct expresses that the enclosed formula φ holds with overwhelming probability. Such authentication formulas might be consequences of axioms expressing (overwhelmingly true) cryptographic assumptions, e.g. about signatures, and they might have more complex formulas as consequences – the CCSA logics feature a predicate expressing *computational indistinguishability*, which we will leave aside in this work. In typical Squirrel proofs, one often works with formulas of the form $\forall \tau.\ [\text{happens}(\tau)] \Rightarrow \varphi$, where we say that φ holds for any time point τ in the trace. If the considered protocol only features actions A and B, an axiom will allow to rewrite this as $\forall \tau.\ [\tau = A \vee \tau = B] \Rightarrow \varphi$. At this point it would be tempting to conclude that our formula holds provided that both $\varphi[\tau := A]$ and $\varphi[\tau := B]$ hold; proving these properties might then rely on the specification of the actions A and B. However, such a case analysis is in general unsound due to the probabilistic nature of the logic: $\tau \in \{A, B\}$ might be true with overwhelming probability for a random variable τ that is equal to A (resp. B) with probability $1/2$, in which case neither $\tau = A$ nor $\tau = B$ will be true with overwhelming probability. If relevant, this problem might be worked around by assuming that τ is actually deterministic, i.e. considering the formula $\forall \tau.\ \det(\tau) \Rightarrow [\tau = A \vee \tau = B] \Rightarrow \varphi$.

Research in the CCSA line of work has been mainly concerned with justifying the soundness of the logic wrt. the cryptographic model, the soundness of the proposed proof systems wrt. the logic, and with concretely verifying cryptographic protocols to justify the practicality of the approach. In comparison, few investigations have looked carefully into the fine structure of the logic and associated proof systems. Scerri and Koutsos have established the decidability of provability for some CCSA systems [15, 22], and a recent work in a variant of the CCSA logic with explicit bounds improves these bounds via proof transformations [4]. However, no completeness result has ever been proved for a CCSA logic, and the proof systems proposed in [2, 3, 5] are just sound collections of rules, some of which may be deemed ad-hoc. Over time, these systems have been structured around notions of local and global sequents [3] which are only justified by their practical usefulness. Moreover, proof systems incorporate a few ad-hoc rules [5] that can take into account the det(_) assumptions: for instance, a rule essentially allows to treat $[\varphi \vee \psi]$ as $[\varphi] \vee [\psi]$ when $\det(\varphi)$. The complexity of the logic, as well as its practical relevance, calls for a more careful design.

In this paper, we provide the first answers to these concerns. We restrict to the propositional fragment of Squirrel's logic [5], which we naturally reframe as a modal logic: we view $[\varphi]$ as $\Box\varphi$, where the $\Box$ modality is interpreted as overwhelming truth. This allows us to apply the well-established concepts and techniques of modal logic and proof theory. We define in Section 2 (several possible variants of) a modal logic of overwhelming truth and show in Section 3 that it coincides with the modal logic S5. This result allows to transfer existing proof systems and model-theoretic techniques from S5 to our modal logic of overwhelming truth. In particular, we show in Section 4 that Poggiolesi's hypersequent calculus can be nicely adapted to incorporate reasoning about determined formulas – a slight variant of the det(_) predicate presented above. We also show that, for the fragment of our logic that most closely corresponds to the language of Squirrel, hypersequents are not necessary: we introduce in Section 5 a sound and complete sequent calculus based on Squirrel's notions of local and global sequents. We conclude in Section 6 with some discussion of related and future works.

2 Modal logics of overwhelming truth

We define several *modal logics of overwhelming truth*, whose formulas are standard modal logic formulas, interpreted as families of random variables with the box modality corresponding to overwhelming truth.

We recall that a *probability space* is a triple $X = (S, \Omega, \mu)$ where S is a non-empty set of *samples*, the set of *events* $\Omega \subseteq 2^S$ is a σ-algebra[1], and $\mu : \Omega \to [0; 1]$ is a measure[2] assigning a probability to each event of Ω, such that $\mu(S) = 1$. A *random variable* $V : X \to Y$ from probability space $X = (S, \Omega, \mu)$ to probability space $Y = (S', \Omega', \mu')$ is a function from S to S' such that the pre-image of any event in Y is an event in X: for all $E' \in \Omega'$, $V^{-1}(E') \in \Omega$. This allows us to define $\Pr_{x\in X}\big(V(x) \in E'\big)$ as $\mu(V^{-1}(E'))$ for any event $E' \in \Omega'$.

As is common, we will identify a probability space $X = (S, \Omega, \mu)$ with its sample space, saying for instance that $x \in X$ when $x \in S$. Conversely, we identify a finite set S with the *discrete probability space* $(S, 2^S, \mu)$ where $\mu(E) = |E|/|S|$ – we typically take S to be $\{0, 1\}$ or $\{0, 1\}^k$.

▶ **Definition 1** (Modal formulas). *We assume a set $\mathcal{P}$ of propositional variables. Modal formulas are then built from the following grammar: $\varphi ::= \bot \mid p \mid \varphi \Rightarrow \varphi \mid \Box\varphi$. As usual, we will use other logical connectives as they can be defined from the above ones. For instance, we will write $\varphi \vee \psi$ for $\neg\varphi \Rightarrow \psi$ and $\Diamond\varphi$ for $\neg\Box\neg\varphi$.*

We define next an *abstract* modal logic of overwhelming truth, where validity corresponds to overwhelming truth in a general class of models. Several variants of this logic are then obtained by restricting to particular classes of models. As we shall see in the next section, all these variants are actually equivalent.

2.1 The abstract modal logic of overwhelming truth

We will interpret formulas as families of random variables, indexed by the security parameter $\eta \in \mathbb{N}$. To do so, we use an abstract notion of *cryptographic structure* that provides a family of measure spaces, and interprets each variable as a family of random variables over these spaces.

[1] A σ-algebra must be non-empty and closed under complement, countable unions and intersections.
[2] A measure must satisfy $\mu(\biguplus_{i\in\mathbb{N}} E_i) = \sum_{i\in\mathbb{N}} \mu(E_i)$.

▶ **Definition 2** (Cryptographic structure)**.** *A cryptographic structure $\mathcal{S}$ is given by:*

- *A sequence of probability spaces $(X_\eta^{\mathcal{S}})_{\eta\in\mathbb{N}}$. For $\eta \in \mathbb{N}$, we let $\mathsf{RV}_\eta^{\mathcal{S}}$ be the set of random variables from $X_\eta^{\mathcal{S}}$ to $\{0,1\}$, and $\mathsf{RV}^{\mathcal{S}} = \{\ (U_\eta)_{\eta\in\mathbb{N}} \mid U_\eta \in \mathsf{RV}_\eta^{\mathcal{S}}$ for all $\eta\ \}$.*
- *For each propositional variable $p \in \mathcal{P}$, an interpretation $p^{\mathcal{S}} \in \mathsf{RV}^{\mathcal{S}}$.*

When $U = (U_\eta)_{\eta\in\mathbb{N}} \in \mathsf{RV}^{\mathcal{S}}$, $\eta \in \mathbb{N}$ and $\rho \in X_\eta^{\mathcal{S}}$, we write $U(\eta,\rho)$ for $U_\eta(\rho)$. Conversely, we may define an element $U \in \mathsf{RV}^{\mathcal{S}}$ by defining $U(\eta,\rho)$ for each $\eta \in \mathbb{N}$ and $\rho \in \mathsf{RV}_\eta^{\mathcal{S}}$.

A function $f : \mathbb{N} \to [0,1]$ is *negligible* when f is asymptotically smaller than $\eta \mapsto \eta^{-k}$ for any $k \in \mathbb{N}$. The function is *overwhelming* when $\eta \mapsto 1 - f(\eta)$ is negligible. For conciseness, we will also say that a family of random variables $U \in \mathsf{RV}^{\mathcal{S}}$ (for some cryptographic structure $\mathcal{S}$) is overwhelming when $\eta \mapsto \Pr_{\rho\in X_\eta^{\mathcal{S}}}\big(U(\eta,\rho) = 1\big)$ is overwhelming.

▶ **Definition 3.** *Given a cryptographic structure $\mathcal{S}$ and a formula φ, we define the interpretation $[\![\varphi]\!]_{\mathcal{S}} \in \mathsf{RV}^{\mathcal{S}}$ as follows, for all $\eta \in \mathbb{N}$ and $\rho \in X_\eta^{\mathcal{S}}$:*

- $[\![p]\!]_{\mathcal{S}}(\eta,\rho) = p^{\mathcal{S}}(\eta,\rho)$ *for $p \in \mathcal{P}$;*
- $[\![\bot]\!]_{\mathcal{S}}(\eta,\rho) = 0$*;*
- $[\![\varphi \Rightarrow \psi]\!]_{\mathcal{S}}(\eta,\rho) = 1$ *iff* $[\![\varphi]\!]_{\mathcal{S}}(\eta,\rho) \leq [\![\psi]\!]_{\mathcal{S}}(\eta,\rho)$*;*
- $[\![\Box\varphi]\!]_{\mathcal{S}}(\eta,\rho) = 1$ *iff $[\![\varphi]\!]_{\mathcal{S}}$ is overwhelming.*

▶ **Definition 4** (Validity)**.** *A formula φ is* valid *wrt. a class of cryptographic structures when $[\![\varphi]\!]_{\mathcal{S}}$ is overwhelming for any $\mathcal{S}$ in that class. We simply say that φ is valid when it is valid wrt. all cryptographic structures.*

We can now define our first logic. As is standard, we define a logic as a set of formulas called the *theorems* of that logic.

▶ **Definition 5.** *The* abstract modal logic of overwhelming truth *is the set of modal formulas that are valid wrt. all cryptographic structures.*

▶ **Example 6.** Let φ and ψ be arbitrary modal formulas.

- We have that φ is valid iff $\Box\varphi$ is valid: for any $\mathcal{S}$, $[\![\varphi]\!]_{\mathcal{S}}$ is overwhelming iff $[\![\Box\varphi]\!]_{\mathcal{S}}$ is overwhelming.
- The formula $\Box(\varphi \wedge \psi) \Rightarrow \Box\varphi \wedge \Box\psi$ is valid. Indeed, in any $\mathcal{S}$ where $[\![\varphi \wedge \psi]\!]_{\mathcal{S}}$ is overwhelming, so are $[\![\varphi]\!]_{\mathcal{S}}$ and $[\![\psi]\!]_{\mathcal{S}}$.
- The formula $\Box(\varphi \vee \psi) \Rightarrow \Box\varphi \vee \Box\psi$ is not valid. Indeed, φ and ψ might both be true with probability $1/2$ (for all η) in such a way that their disjunction is true with probability 1 (for all η).

Unlike Squirrel's logic, our modal logic allows the nesting of modal boxes expressing overwhelming truth. Our logic is otherwise much less expressive than Squirrel's, not only because that logic allows (higher-order) quantifications. In our modal logic, propositional variables are interpreted as families of random variables, where the random variables corresponding to different values of η might be completely unrelated. In contrast, Squirrel's logic features predicates that allow to restrict to families of random variables that do not vary with η, or to deterministic families, i.e. families of constant random variables. Finally, Squirrel's logic features the computational indistinguishability predicate, absent from our modal logic, which allows to state, e.g., that two propositional formulas yield probability distributions that are negligibly different.

▶ **Proposition 7.** *The abstract logic of overwhelming truth is a* normal modal logic*: its theorems are closed under substitution and* modus ponens*; they contain classical tautologies and the K axiom $\Box(p \Rightarrow q) \Rightarrow \Box p \Rightarrow \Box q$; moreover, $\Box\varphi$ is a theorem whenever φ is.*

This interested reader may find the proof of this result in Section A.

2.2 Variants

It is natural to consider several variations on the abstract modal logic of overwhelming truth, obtained by considering validity wrt. restricted classes of cryptographic structures. The first variant is of particular interest to us, since it corresponds to the class of models considered in Squirrel, i.e., to the *term structures* that are suitable for interpreting *names* over *large* types [5].

▶ **Definition 8.** *We say that a cryptographic structure $\mathcal{S}$ is* concrete *when each $X_\eta^{\mathcal{S}} = \{0,1\}^{\ell_\eta}$ for some ℓ_η, with $\eta \mapsto \ell_\eta$ strictly increasing. The* concrete modal logic of overwhelming truth *is the set of modal formulas that are valid wrt. concrete cryptographic structures.*

The next variant is a modal logic of "truth with probability 1", obtained by restricting to a class of structures where overwhelming truth is the same as truth with probability 1.

▶ **Definition 9.** *The* static modal logic of overwhelming truth *is the set of modal formulas that are valid wrt. all cryptographic structures $\mathcal{S}$ such that the same probability space is used for all η (i.e., $X_\eta^{\mathcal{S}} = X_{\eta'}^{\mathcal{S}}$ for all η, η').*

Restricting even further the considered set of structures to discrete probability spaces, we obtain a logic where $\Box\varphi$ reads as "φ is true for all samples (with non-zero probability)".

▶ **Definition 10.** *The* static discrete modal logic of overwhelming truth *is the set of modal formulas that are valid wrt. all cryptographic structures $\mathcal{S}$ such that the same* discrete *probability space is used for all η.*

Note that the observations of Example 6 still hold if one considers validity wrt. any of the above classes. Proposition 7 also holds for all of our variant logics. As we shall see, all of the above variants are actually the same logic. Other variants are possible that would yield the same logic, e.g. considering infinite concrete cryptographic structures where samples are infinite bitstrings. We do not intend to exhaustively list equivalent variants, and believe that the reader should be able to adapt our techniques to handle new variants.

3 Soundness and completeness with respect to S5

We now prove that our modal logics of overwhelming truth coincide with S5, the smallest normal modal logic containing the following axioms:

$$\begin{array}{lll} \text{(axiom T)} & \Box p & \Rightarrow\ p \\ \text{(axiom 4)} & \Box p & \Rightarrow\ \Box\Box p \\ \text{(axiom 5)} & \Diamond p & \Rightarrow\ \Box\Diamond p \end{array}$$

We can immediately observe that S5 is included in our modal logics; a detailed proof is given in Section B.

▶ **Lemma 11.** *All S5 theorems are theorems of the modal logics of overwhelming truth.*

In order to prove the converse, a model-theoretic characterization of S5 will be useful.

▶ **Definition 12** (Kripke structure)**.** *A Kripke structure $\mathcal{K}$ is given by:*

- *a frame $(W^{\mathcal{K}}, \mathcal{R}^{\mathcal{K}})$ where W is a set of* worlds*, and $\mathcal{R}$ is a binary relation over W;*
- *for each world $w \in W$, a set of propositional variables $V^{\mathcal{K}}(w) \subseteq \mathcal{P}$.*

▶ **Definition 13** (Equivalence and clique frames). *A Kripke frame $(W, \mathcal{R})$ is an equivalence relation when $\mathcal{R}$ is symmetric, reflexive and transitive. It is a clique when $\mathcal{R}$ is the full binary relation over W. Further, it is a finite clique when W is finite.*

We may omit the $\mathcal{K}$ superscripts when they are clear from the context.

▶ **Definition 14** (Satisfaction). *Given a modal logic formula φ, a Kripke structure $\mathcal{K}$ and a world $w \in W^{\mathcal{K}}$, we define the satisfaction relation $\mathcal{K}, w \models \varphi$ as follows:*

- $\mathcal{K}, w \models p$ *iff* $p \in V^{\mathcal{K}}(w)$;
- $\mathcal{K}, w \not\models \bot$;
- $\mathcal{K}, w \models \varphi \Rightarrow \psi$ *iff* $\mathcal{K}, w \models \varphi$ *implies* $\mathcal{K}, w \models \psi$;
- $\mathcal{K}, w \models \Box\varphi$ *iff* $\mathcal{K}, w' \models \varphi$ *for all* $w' \in W^{\mathcal{K}}$ *such that* $w \mathcal{R}^{\mathcal{K}} w'$.

A formula is valid wrt. a class of frames when it is satisfied at all worlds of all Kripke structures whose frame belongs to the class. For instance, a formula is valid wrt. clique frames when it is satisfied at all worlds of all Kripke structures whose frame is a clique.

We summarize next some well-known characterizations of S5, proved e.g. in [12]. More details are given in Section B.

▶ **Proposition 15.** *For any modal formula φ, the following conditions are equivalent:*

1. *φ is a theorem of S5;*
2. *φ is valid wrt. equivalence frames;*
3. *φ is valid wrt. clique frames.*
4. *φ is valid wrt. finite clique frames.*

▶ **Lemma 16.** *Our modal logics of overwhelming truth are contained in S5.*

Proof. We prove the result for the concrete logic of overwhelming truth, and then explain how the argument can be adapted for the other logics.

By Proposition 15 it suffices to show that, if there is a finite clique counter-model of a modal formula, then there is a concrete cryptographic structure in which the formula is not overwhelmingly true. Let $\mathcal{K}$ be a finite clique Kripke structure, with $W^{\mathcal{K}} = \{w_1, \ldots, w_n\}$. We define a concrete cryptographic structure $\mathcal{S}$ with $X^{\mathcal{S}}_\eta = \{0,1\}^\eta$, for all η. We can partition each $X^{\mathcal{S}}_\eta$ into n disjoint sets called $X^{\mathcal{S}}_{\eta,i}$ for $i \in [1; n]$, such that asymptotically (in η) they all have size η/n. We define the interpretation of propositional variables in $\mathcal{S}$ so that, for each η, the same variables are true in w_i and $X^{\mathcal{S}}_{\eta,i}$. Formally, for $p \in \mathcal{P}$ in $\mathcal{S}$, we define:

$$p^{\mathcal{S}}(\eta, \rho) = 1 \text{ iff } p \in V^{\mathcal{K}}(w_i) \text{ for the unique } i \in [1; n] \text{ such that } \rho \in X^{\mathcal{S}}_{\eta,i}$$

As a result of our construction, $[\![\psi]\!]_{\mathcal{S}}(\eta, \rho) = [\![\psi]\!]_{\mathcal{S}}(\eta, \rho')$ for any modal formula ψ, $\eta \in \mathbb{N}$, and $\rho, \rho' \in X^{\mathcal{S}}_{\eta,i}$. Further, we shall see that, for any modal formula ψ, $w_i \in W^{\mathcal{K}}$, $\eta \in \mathbb{N}$ and $\rho \in X^{\mathcal{S}}_{\eta,i}$, we have:

$$\mathcal{K}, w_i \models \psi \text{ iff } [\![\psi]\!]_{\mathcal{S}}(\eta, \rho) = 1.$$

This is proved by induction on ψ. The cases where ψ is $\bot$, $\psi \in \mathcal{P}$ or ψ is an implication are easily verified. Assume now that $\psi = \Box\theta$. We have:

$$\begin{array}{lll} \mathcal{K}, w_i \models \psi & \text{iff} & \mathcal{K}, w_j \models \theta \text{ for all } j \in [1; n] \qquad (\mathcal{K} \text{ is a clique}) \\ & \text{iff} & [\![\theta]\!]_{\mathcal{S}}(\eta', \rho') \text{ for all } \eta', \rho' \qquad (\text{induction hypothesis, and } X^{\mathcal{S}}_\eta = \cup_j X^{\mathcal{S}}_{\eta,j}) \end{array}$$

As a result $\mathcal{K}, w_i \models \psi$ does imply that $[\![\theta]\!]_{\mathcal{S}}$ is overwhelming, i.e. $[\![\psi]\!]_{\mathcal{S}}(\eta, \rho) = 1$. Conversely, if $[\![\theta]\!]_{\mathcal{S}}$ is overwhelming in $\mathcal{S}$, then because the $X^{\mathcal{S}}_{\eta,i}$ have asymptotic size η/n, we must have $[\![\theta]\!]_{\mathcal{S}}(\eta, \rho) = 1$ for η large enough and any $\rho \in X^{\mathcal{S}}_\eta$. Hence $\mathcal{K}, w_j \models \theta$ for all i, and $\mathcal{K}, w_i \models \psi$.

To conclude, observe that if $\mathcal{K}$ is a finite clique counter-model of a modal formula φ, then $\mathcal{K}, w_i \not\models \varphi$ for some i. By our observation, $[\![\varphi]\!]_{\mathcal{S}}(\eta, \rho) = 0$ for all η and $\rho \in X^{\mathcal{S}}_{\eta,i}$. Because $X^{\mathcal{S}}_{\eta,i}$ has asymptotic size η/n, $[\![\varphi]\!]_{\mathcal{S}}$ is not overwhelming. Hence the concrete modal logic of overwhelming truth is contained in S5.

This argument also shows that the abstract modal logic of overwhelming truth is contained in S5. To obtain the result for the static and static discrete[3] modal logics of overwhelming truth, the argument can be easily adapted: it suffices to take $X^{\mathcal{S}}_{\eta} = \{w_1, \ldots, w_n\}$ for each η, with $X^{\mathcal{S}}_{\eta,i} = \{w_i\}$. ◀

Putting our two lemmas together, we obtain the characterization of our modal logics of overwhelming truth. In the rest of the paper, we shall indiscriminately talk of *the* modal logic of overwhelming truth, and we will interchangeably use this logic and S5.

▶ **Theorem 17.** *S5 coincides with all of our modal logics of overwhelming truth.*

4 Hypersequents for overwhelming truth with determined formulas

Having proved that the modal logic of overwhelming truth coincides with S5 directly provides a deductive system for overwhelming truth, in the form of the S5 axioms. More interestingly, it allows to benefit from better structured proof systems for S5, such as Poggiolesi's sound and complete hypersequent calculus for S5 [25]. This system is analytical, enjoys cut elimination and is well-adapted to proof-search. As we shall see, Poggiolesi's system can also be elegantly adapted to incorporate reasoning on determined formulas.

4.1 A variant of Poggiolesi's hypersequent calculus for S5

In [25], Poggiolesi introduces the hypersequent calculus $\mathrm{CSS5}_s$ and proves that it is complete for S5 using syntactical methods. A semantical proof is also possible, which will provide a more convenient foundation for our needs. We introduce below a slight modification of her calculus that facilitates such a proof – we explain these modifications afterwards. Unlike Poggiolesi, we view sequents and hypersequents as sets rather than multisets.

▶ **Definition 18.** *A* classical *sequent $\Gamma \vdash \Delta$ is composed of two finite sets of formulas Γ and Δ. We use the comma to denote the union of sets of formulas, also writing Γ, φ for $\Gamma \cup \{\varphi\}$.*

▶ **Definition 19.** *A* hypersequent *is a finite set of classical sequents. We use the letter $\mathcal{H}$ to denote hypersequents, and the vertical bar to denote the addition of a sequent to a hypersequent, i.e. $\mathcal{H} \mid \Gamma \vdash \Delta$ stands for $\mathcal{H} \cup \{\Gamma \vdash \Delta\}$.*

When $\mathcal{H} = (\Gamma_1 \vdash \Delta_1 \mid \ldots \mid \Gamma_n \vdash \Delta_n)$ is a hypersequent, we note $\mathrm{rhs}(\mathcal{H}) = \cup_i \Delta_i$ the union of the right-hand sides of its sequents. We define similarly $\mathrm{lhs}(\mathcal{H}) = \cup_i \Gamma_i$.

▶ **Definition 20.** *The* formula interpretation *of a hypersequent $\Gamma_1 \vdash \Delta_1 \mid \ldots \mid \Gamma_n \vdash \Delta_n$ is the modal formula $\Box(\bigwedge \Gamma_1 \Rightarrow \bigvee \Delta_1) \vee \ldots \vee \Box(\bigwedge \Gamma_n \Rightarrow \bigvee \Delta_n)$.*

We present in Figure 1 a slight variant of Poggiolesi's $\mathrm{CCS5}_s$ hypersequent calculus. Rules on the first two lines are immediate embeddings of classical propositional rules of sequent calculus in hypersequents: they apply to any sequent in the hypersequent, and the

[3] The result should not come as a surprise for the static discrete modal logic of overwhelming truth, given the analogy between the S5 characterization in terms of clique Kripke structure and the fact that overwhelming truth in static discrete structures is equivalent to truth for all samplings.

$$\frac{}{\mathcal{H} \mid \Gamma, \varphi \vdash \varphi, \Delta} \qquad \frac{}{\mathcal{H} \mid \Gamma, \bot \vdash \Delta}$$

$$\frac{\mathcal{H} \mid \Gamma, \varphi \Rightarrow \psi \vdash \varphi, \Delta \quad \mathcal{H} \mid \Gamma, \varphi \Rightarrow \psi, \psi \vdash \Delta}{\mathcal{H} \mid \Gamma, \varphi \Rightarrow \psi \vdash \Delta} \qquad \frac{\mathcal{H} \mid \Gamma, \varphi \vdash \psi, \varphi \Rightarrow \psi, \Delta}{\mathcal{H} \mid \Gamma \vdash \varphi \Rightarrow \psi, \Delta}$$

$$\frac{\mathcal{H} \mid \Gamma, \Box\varphi, \varphi \vdash \Delta}{\mathcal{H} \mid \Gamma, \Box\varphi \vdash \Delta} \qquad \frac{\mathcal{H} \mid \Gamma, \Box\varphi \vdash \Delta \mid \Gamma', \varphi \vdash \Delta'}{\mathcal{H} \mid \Gamma, \Box\varphi \vdash \Delta \mid \Gamma' \vdash \Delta'} \qquad \frac{\mathcal{H} \mid \Gamma \vdash \Box\varphi, \Delta \mid \cdot \vdash \varphi}{\mathcal{H} \mid \Gamma \vdash \Box\varphi, \Delta} \; \varphi \notin \mathrm{rhs}(\mathcal{H}), \Delta$$

Figure 1 Rules of Poggiolesi's (modified) hypersequent calculus for S5.

surrounding hypersequent structure is simply copied in the usual premisses. The axiom rule (top left) derives any hypersequent featuring a sequent that has the same formula on both sides. As usual, this rule can be restricted to the case where φ is atomic [25] without loosing completeness. The implication left rule applies to any hypersequent featuring a sequent that has an implication formula $\varphi \Rightarrow \psi$ on its left side; it features two premisses, one where φ has moved to the right-hand side of the corresponding sequent and one where ψ has replaced $\varphi \Rightarrow \psi$. In both the left and right implication rules, we keep the principal formula $\varphi \Rightarrow \psi$ (shown in gray) in the premises rather than removing it, as would be more usual. This minor technical difference makes it easier to prove that proof search is terminating, with no negative effect on the efficiency of proof search because the strategies that we will consider can never introduce twice the same formula. Hence, from the proof-search point of view, the grayed out formulas can be seen as pure book-keeping devices.

The system features a right modal rule (bottom right of Figure 1) which, from a (top-down) proof search perspective, creates a new sequent $\vdash \varphi$ when $\Box\varphi$ is found on the right of a sequent – note that formula interpretations of the premise and conclusion hypersequents of this rule are (propositionally) equivalent. Here again, we keep the principal formula $\Box\varphi$ (shown in gray) in the premise. Importantly, the right modal rule can only be applied with principal formula $\Box\varphi$ when the φ does not occur (at toplevel) on the right-hand side of any sequent of the conclusion hypersequent. There are two left rules, which may be seen as two versions of the same rule: from a proof-search perspective, it allows to add φ on the left-hand side of any sequent in the hypersequent if one sequent features $\Box\varphi$ on its left; the first variant of the rule is for when φ is added to the sequent that contains $\Box\varphi$, while the second one if for when φ is added to another sequent.

The rules of our system differ from Poggiolesi's [25] in several minor ways. We use (hyper)sequents as sets rather than multisets, due to our focus on proof-search and semantical methods, while Poggiolesi focuses on syntactic methods, including cut elimination. Poggiolesi's calculus also treats conjunction and negation as elementary connectives, while we only consider implication as an elementary connectives. However, our rule for implication is the straightforward combination of her rules for conjunction and negation, applied on $\varphi \Rightarrow \psi$ seen as $\neg(\varphi \wedge \neg\psi)$. The more important difference is our inclusion of the principal formulas $\varphi \Rightarrow \psi$ and $\Box\varphi$ in the premisses of the corresponding rules, i.e. the occurrences shown in gray in Figure 1, and the addition of the side condition on the right modal rule. These modifications allow for a simple proof that proof-search is terminating.

Before providing a proof of this result, let us discuss it in more details. When considering (top-down) proof search in Poggiolesi's calculus, without our gray formulas, the number of logical connectives in a hypersequent decreases strictly with the application of any rule other than the left modal rules. Indeed, it is possible to repeatedly apply a left modal rule, adding

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Box\neg\Box p \vdash p \mid \cdot \vdash p \mid \cdot \vdash p}}{\Box\neg\Box p \vdash p \mid \cdot \vdash \Box p, p}}{\Box\neg\Box p \vdash p \mid \neg\Box p \vdash p}}{\Box\neg\Box p \vdash p \mid \cdot \vdash p}}{\Box\neg\Box p \vdash \Box p, p}}{\Box\neg\Box p, \neg\Box p \vdash p}}{\Box\neg\Box p \vdash p}$$

Figure 2 Non-terminating proof-search in $\mathrm{CSS5}_s$.

an even increasing number of copies of φ to the left-hand sides of sequents. However, as observed in [25], this behaviour is useless: only a single application of the left rule (for any given φ and target sequent) is present in derivations of minimal height. Building on this observation, [25, Theorem6.5] claims that $\mathrm{CSS5}_s$ allows terminating proof-search: while this is true, the proof seems to overlook the non-terminating behaviour induced by the right modal rule. Indeed, non-termination can arise because of the right modal rule (in its original version without the side condition present in Figure 1), despite the parcimonious use of left modal rules, as illustrated in Figure 2.

In order to avoid this issue, proof-search must not only avoid repeated applications of the left modal rules but also forbid the application of the right modal rule on $\Box\varphi$ when φ is already present in the right-hand side of some sequent, *but also* when φ has been present in a right-hand side of a previously encountered hypersequent. We incorporate this condition in our version of the right modal rule, which can be formulated in a local manner thanks to the inclusion of the (otherwise superfluous) gray formulas.

▶ **Proposition 21.** *The rules of Figure 1 allow terminating top-down proof-search.*

Proof. Consider an initial hypersequent $\mathcal{H}$ and an attempt at deriving $\mathcal{H}$ by applying any rule of conclusion $\mathcal{H}$, and then recursively deriving the obtained premisses in the same manner. We only impose a *progress* condition: at any point in this process, the applied rule must produce as premisses hypersequents that are all distinct from the rule's conclusion – this forbids, e.g., the repeated application of a propositional rule on the same formula of the same sequent. This proof-search attempt may eventually succeed by deriving an hypersequent using an initial rule, or stop. Our concern here is to show that it cannot run forever.

It is clear from our rules that any hypersequent $\mathcal{H}'$ arising in this process can only be formed from subformulas of the initial hypersequent $\mathcal{H}$. Moreover, if at any point in the derivation some formula φ appears on the left-hand (resp. right-hand) side of a sequent, it will remain present on the left-hand (resp. right-hand) side of some sequent in all hypersequents of that subderivation.

In particular, this means that the right modal rule can only be applied on a finite number of distinct formulas. Applying the rule on $\Box\varphi$ creates a new sequent with φ on its right-hand side, which will remain present in the rest of the derivation. Therefore, the right modal rule cannot be applied again on the same formula in that subderivation, due to its side condition. Thus, the number of applications of the right modal rule is bounded in any branch of the proof-search process by the number of boxed subformulas of $\mathcal{H}$.

Now, consider the application of a rule other than the right modal rule, with conclusion $\mathcal{H}'$ and $\mathcal{H}''$ as one of its premise. We observe that $|\mathcal{H}''| \leq |\mathcal{H}'|$ – the inequality can be strict, when the addition of a new formula in a sequent of $\mathcal{H}'$ results in a sequent that was already

present in $\mathcal{H}'$. Moreover, for any sequent $\Gamma' \vdash \Delta'$ of $\mathcal{H}''$, there exists a sequent $\Gamma \vdash \Delta$ of $\mathcal{H}'$ such that $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$. Therefore, the repeated application of rules other than the right modal, subject to our progress condition, can only go on for at most $2 \times n \times k$ where n is the number of sequents in the initial hypersequent $\mathcal{H}$ and k is the number of subformulas of $\mathcal{H}$.

Because proof-search is bounded between any two applications of the right modal rule, and the number of such applications is itself bounded, the whole proof-search process must terminate. ◀

As is standard, we will combine this proof-search termination result with the invertibility of rules to obtain the completeness of our calculus. We rely on *semantic* invertibility, i.e. the validity of the conclusion of a rule implies the validity of all of its premisses. Recall that an hypersequent $\mathcal{H}$ is valid when its formula interpretation is in S5, which is equivalent to saying that it is satisfied in all worlds of all clique Kripke structures. Thus it is not valid iff it has a *counter-model*, that is a clique Kripke structure $\mathcal{K}$ such that, for each sequent $\Gamma \vdash \Delta$ of $\mathcal{H}$, there exists a world w of $\mathcal{K}$ which satisfies all formulas of Γ but no formula of Δ (we say that w falsifies $\Gamma \vdash \Delta$). Such clique counter-models provide a convenient tool for proving the following result, by the contrapositive (see Section C for details).

▶ **Proposition 22.** *The rules of Figure 1 are invertible.*

▶ **Theorem 23.** *The calculus of Figure 1 is sound and complete wrt. S5.*

Proof. Soundness is easily verified by checking that each rule is sound – or observing that our rules can be obtained from Poggiolesi's and contraction.

For completeness, we show that, for some appropriate proof-search procedure, proof-search fails on a hypersequent $\mathcal{H}$ only when this hypersequent has a counter-model. To obtain this result we only need to assume that proof-search is progressing (cf. proof of Proposition 21) and that it only stops when no rule applies – we do not even need to assume that it applies initial rules eagerly, though of course this would make proof-search more efficient. By the previous result, this proof-search is terminating. In case of failure, it yields a finite partial derivation, i.e. a proof tree featuring at least one unjustified leaf $\mathcal{H}'$, which cannot be the conclusion of any rule. We shall exhibit a counter-model for $\mathcal{H}'$, which, by invertibility, will prove that $\mathcal{H}$ has a counter-model as expected.

We thus construct a counter-model for $\mathcal{H}'$, exploiting the fact that any rule instance whose conclusion is $\mathcal{H}'$ would also have $\mathcal{H}'$ as one of its premisses. We consider the Kripke clique over worlds $\{w_{\Gamma\vdash\Delta} \mid (\Gamma \vdash \Delta) \in \mathcal{H}'\}$, where $w_{\Gamma\vdash\Delta} \models p$ iff $p \in \Gamma$. We verify, for any world $w_{\Gamma\vdash\Delta}$, that $w_{\Gamma\vdash\Delta} \models \varphi$ for any $\varphi \in \Gamma$, and $w_{\Gamma\vdash\Delta} \not\models \varphi$ for any $\varphi \in \Delta$. This is proved by induction over φ. The cases for $\bot$ and atoms is immediate. Assume now that φ is of the form $\varphi_1 \Rightarrow \varphi_2$:

- If $\varphi \in \Gamma$ then, because the implication left rule would have $\mathcal{H}'$ itself as premise, it must be that either $\varphi_2 \in \Gamma$ or $\varphi_1 \in \Delta$. In either case we conclude, by induction hypotheses on φ_1 and φ_2, that $w_{\Gamma\vdash\Delta} \models \varphi$.
- Similarly, if $\varphi \in \Delta$, we obtain that $\varphi_1 \in \Gamma$ and $\varphi_2 \in \Delta$ by hypothesis on $\mathcal{H}'$, and conclude by induction hypotheses on φ_1 and φ_2.

Finally, consider the case where φ is some $\Box\varphi'$:

- If $\varphi \in \Gamma$, then because the left modal rules yield $\mathcal{H}'$ as a premise, the subformula φ' is on the left-hand side of all sequents of $\mathcal{H}'$, thus $w_{\Gamma\vdash\Delta} \models \varphi$.
- If $\varphi \in \Delta$, the right modal rule does not apply without creating a repetition, thus there exists a sequent of $\mathcal{H}'$ which has φ' on its right-hand side, providing a world w' which does not satisfy φ', hence $w_{\Gamma\vdash\Delta} \not\models \varphi$. ◀

$$\frac{}{\mathcal{H} \mid \Gamma, \varphi \vdash \Delta \mid \Gamma' \vdash \varphi, \Delta'} \ \varphi \text{ determined}$$

Figure 3 Additional axiom rule for determined formulas.

4.2 Reasoning on determined formulas in hypersequent calculus

Now that we have presented Poggiolesi's hypersequent calculus for S5, in a slightly reformulated form, we show that it can easily adapted to elegantly incorporate reasoning on determined formulas. In the context of the logic of overwhelming truth, way say that a formula φ is determined if it is either overwhelmingly true or overwhelmingly false (i.e. negligibly true). In our modal language, this can simply be expressed as $\Box\varphi \vee \Box\neg\varphi$. When reasoning about cryptographic protocols, determined formulas often arise, and it is necessary to take this property into account to carry out formal proofs. We are thus interested in designing proof systems that can take the determined character of formulas into account.

To formally define our problem, we first parameterize our logic with a subset of atoms $\mathcal{D}$ that are assumed to be determined – this is a modelling assumption. We consider the problem of deciding whether a formula φ is a consequence of $\{\Box d \vee \Box\neg d \mid d \in \mathcal{D}\}$ – of course, this can also be phrased in terms of cryptographic structures or clique Kripke structures by the previous results. More precisely, we would like to find a proof system with nice proof-search behaviour for proving such formulas.

▶ **Example 24.** Take $\mathcal{D} = \{q\}$ for some $q \in \mathcal{P}$. Then $\Box(q \vee \psi) \Rightarrow \Box q \vee \Box\psi$ is valid for any ψ. Moreover, for some $p \in \mathcal{P} \setminus \mathcal{D}$, we have $\Box\varphi \vee \Box\neg\varphi$ for $\varphi \in \{q \Rightarrow \bot,\ \Box p,\ q \wedge (p \Rightarrow p)\}$. For $\varphi = (p \Rightarrow q)$, $\Box\varphi \vee \Box\neg\varphi$ is however not valid.

A naive solution to our problem can be obtained from any complete proof system for our modal logic: it suffices to look for proofs of $(\bigwedge_{d\in\mathcal{D}} \Box d \vee \Box\neg d) \Rightarrow \varphi$, using the proof system at hand (assuming wlog. that $\mathcal{D}$ is finite). This is however unsatisfactory, as the addition of the hypotheses $\Box d \vee \Box\neg d$ would yield an explosion of the size of proofs in a typical proof-search, as is the case with our hypersequent calculus.

We propose a better solution, where the determined character of formulas is only used when relevant. Strikingly, it is obtained by adding to the rules of Figure 1 a single rule, shown in Figure 3: instead of adding hypotheses expressing $\mathcal{D}$ in the conclusion hypersequent, we incorporate $\mathcal{D}$ in a modified axiom rule. As with the standard axiom rule, our modified axiom rule can be restricted to the case where φ is atomic, without losing completeness – in that case, the side condition is equivalent to requiring that $\varphi \in \mathcal{D}$. In fact, it seems that any reasonable use of this rule in a proof-search would rely on a side condition that is easier to verify than the fact that φ is determined: although this is decidable, it is costly; a syntactic criterion for detecting simple determined formulas can be used instead, e.g. checking that the formula is built using propositional connectives from atoms in $\mathcal{D}$ and boxed formulas.

▶ **Theorem 25.** *The proof system of Figure 1 augmented with the rule of Figure 3 is sound and complete for the logic of overwhelming truth with determined formulas in $\mathcal{D}$: a hypersequent $\mathcal{H}$ is derivable in this system iff the formula interpretation of $\mathcal{H}$ is valid in all structures satisfying $\Box\psi \vee \Box\neg\psi$ for all $\psi \in \mathcal{D}$.*

Proof. For soundness, it suffices to observe that the conclusion of our new rule cannot have counter-models consistent with $\mathcal{D}$: we would then have a world satisfying ψ and another satisfying $\neg\psi$.

For completeness, we adapt the argument provided for S5 in the previous section, to show that any hypersequent for which proof-search fails admits a counter-model. We do this with a proof-search procedure that makes use of our additional rule, and ensure that in that case the counter-model is consistent with $\mathcal{D}$. Clearly, proof-search still terminates in our system, and rules are still invertible: all we need to do is adapt the counter-model construction for an hypersequent $\mathcal{H}'$ on which proof-search cannot conclude nor progress. We adapt the construction of Theorem 23, taking the same set of worlds as before, but setting $\mathcal{K}, w_{\Gamma\vdash\Delta} \models p$ iff $p \in \Gamma$ or $p \in \mathcal{D} \cap \mathrm{lhs}(\mathcal{H}')$. In this way, we ensure that the atoms in $\mathcal{D}$ are determined in $\mathcal{K}$. We then verify, for any world $w_{\Gamma\vdash\Delta}$, that $w_{\Gamma\vdash\Delta} \models \varphi$ for any $\varphi \in \Gamma$, and $w_{\Gamma\vdash\Delta} \not\models \varphi$ for any $\varphi \in \Delta$. This is proved by induction on φ as before, and only the case where φ is an atom p is modified:

- If $p \in \Gamma$, we have $w_{\Gamma\vdash\Delta} \models p$ by construction.
- If $p \in \Delta$ and $p \notin \mathcal{D}$, we know that the axiom rule of Figure 1 does not apply, hence $p \notin \Gamma$, from which $w_{\Gamma\vdash\Delta} \not\models p$ follows.
- If $p \in \Delta$ and $p \in \mathcal{D}$, we know that the axiom rules of Figure 1 and Figure 3 do not apply, hence $p \notin \mathrm{lhs}(\mathcal{H}')$ and $w_{\Gamma\vdash\Delta} \not\models p$. ◀

5 Sequent calculus for a non-nested logic of overwhelming truth

In this section, we consider a fragment of modal formulas that corresponds to the propositional fragment of the logic underlying Squirrel [5]. Essentially, it is obtained by forbidding nested modalities and requiring that all atoms occur under modalities. In the style of [5], we present these restrictions by organizing formulas into *local* and *global* ones: local formulas may contain atoms but no modalities; global formulas may not contain atoms but can contain $\Box\varphi$ subformulas, under the condition that φ is local.

▶ **Definition 26.** *We define local formulas (denoted by φ, ψ) and global formulas (denoted by F, G) by the following grammar:*

$$\varphi, \psi ::= \bot \mid p \mid \varphi \Rightarrow \psi \qquad F, G ::= \bot \mid \Box\varphi \mid F \Rightarrow G$$

▶ **Example 27.** The formula $\Box(p \wedge q) \Rightarrow \Box p$ is a global formula. The formula $\Box p \Rightarrow p$ is neither a local nor a global formula.

To understand why the move to global formulas is not a strong restriction, it is useful to recall the well-known fact that nested modalities do not bring extra expressiveness in S5. The next proposition, proved in Section D, states this result more precisely.

▶ **Proposition 28.** *For any modal formula φ, there exist families of* propositional *formulas $(\psi_i)_i$, $(\theta_{i,j})_{i,j}$, $(\chi_{i,j})_{i,j}$ such that φ and the following formula are equivalent in all cryptographic structures (or, equivalently, in all clique Kripke structures):*

$$\bigwedge_{i=1}^{n} \left(\psi_i \vee \left(\bigvee_{j=1}^{l_i} \Box\chi_{i,j} \right) \vee \left(\bigvee_{k=1}^{m_i} \neg\Box\theta_{i,k} \right) \right)$$

Moreover, if φ is a boxed formula, then $\psi_i = \bot$ for all i.

▶ **Example 29.** The formula $\Box(\Box p \Rightarrow q)$ is equivalent to the global formula $\neg\Box p \vee \Box q$.

Because the validity of φ and that of $\Box\varphi$ are equivalent, checking the validity of arbitrary modal formulas can be reduced to checking the validity of global formulas. The catch, however, is that the transformation underlying Proposition 28 may induce an exponential

Global rules

$$\frac{}{\Theta, F \vdash F, \Pi} \qquad \frac{}{\Theta, \bot \vdash \Pi}$$

$$\frac{\Theta \vdash F, \Pi \quad \Theta, G \vdash \Pi}{\Theta, F \Rightarrow G \vdash \Pi} \qquad \frac{\Theta, F \vdash G, \Pi}{\Theta \vdash F \Rightarrow G, \Pi}$$

Local rules

$$\frac{}{\Theta; \Gamma, \varphi \vdash \varphi, \Delta} \qquad \frac{}{\Theta; \Gamma, \bot \vdash \Delta}$$

$$\frac{\Theta; \Gamma \vdash \varphi, \Delta \quad \Theta; \Gamma, \psi \vdash \Delta}{\Theta; \Gamma, \varphi \Rightarrow \psi \vdash \Delta} \qquad \frac{\Theta; \Gamma, \varphi \vdash \psi, \Delta}{\Theta; \Gamma \vdash \varphi \Rightarrow \psi, \Delta}$$

Mixed rules

$$\frac{\Theta; \cdot \vdash \varphi}{\Theta \vdash \Box\varphi, \Pi} \qquad \frac{\Theta; \Gamma, \varphi \vdash \Delta}{\Theta, \Box\varphi; \Gamma \vdash \Delta}$$

Figure 4 A sequent calculus for the global logic of overwhelming truth.

blowup. However, this is not a concern in the context of Squirrel, where formulas are given as local and global ones from the beginning: it appears that these fragments are natural for specifying and reasoning about cryptographic protocols.

We now present a simple sequent calculus proof system for global formulas. It involves sequents featuring several kinds of sets of formulas: we will use the letters Γ and Δ to denote a set of *local* formulas, and the letters Θ and Π for sets of *global* formulas.

▶ **Definition 30.** *A* global *sequent* $\Theta \vdash \Pi$ *is formed from two sets of global formulas* Θ *and* Π. *A* local *sequent* $\Theta; \Gamma \vdash \Delta$ *is formed from a set of global formulas* Θ *and two sets of local formulas* Γ *and* Δ.

▶ **Definition 31.** *The* formula interpretation *of a global sequent* $\Theta \vdash \Pi$ *is the formula* $\bigwedge_{F \in \Theta} F \Rightarrow \bigvee_{G \in \Pi} G$. *The* formula interpretation *of a local sequent* $\Theta; \Gamma \vdash \Delta$ *is:*

$$\bigwedge_{F \in \Theta} F \Rightarrow \Box \left(\bigwedge_{\varphi \in \Gamma} \varphi \Rightarrow \bigvee_{\psi \in \Delta} \psi \right)$$

The rules of our system are shown in Figure 4. The global rules are the usual rules of classical sequent calculus, and local rules are straightforward adaptations of the same rules for local sequents, occurring under the global context Θ. Mixed rules articulate the two kinds of sequents. The first one allows to derive a global sequent from a local one: considering the formula interpretation of sequents, this rule is a mere weakening; its application requires to choose one formula on the right-hand side, and put it under focus by moving it to a local sequent, forgetting at this point all other formulas from the conclusion's right-hand side. The second mixed rule allows to derive a local sequent with a global hypothesis $\Box\varphi$ from a local sequent with a local hypothesis φ. Logically speaking, it is essentially the K axiom: focusing on the relevant parts of the formula interpretation of our sequents, we are deriving

$\Box\varphi \Rightarrow \Box(\Gamma \Rightarrow \Delta)$ from $\Box(\varphi \Rightarrow \Gamma \Rightarrow \Delta)$. It is worthwhile to note that the $\Box\varphi$ formula is not kept in this rule's premise. Overall, our rules do not embed any strong principles underlying S5, yet as we shall see they form a complete system for our fragment of S5.

The proof system underlying Squirrel uses global and local sequents similar to ours. One difference is that Squirrel's sequents have a single conclusion, and its proof system is given in a natural deduction style. This choice has been motivated by usability considerations: users of the Squirrel proof assistant may not be experts in logic, and might have found multiple-conclusion sequents confusing. As a consequence, Squirrel's proof system features *reductio ad absurdum* rules at both the local and global levels [5]. Beyond this common difference of style, both proof systems share an apparent weakness. As observed above, our first mixed rule is not invertible. Relatedly, Squirrel's proof system does not allow to perform a mixed *reduction ad absurdum*, shown next:

$$\frac{\Theta, \neg\Box(\Gamma \Rightarrow \varphi) \vdash \bot}{\Theta; \Gamma \vdash \varphi}$$

We do not imply that such a rule should be considered, but its absence essentially forces the same kind of strong choices in Squirrel proofs as our first mixed rule.

We now prove the key lemma explaining the completeness of our proof system.

▶ **Proposition 32.** *Let φ and $(\psi_j)_{j\in[1;n]}$ be some propositional formulas such that $\Box\varphi \Rightarrow \vee_{j\in[1;n]}\Box\psi_j$ is valid. There exists $k \in [1;n]$ such that $\varphi \Rightarrow \psi_k$ is a propositional tautology.*

Proof. If φ is propositionally unsatisfiable, $\varphi \Rightarrow \psi_k$ is a propositional tautology for any k, hence the result holds.

Assume now that φ is propositionally satisfiable, and consider the clique Kripke structure $\mathcal{K}$ over the non-empty set of worlds $W = \{\ \nu : \mathcal{P} \to \{0,1\} \mid \nu \models \varphi\}$. In words, the worlds of $\mathcal{K}$ are the propositional interpretations ν that satisfy the (propositional) formula φ. We set the propositional variables satisfied by the world ν to be the ones indeed satisfied by ν. It immediately follows that $\mathcal{K}, \nu \models \varphi$ for all $\nu \in W$, and thus $\mathcal{K}, \nu \models \Box\varphi$ for all $\nu \in W$.

Let F be the global formula $\Box\varphi \Rightarrow \vee_{j\in[1;n]}\Box\psi_j$. Let ν be an arbitrary element of W. By hypothesis, F is valid, hence $\mathcal{K}, \nu \models F$. Since $\mathcal{K}, \nu \models \Box\varphi$, we conclude that there exists k such that $\mathcal{K}, \nu \models \Box\psi_k$. This means that $\mathcal{K}, \nu' \models \psi_k$ for all $\nu' \in W$. In other words, the propositional formula ψ_k is satisfied in all propositional interpretations that satisfy φ: $\varphi \Rightarrow \psi_k$ is propositionally valid. ◀

▶ **Theorem 33.** *The rules of Figure 4 are sound and complete: a global (resp. local) sequent is derivable iff its formula interpretation is a theorem of the modal logic of overwhelming truth (or, equivalently, of S5).*

Proof. Soundness is easily verified; we only detail the completeness argument. We proceed by induction on the number of connectives of the sequents. Any (local or global) sequent featuring an implication formula or the $\bot$ constant at toplevel is the conclusion of one of our propositional rules. These rules are invertible and yield premises with one less logical connective, allowing to conclude by induction hypothesis.

Next, consider a valid global sequent of the form $\Box\varphi_1, \ldots, \Box\varphi_m \vdash \Box\psi_1, \ldots, \Box\psi_n$. Note that its formula interpretation is equivalent to $\Box\varphi \Rightarrow \Box\psi_1 \vee \ldots \vee \Box\psi_n$ for $\varphi = \wedge_i\varphi_i$. By Proposition 32, there exists k such that $\varphi \Rightarrow \psi_k$ is a propositional tautology. Equivalently, $\cdot; \varphi_1, \ldots, \varphi_m \vdash \psi_k$ is derivable in our system using only local rules. From there we can derive our initial sequent using the first mixed rule to select ψ_k and m applications of the second mixed rule to transfer the $\Box\varphi_i$ global hypotheses to the local context.

A similar argument can be used to show that any valid local sequent of the form $\Box\varphi_1, \ldots, \Box\varphi_m; \Gamma \vdash \Delta$ can be derived in our system. This concludes the proof. ◀

6 Discussion

We have defined the modal logic of overwhelming truth, and shown that it coincides with S5, and that this result remains for several variants of the logic. Next, we have shown that Poggiolesi's CCS5$_s$ hypersequent calculus for S5 can be elegantly adapted to incorporate reasoning over determined formulas – a problem that arises naturally when carrying out formal proof involving both probabilistic and deterministic formulas. In passing, we have refined the analysis of proof-search in Poggiolesi's system, providing clean foundations for semantic proofs of completeness in that setting. Finally, considering the fragment of global formulas (that is, the propositional fragment of Squirrel's logic) we have shown that the hypersequent structure is not necessary, providing instead a sound and complete proof system based on simple (local and global) sequents. Overall, our work provides the first completeness results for logics in the CCSA line of work.

Related works. The idea of giving a probabilistic semantics to propositional or modal logics is not new; see e.g. [17] for a survey. For instance, [19] considers a modal logic where a threshold $t \in [0; 1]$ is chosen and $\Box\varphi$ reads as "φ holds with probability more than t". With this interpretation, $(\Box\varphi \wedge \Box\psi) \Rightarrow \Box(\varphi \wedge \psi)$ is not valid, while it is in our overwhelming truth semantics. Charles G. Morgan also worked on developping a characterization of many logics in terms of conditional probabilities [23, 24]. In particular, he provided an axiomatization of conditional probability measures that is sound and complete wrt. S5. Our approach and motivation is different: we start from a concrete probabilistic semantics and seek to characterize the resulting logic in terms of modal logic axioms.

There has been much work on designing well-structured proof systems for S5, culminating but not ending with Poggiolesi's [25]. While hypersequents provide a satisfying answer to this quest, it appears that this is not possible using simple sequent calculus, at least not without making some sacrifices. We note the work of Indrzejczak [20] which proposes a simple sequent calculus for S5, which he essentially obtains by incorporating into his sequent calculus some rewrite rules that simplify modal formulas to removed nested modalities. This is related in spirit to the results of Section 5, where we propose a simple proof system for formulas that may result from such rewriting. However, our proof system does not feature more rewriting, and thus notably satisfies the subformula property. Indrzejczak's work does not seem to yield a sub-system with such properties, but it would be interesting to try to obtain such a result.

Directions for future work. We have only established weak completeness results, at the level of validity, and it would be interesting to investigate whether the modal logic of overwhelming truth coincides with S5 at the level of logical implications. This would yield the strong completeness for the hypersequent calculus wrt. the logic of overwhelming truth, as well as compacity for that logic.

We have used semantical arguments to establish several completeness results, refining in particular Poggiolesi's proof-search analysis to obtain semantical proofs of completeness for hypersequent calculi. It is also possible to establish the completeness of our hypersequent calculus with special axiom for determined formulas, using syntactic methods. Looking for a syntactic proof of completeness for our simple sequent calculus would also be worthwhile.

In both cases, syntactic arguments might provide results that carry over more smoothly to richer fragments, e.g. the first-order case. More generally, it would naturally be desirable to extend our results to richer fragments of the logic of overwhelming truth: one possibility is to extend our language of formulas to the first-order case; it is also possible to stick to the propositional case but incorporate the computational indistinguishability predicate of the CCSA logics. Both directions seem very challenging.

References

1 Carmine Abate, Philipp G. Haselwarter, Exequiel Rivas, Antoine Van Muylder, Théo Winterhalter, Catalin Hritcu, Kenji Maillard, and Bas Spitters. Ssprove: A foundational framework for modular cryptographic proofs in coq. In *CSF*, pages 1–15. IEEE, 2021. doi:10.1109/CSF51468.2021.00048.

2 David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutsos, and Solène Moreau. An interactive prover for protocol verification in the computational model. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*, pages 537–554. IEEE, 2021. doi:10.1109/SP40001.2021.00078.

3 David Baelde, Stéphanie Delaune, Adrien Koutsos, and Solène Moreau. Cracking the stateful nut: Computational proofs of stateful security protocols using the squirrel proof assistant. In *CSF*, pages 289–304. IEEE, 2022. doi:10.1109/CSF54842.2022.9919665.

4 David Baelde, Caroline Fontaine, Adrien Koutsos, Guillaume Scerri, and Théo Vignon. A Probabilistic Logic for Concrete Security. In *CSF 2024 - 37th IEEE Computer Security Foundations Symposium*, Enschede, Netherlands, July 2024. URL: https://hal.science/hal-04577828.

5 David Baelde, Adrien Koutsos, and Joseph Lallemand. A Higher-Order Indistinguishability Logic for Cryptographic Reasoning. In *LICS'23*. ACM, 2023. URL: https://inria.hal.science/hal-03981949.

6 Gergei Bana, Rohit Chadha, and Ajay Kumar Eeralla. Formal analysis of vote privacy using computationally complete symbolic attacker. In *ESORICS (2)*, volume 11099 of *Lecture Notes in Computer Science*, pages 350–372. Springer, 2018. doi:10.1007/978-3-319-98989-1_18.

7 Gergei Bana and Hubert Comon-Lundh. Towards unconditional soundness: Computationally complete symbolic attacker. In *International Conference on Principles of Security and Trust*, pages 189–208. Springer, 2012. doi:10.1007/978-3-642-28641-4_11.

8 Gergei Bana and Hubert Comon-Lundh. A computationally complete symbolic attacker for equivalence properties. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 609–620. ACM, 2014. doi:10.1145/2660267.2660276.

9 Manuel Barbosa, Gilles Barthe, Karthik Bhargavan, Bruno Blanchet, Cas Cremers, Kevin Liao, and Bryan Parno. Sok: Computer-aided cryptography. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 777–795, 2021. doi:10.1109/SP40001.2021.00008.

10 Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. Computer-aided security proofs for the working cryptographer. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 71–90. Springer, 2011. doi:10.1007/978-3-642-22792-9_5.

11 David A. Basin, Andreas Lochbihler, and S. Reza Sefidgar. Crypthol: Game-based proofs in higher-order logic. *J. Cryptol.*, 33(2):494–566, 2020. doi:10.1007/S00145-019-09341-Z.

12 Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal logic*, volume 53. Cambridge University Press, 2001. doi:10.1017/CBO9781107050884.

13 Bruno Blanchet, Martín Abadi, and Cédric Fournet. Automated verification of selected equivalences for security protocols. *The Journal of Logic and Algebraic Programming*, 75(1):3–51, 2008. doi:10.1016/J.JLAP.2007.06.002.

14 Hubert Comon and Adrien Koutsos. Formal computational unlinkability proofs of RFID protocols. In *CSF*, pages 100–114. IEEE Computer Society, 2017. doi:10.1109/CSF.2017.9.

15 Hubert Comon-Lundh, Véronique Cortier, and Guillaume Scerri. Tractable inference systems: An extension with a deducibility predicate. In Maria Paola Bonacina, editor, *Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings*, volume 7898 of *Lecture Notes in Computer Science*, pages 91–108. Springer, 2013. doi:10.1007/978-3-642-38574-2_6.

16 Cas Cremers, Caroline Fontaine, and Charlie Jacomme. A logic and an interactive prover for the computational post-quantum security of protocols. In *SP*, pages 125–141. IEEE, 2022. doi:10.1109/SP46214.2022.9833800.

17 Lorenz Demey, Barteld Kooi, and Joshua Sack. Logic and Probability. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2019 edition, 2019. URL: https://plato.stanford.edu/archives/sum2019/entries/logic-probability/.

18 Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. doi:10.1016/0022-0000(84)90070-9.

19 Charles L Hamblin. The modal" probably". *Mind*, 68(270):234–240, 1959.

20 Andrzej Indrzejczak. Simple Decision Procedure for S5 in Standard Cut-Free Sequent Calculus. *Bulletin of the Section of Logic*, 45(2), June 2016. doi:10.18778/0138-0680.45.2.05.

21 Adrien Koutsos. The 5G-AKA authentication protocol privacy. In *EuroS&P*, pages 464–479. IEEE, 2019. doi:10.1109/EUROSP.2019.00041.

22 Adrien Koutsos. Decidability of a sound set of inference rules for computational indistinguishability. In *32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019*, pages 48–61. IEEE, 2019. doi:10.1109/CSF.2019.00011.

23 Charles G. Morgan. Simple Probabilistic Semantics for Propositional K, T, B, S4, and S5. *Journal of Philosophical Logic*, 11(4):443–458, 1982. URL: https://www.jstor.org/stable/30226261, doi:10.1007/BF00284979.

24 Charles G. Morgan. There Is a Probabilistic Semantics for Every Extension of Classical Sentence Logic. *Journal of Philosophical Logic*, 11(4):431–442, 1982. URL: https://www.jstor.org/stable/30226260, doi:10.1007/BF00284978.

25 Francesca Poggiolesi. A cut-free simple sequent calculus for modal logic S5. *The Review of Symbolic Logic*, 1(1):3–15, 2008. doi:10.1017/S1755020308080040.

26 Guillaume Scerri and Ryan Stanley-Oakes. Analysis of key wrapping apis: Generic policies, computational security. In *CSF*, pages 281–295. IEEE Computer Society, 2016. doi:10.1109/CSF.2016.27.

27 Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *cryptology eprint archive*, 2004.

A Proofs of Section 2

▶ **Proposition 7.** *The abstract logic of overwhelming truth is a* normal modal logic*: its theorems are closed under substitution and* modus ponens*; they contain classical tautologies and the K axiom* $\Box(p \Rightarrow q) \Rightarrow \Box p \Rightarrow \Box q$*; moreover,* $\Box\varphi$ *is a theorem whenever* φ *is.*

Proof. It is clear, by definition of our semantics, that the set of theorems of our logic is closed under substitution. It is also obvious that the validity of φ entails that of $\Box\varphi$.

Next, we verify that any classical tautology φ is a theorem of our logic. For any $\mathcal{S}$, η and $\rho \in X_\eta^{\mathcal{S}}$, notice that $[\![\varphi]\!]_{\mathcal{S}}(\eta,\rho)$ only relies on the interpretation of its subformulas for the same values of η and ρ. In other words, $[\![\varphi]\!]_{\mathcal{S}}(\eta,\rho)$ can be seen as the classical interpretation of φ in the boolean structure $\mathcal{S}(\eta,\rho)$ defined by $p^{\mathcal{S}(\eta,\rho)} = p^{\mathcal{S}}(\eta,\rho)$. Because φ is a tautology, we thus have $[\![\varphi]\!]_{\mathcal{S}} = \mathbf{1}$, hence $[\![\varphi]\!]_{\mathcal{S}}$ is overwhelming in any $\mathcal{S}$.

The K axiom is valid: indeed, if both $[\![p \Rightarrow q]\!]_{\mathcal{S}}$ and $[\![p]\!]_{\mathcal{S}}$ are overwhelming in a structure $\mathcal{S}$, then so is $[\![q]\!]_{\mathcal{S}}$ because $\Pr([\![q]\!]_{\mathcal{S}} = 0) \leq \Pr([\![p \Rightarrow q]\!]_{\mathcal{S}} = 0) + \Pr([\![p]\!]_{\mathcal{S}} = 0)$ and the sum of negligible functions is negligible. By the same argument, theorems are closed under modus ponens. ◀

B Proofs of Section 3

▶ **Lemma 11.** *All S5 theorems are theorems of the modal logics of overwhelming truth.*

Proof. Since modal logics of overwhelming truth are normal, it suffices to check that the three axioms defining S5 are valid wrt. arbitrary cryptographic structures.

- Axiom 4 is obvious, as $[\![\Box p]\!]_{\mathcal{S}} = [\![\Box\Box p]\!]_{\mathcal{S}}$ for all $\mathcal{S}$.
- For axiom T, consider an arbitrary $\mathcal{S}$, and proceed by case analysis on whether $[\![p]\!]_{\mathcal{S}}$ is overwhelming. If this is the case, then $[\![\Box p]\!]_{\mathcal{S}} = 1$ thus $[\![\Box p \Rightarrow p]\!]_{\mathcal{S}} = [\![p]\!]_{\mathcal{S}}$, which is overwhelming by hypothesis. Otherwise, $[\![\Box p]\!]_{\mathcal{S}} = 0$ and $[\![\Box p \Rightarrow p]\!]_{\mathcal{S}} = 1$ is overwhelming.
- For axiom 5, note that $[\![\Diamond\varphi]\!]_{\mathcal{S}} = 1$ when $[\![\varphi]\!]$ is non-negligible, and 0 otherwise – in both cases, $[\![\Diamond\varphi]\!]_{\mathcal{S}}(\eta,\rho) = [\![\Diamond\varphi]\!]_{\mathcal{S}}(\eta',\rho')$ for all η,ρ,η',ρ'. Thus $[\![\Box\Diamond p]\!]_{\mathcal{S}} = [\![\Diamond p]\!]_{\mathcal{S}}$, hence the result. ◀

▶ **Proposition 15.** *For any modal formula φ, the following conditions are equivalent:*

1. *φ is a theorem of S5;*
2. *φ is valid wrt. equivalence frames;*
3. *φ is valid wrt. clique frames.*
4. *φ is valid wrt. finite clique frames.*

Proof sketch. The equivalence between (1) and (2) is a classic result [12]. Equivalence between (2) and (3) is an easy observation: for any equivalence structure $\mathcal{K}$ and $w \in W^{\mathcal{K}}$, let $\mathcal{K}_w$ be the (clique) substructure corresponding to the equivalence class of w in $\mathcal{K}$; it is easy to see that, for any φ, $\mathcal{K}, w \models \varphi$ is equivalent to $\mathcal{K}_w, w \models \varphi$. Finally, the equivalence between (3) and (4) is obtained by using the filtration technique [12]: given any initial formula φ, it allows to extract from a clique structure $\mathcal{K}$ a finite clique structure $\mathcal{K}'$ whose worlds are equivalence classes of worlds of $\mathcal{K}$, such that $\mathcal{K}, w \models \psi$ is equivalent to $\mathcal{K}', [w] \models \psi$ for any subformula ψ of φ. ◀

C Proofs of Section 4

▶ **Proposition 22.** *The rules of Figure 1 are invertible.*

Proof. We prove the contrapositive: assuming a counter-model of a premise $\mathcal{H}'$ of some rule instance, we build a counter-model of the conclusion $\mathcal{H}$. This is immediate for all rules except the right modal rule. We thus consider the case where $\mathcal{H}$ is of the form $\mathcal{H}_0 \mid \Gamma \vdash \Box\varphi, \Delta$ and $\mathcal{H}'$ is of the form $\mathcal{H}_0 \mid \Gamma \vdash \Box\varphi, \Delta \mid \cdot \vdash \varphi$. By hypothesis, we have a counter-model of $\mathcal{H}$, that is a Kripke structure $\mathcal{K}$ with worlds $\vec{w}$ falsifying the sequents of $\mathcal{H}_0$, and a world w' falsifying $\Gamma \vdash \Box\varphi, \Delta$. In particular, $w' \not\models \Box\varphi$, hence there exists w'' such that $w'' \not\models \varphi$. Thus, $\mathcal{K}$, together with the worlds $\vec{w}, w'$ and w'', provide a counter-model for $\mathcal{H}'$. ◀

D Proofs of Section 5

▶ **Lemma 34.** *Let φ and ψ be any modal formulas. Then for Kripke equivalence models:*

1. $\Box(\varphi \wedge \psi) \equiv \Box\varphi \wedge \Box\psi$,
2. $\Box(\varphi \vee \Box\psi) \equiv \Box\varphi \vee \Box\psi$,
3. $\Box(\varphi \vee \neg\Box\psi) \equiv \Box\varphi \vee \neg\Box\psi$

Proof. Let $\mathcal{S} = (W, R, V)$ be an equivalence Kripke model, $w \in W$.

1. By definition, $(\mathcal{S}, w) \models \Box(\varphi \wedge \psi)$ iff for all successor v of w by R, $(\mathcal{S}, v) \models \varphi \wedge \psi$ iff $(\mathcal{S}, v) \models \varphi$ and $(\mathcal{S}, v) \models \psi$.
   This is the case if and only if for all successor v of w by R, $(\mathcal{S}, v) \models \varphi$ and for all successor u, $(\mathcal{S}, u) \models \psi$, i.e. $(\mathcal{S}, w) \models \Box\varphi$ and $(\mathcal{S}, w) \models \Box\psi$, i.e. $(\mathcal{S}, w) \models \Box\varphi \wedge \Box\psi$, which concludes the proof.
2. Assume $(\mathcal{S}, w) \models \Box(\varphi \vee \Box\psi)$. Then for all successor v of w by R, $(\mathcal{S}, v) \models \varphi \vee \Box\psi$. Then there are two cases:
   - Either for all successor v of w by R, $(\mathcal{S}, v) \models \varphi$. In this case $(\mathcal{S}, w) \models \Box\varphi$.
   - Either there is a successor v of w by R such that $(\mathcal{S}, v) \not\models \varphi$. In this case, $(\mathcal{S}, v) \models \Box\psi$. Let u be a successor of w for R. Then since R is an equivalence relation, u is also a successor of v and $(\mathcal{S}, u) \models \psi$.
     Since this holds for all successors of w, $(\mathcal{S}, w) \models \Box\psi$.

   Now for the converse, assume $(\mathcal{S}, w) \models \Box\varphi \vee \Box\psi$. Since R is transitive, $(\mathcal{S}, w) \models \Box\varphi \vee \Box\Box\psi$. It is easy to check next that $(\mathcal{S}, w) \models \Box(\varphi \vee \Box\psi)$.
3. The proof in this case is very similar to the previous one, except in the proof of the converse. We have to use the transitivity as well as the symmetry of R to prove that $(\mathcal{S}, w) \models \Box\varphi \vee \neg\Box\psi$ implies $(\mathcal{S}, w) \models \Box\varphi \vee \Box\neg\Box\psi$. ◀

Proof of Proposition 28. We proceed by induction on φ, considering for this proof that the elementary propositional connectives are negation and conjunction. The case of propositional variables and the inductive case of conjunctions are clear using equivalence 1 of the previous lemma.

- **Case $\varphi = \neg\varphi'$**
  By induction hypothesis on φ' and using De Morgan's laws, we know that there are families of propositional formulas $(\psi_i), (\chi_{i,j}), (\theta_{i,j})$ such that

$$\varphi \equiv \bigvee_{i=1}^{n} \left(\neg\psi_i \wedge \left(\bigwedge_{j=1}^{l_i} \neg\Box\chi_{i,j} \right) \wedge \left(\bigwedge_{k=1}^{m_i} \Box\theta_{i,k} \right) \right).$$

  Then, by distributivity of disjunction over conjunction, there is an integer N such that

$$\varphi \equiv \bigwedge_{i=1}^{N} (\omega_{i,1} \vee \cdots \vee \omega_{i,h_i})$$

  where each $\omega_{i,j}$ is either a $\neg\psi_{i'}$, a $\Box\theta_{i',k'}$ or a $\neg\Box\chi_{i',j'}$.
  Finally by grouping for each i the $\neg\psi_{i'}$ into a single propositional formula, the $\Box\theta_{i',k'}$ together and the $\neg\Box\chi_{i',j'}$, we prove φ is equivalent to a formula having the desired shape.
- **Case $\varphi = \Box\varphi'$**
  By induction hypothesis on φ' and using the first equivalence in the previous lemma, we know there are families of propositional formulas $(\psi_i), (\chi_{i,j}), (\theta_{i,j})$ such that

$$\varphi \equiv \bigwedge_{i=1}^{n} \Box \left(\psi_i \vee \left(\bigvee_{j=1}^{l_i} \Box\chi_{i,j} \right) \vee \left(\bigvee_{k=1}^{m_i} \neg\Box\theta_{i,k} \right) \right).$$

  Then, applying equivalences 2 and 3 of the previous lemma on each one of the $\Box\chi_{i,j}$ and $\neg\Box\theta_{i,k}$, we have that

$$\varphi \equiv \bigwedge_{i=1}^{n} \left(\Box\psi_i \vee \left(\bigvee_{j=1}^{l_i} \Box\chi_{i,j} \right) \vee \left(\bigvee_{k=1}^{m_i} \neg\Box\theta_{i,k} \right) \right)$$

  which is what we wanted since for all i, the $\Box\psi_i$ can be grouped with the $\Box\chi_{i,j}$. ◀

Exponential Lower Bounds on Definable Fixed Points

Konstantinos Papafilippou
Department of Mathematics, University of Ghent, Belgium

David Fernández-Duque
Department of Philosophy, University of Barcelona, Spain

Abstract

It is known that the μ-calculus is no more expressive than basic modal logic over the class of finite partial orders, as well as over the class of finite, strict partial orders. Nevertheless, we show that the μ-calculus is exponentially more succinct, even when a reflexive modality is added as primitive. As corollaries, we obtain a lower bound for the fixed-point theorem for Gödel-Löb logic and a variant for Grzegorczyk logic, as well as lower bounds on interpolants for the interpolation theorem of Gödel-Löb logic.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Complexity theory and logic

Keywords and phrases Modal logic, Provability Logic, Fixed-Point Theorem, mu-calculus, Interpolation, Succinctness

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.25

Funding This work has been partially supported by FWO-FWF grant G030620N/I4513N and SNSF–FWO Lead Agency Grant 200021L_196176/G0E2121N.

1 Introduction

Expressivity and complexity are two crucial criteria in the design of formal languages for logical reasoning. However, these two properties alone only paint part of the picture, as a more *succinct* language may have advantages over an equally (or even more) expressive language if formula size is reduced sufficiently to considerably save on storage space and improve processing time. Two formal languages $\mathcal{L}_1$ and $\mathcal{L}_2$ may be equally expressive, yet certain properties may be expressed in $\mathcal{L}_1$ by much shorter expressions than in $\mathcal{L}_2$; when the size difference is e.g. exponential, it may dwarf any potential advantage offered by $\mathcal{L}_2$ on account of purely complexity-theoretic considerations.

Modal logics are an appealing framework for computational logic precisely due to the balance between expressivity and complexity, making them more adaptable than propositional logic but more tractable than first or higher order logic. But a well-informed choice of the "right" modal logic for a given task should also involve an understanding of how it fares in terms of succinctness.

In particular, Gödel-Löb logic (**GL**) provides a textbook example of a success story in modal logic: it is the logic of finite (or, more generally, converse well-founded) strict partial orders, hence it governs the behaviour of computational processes that terminate in finite time. It is obtained from the basic modal logic **K** by adding *Löb's axiom,* $\Box(\Box\varphi \to \varphi) \to \Box\varphi$. It is also the logic of provability in Peano arithmetic and related theories, as well as the logic of scattered topological spaces, granting it applications in foundations of mathematics and spatial reasoning. For the first setting, one interprets modal formulas as arithmetical statements, with variables representing arbitrary statements in the language of **PA** and $\Box\varphi$ being interpreted as Gödel's $\mathsf{Bew}(\ulcorner\varphi\urcorner)$, which formalises the statement that φ is provable in **PA**. Solovay [23] showed that the set of valid formulas (i.e., those that correspond to

theorems of **PA**) in this setting are precisely those provable in **GL**. For the second, one interprets $\Diamond$ as a topological Cantor derivative operator, e.g. $\Diamond\varphi$ is the set of limit points of those points satisfying φ. **GL** once again captures the set of validities in this context (see e.g. [6]).

Moreover, **GL** is remarkably well-behaved, being decidable, finitely axiomatizable, and enjoying Craig interpolation [8] and definable fixed points [21]. The latter in particular means that the μ-calculus adds no expressive power to **GL** [2]. This is also true for the class **Grz** of Grzegorczyk frames, based on *Noetherian* posets; essentially, the reflexive closures of **GL** frames [9]. **Grz** is also the logic of "provably true" over Peano arithmetic and is the greatest modal companion of intuitionistic propositional logic [26, 8] and is characterised by the axiom $\Box(\Box(\varphi \to \Box\varphi) \to \varphi) \to \Box\varphi$.

One could thus jump to the conclusion that the μ-calculus over finite posets (either reflexive or irreflexive) is not worth considering. However, such disinterest would be misguided, as it does not take questions of succinctness into account. The fixed-point theorem for **GL** states that for any formula $\varphi(x)$ where x occurs only in the scope of $\Box$ (or $\Diamond$), there is a formula ψ such that $\psi \leftrightarrow \varphi(\psi)$ is derivable. All proofs [7, 16, 20, 21, 22] yield some ψ that is at least exponentially larger than φ. This raises the question of whether this bound is optimal, to which we provide a positive answer. In contrast, the μ-calculus formula $\mu x.\varphi(x)$ yields a fixed point of φ and is only slightly larger than φ itself.[1] Thus we conclude that, despite fixed points already being definable in the basic modal language, there is much to be gained by passing to a language with explicit fixed point constructors.

Research in succinctness involves delicate techniques and it has been an active area in the last decades; see e.g. [17, 10, 1]. Closest to the present work, [11] show that over **GL** frames, a language with the reflexive modality $\diamondsuit$ is exponentially more succinct than a language with $\Diamond$. As a corollary, exponential succinctness of the μ-calculus is obtained for a language with $\Diamond$ alone, given that $\diamondsuit$ can be defined succinctly in the μ-calculus. However, this result has two shortcomings with regards to our current goal. First, it does not clarify if the μ-calculus is more succinct than a language with the reflexive modality $\diamondsuit$, so that the results cannot be applied to the logic **Grz** which enjoys a restricted version of the fixed-point theorem. Second, succinctness is obtained via nested fixed point operators, and the lower bound for the fixed-point theorem would require a *single* application of μ. We thus aim for a sharper result for the μ-calculus, for which we extend known techniques and provide new constructions not contingent on the distinction between $\Diamond$ and $\diamondsuit$.

Intuitively, proving that one language $\mathcal{L}_1$ is more succinct than another language $\mathcal{L}_2$ ultimately boils down to proving a sufficiently big lower bound on the size of $\mathcal{L}_2$-formulas expressing some semantic property. If we want to show that $\mathcal{L}_1$ is exponentially more succinct than $\mathcal{L}_2$, we must find an infinite sequence of semantic properties (i.e., classes of models) $\mathbf{P}_1, \mathbf{P}_2, \ldots$ definable in both $\mathcal{L}_1$ and $\mathcal{L}_2$, show that there are $\mathcal{L}_1$-formulas $\varphi_0, \varphi_1, \ldots$ defining $\mathbf{P}_1, \mathbf{P}_2, \ldots$ and prove that, for every n, every $\mathcal{L}_2$-formula ψ_n defining $\mathbf{P}_n$ has size exponential in the size of φ_n. There are various techniques used for achieving such results; here we use formula-size games developed in the setting of Boolean function complexity by [19] and in the setting of first-order logic and some temporal logics by [1]. By now, the formula-size games have been adapted to a host of modal logics (see for example [14], [18], [13], [25]) and used to obtain lower bounds on modal formulas expressing properties of Kripke models.

[1] Normally the μ-calculus requires that x appear only positively in $\varphi(x)$, but over **GL** this condition can be weakened to allow for modalized formulas.

2 Modal logic

In this section, we present the modal μ-calculus and formalize its Kripke semantics. Let us begin by defining the base modal language we will work with. We will consider logics over variants of the language $\mathcal{L}_\Diamond$ given by the following grammar (in Backus-Naur form). Fix a set $\mathbb{P}$ of *propositional variables* (also called *atoms*), and define:

$$\varphi, \psi := \quad \top \quad | \quad \bot \quad | \quad p \quad | \quad \overline{p} \quad | \quad \varphi \vee \psi \quad | \quad \varphi \wedge \psi \quad | \quad \Diamond\varphi \quad | \quad \Box\varphi$$

Here, $p \in \mathbb{P}$ and $\overline{p}$ denotes the negation of p. For the game-theoretic techniques we will use, it is convenient to allow negations only at the atomic level, and thus we include all duals as primitives, but not negation or implication; however, we may use the latter as shorthands, defined via De Morgan's laws. Formulas of the forms $p, \overline{p}$ are *literals.* The *size* of a formula φ is denoted $|\varphi|$ and is defined as follows.

▶ **Definition 1.** *We define a function* $|\cdot| : \mathcal{L}_\Diamond \to \mathbb{N}$ *recursively by*

- $|p| = |\overline{p}| = 1$
- $|\varphi \wedge \psi| = |\varphi \vee \psi| = |\varphi| + |\psi| + 1$
- $|\Diamond\varphi| = |\Box\varphi| = |\varphi| + 1$.

Next we review semantics for modal logic in general, and for **GL** in particular.

▶ **Definition 2.** *A* Kripke frame *is a structure* $\mathcal{A} = (|\mathcal{A}|, R_\mathcal{A})$ *where* $R_\mathcal{A}$ *is a binary relation on* $|\mathcal{A}|$. *If* $\mathcal{A}$ *is a Kripke frame, a* valuation *on* $\mathcal{A}$ *is a function* $V : |\mathcal{A}| \to 2^\mathbb{P}$ *(recall that* $\mathbb{P}$ *is the set of atoms). A frame* $\mathcal{A}$ *equipped with a valuation* V *(often denoted* $V_\mathcal{A}$*) is a* Kripke model.

By abuse on notation we will write $x \in \mathcal{A}$ instead of $x \in |\mathcal{A}|$. The valuation V can be extended recursively to define truth of all formulas of the modal language.

▶ **Definition 3.** *Let* $\mathcal{A} = (\mathcal{A}, R_\mathcal{A})$ *be any Kripke frame and* V *a valuation. We define the truth set*

$$\|\varphi\|_\mathcal{A} =: \{w \in \mathcal{A} : (\mathcal{A}, w) \Vdash \varphi\}$$

by structural induction on φ*:*

$$\begin{array}{lll}
w \in \|p\|_\mathcal{A} & \Leftrightarrow & p \in V(w) \\
w \in \|\overline{p}\|_\mathcal{A} & \Leftrightarrow & p \notin V(w) \\
w \in \|\varphi \wedge \psi\|_\mathcal{A} & \Leftrightarrow & w \in \|\varphi\|_\mathcal{A} \cap \|\psi\|_\mathcal{A} \\
w \in \|\varphi \vee \psi\|_\mathcal{A} & \Leftrightarrow & w \in \|\varphi\|_\mathcal{A} \cup \|\psi\|_\mathcal{A} \\
w \in \|\Diamond\varphi\|_\mathcal{A} & \Leftrightarrow & \exists v (w R_\mathcal{A} v \,\&\, w \in \|\varphi\|_\mathcal{A}) \\
w \in \|\Box\varphi\|_\mathcal{A} & \Leftrightarrow & \forall v (w R_\mathcal{A} v \Rightarrow w \in \|\varphi\|_\mathcal{A})
\end{array}$$

Given a model $\mathcal{A}$ *and formulas* φ, ψ*, we say that* φ *is* equivalent to ψ *on* $\mathcal{A}$ *if* $\|\varphi\|_\mathcal{A} = \|\psi\|_\mathcal{A}$. *If* $\mathbf{A}$ *is a class of models, we say that* φ, ψ *are equivalent over* $\mathbf{A}$ *if they are equivalent on any element of* $\mathbf{A}$. *We may also say that* $\varphi \equiv \psi$ *over* $\mathbf{A}$ *and omit mention of* $\mathbf{A}$ *if it is the class of all Kripke models.*

We will focus our attention mostly on the logics **GL** and **Grz**, which as we will see can be regarded as a fragment. **GL** may be interpreted over structures with a converse well-founded relation and for our purposes we may restrict our attention to models based on trees, presented as strict partial orders.

▶ **Definition 4.** *A* tree *is a pair $(T, \prec)$, where T is a set and $\prec$ is a strict partial order such that, if $\eta \in T$ then $\{\zeta \in T : \zeta \prec \eta\}$ is finite and linearly ordered, and T has a minimum element called its* root*. We will sometimes notationally identify $(T, \prec)$ as T, and write $\preceq$ for the reflexive closure of $\prec$.*

Maximal elements of T are leaves. For $\eta, \zeta \in T$, we say that ζ is a child of η if ζ is the least element ξ (if it exists) such that $\eta \prec \xi$. A path (of length m) on T is a sequence $\vec{\eta} = (\eta_i)_{i \leq m}$ such that η_{i+1} is the child of η_i.

For our purposes, a **GL** *model* is a model $\mathcal{A}$ where $(\mathcal{A}, R_{\mathcal{A}})$ is a finite tree, in which case we write $\sqsubset_{\mathcal{A}}$ instead of $R_{\mathcal{A}}$. As we will be working exclusively with **GL** frames and models, in the sequel we write simply *frame* or *model* instead of **GL** frame or **GL** model.

▶ Remark 5. It should be stressed that working in a more restrictive class of models yields *stronger* results as far as succinctness is concerned: for example, if no small modal formula ψ is equivalent to some μ-calculus expression φ over the class of **GL** models as we have defined them, then certainly no small ψ' is equivalent to φ over the class of *all* Kripke models, as in particular ψ' would still have to be equivalent to φ over the smaller class of **GL** models.

3 Extensions and fixed points

The modal language, as we have presented it, may be naturally extended to include other operations. Even when these operations do not add expressive power to our language, they can yield considerable gains in terms of succinctness, as we will see later in the text. We begin by discussing the reflexive modality.

3.1 The reflexive modality

We may define a modality based on $\sqsubseteq$ rather than $\sqsubset$. This may be defined in $\mathcal{L}_\Diamond$ by letting $\boxdot\varphi$ be a shorthand for $\varphi \wedge \Box\varphi$. Dually, $\diamondsuit\varphi$ is defined as a shorthand for $\varphi \vee \Diamond\varphi$. Let $\mathcal{L}_{\Diamond\diamondsuit}$ be the extension of $\mathcal{L}_\Diamond$ that includes $\diamondsuit, \boxdot$ as primitives. Semantics for $\mathcal{L}_{\Diamond\diamondsuit}$ are defined by setting $\|\varphi\|_{\mathcal{A}} = \|\varphi'\|_{\mathcal{A}}$, where φ' is obtained by replacing instances of $\diamondsuit, \boxdot$ by their definitions; note that in general, φ' tends to be exponentially larger than φ [11]. We extend Definition 1 to $\mathcal{L}_{\Diamond\diamondsuit}$ in the obvious way, by

$$|\diamondsuit\varphi| = |\boxdot\varphi| = |\varphi| + 1.$$

Closely related to **GL** is the logic **Grz** of Noetherian (reflexive) partial orders, but it is easy to see that **Grz** is also the logic of **GL** frames, although based on $\mathcal{L}_\Diamond$ rather than $\mathcal{L}_\Diamond$. By working over the combined language $\mathcal{L}_{\Diamond\diamondsuit}$, our succinctness results apply to both **GL** and **Grz**, as well as many weaker logics.

3.2 Fixed Point Theorems

The celebrated De Jongh-Sambin theorem states that fixed points for modalized formulas are definable in **GL**, where x is *modalized* in φ if it only appears in the scope of $\Diamond$ or $\Box$.[2] For example, the formula $\neg\Box p$ has a fixed point ψ such that $\psi \equiv \neg\Box\psi$ over **GL**; in this case, we can take $\psi = \Diamond\top$. An upper bound on the size of ψ can be obtained by analyzing existing proofs.

[2] In other words, φ is of the form $\psi(\Box(\chi_1(x)), \ldots, \Box(\chi_n(x)))$ with x not occuring in $\psi(p_1, \ldots, p_n)$.

▶ **Theorem 6** (De Jongh ~1975, Sambin [21]). *Given a formula $\varphi(x)$ in which x is modalized, there is a formula ψ such that $\varphi(\psi) \equiv \psi$ over the class of* **GL** *models. The formula ψ is unique up to equivalence, and is of size $2^{O(|\varphi| \log(|\varphi|))}$.*

Proof. We follow the construction of the fixed point formula in [21]. Since x is modalized in φ, we have that $\varphi = \psi\big(\Box\chi_1(x), \dots, \Box\chi_n(x)\big)$. Let σ_i^n be the fixed point of $\psi[\Box\chi_i(x)/\top]$, then the fixed point σ^{n+1} of φ is $\psi\big(\Box\chi_1(\sigma_1^n), \dots, \Box\chi_n(\sigma_n^n)\big)$. Let $m = |\varphi|$ and $k \leq |\psi|$. Then by induction on n, one readily shows that $|\sigma^1| \leq m + k$ and then $|\sigma^n + 1| \leq m + n \cdot |\sigma^{n-1}| \leq m \cdot 2^{(n+1)\log(n+1)}$. ◀

The logic **Grz** also enjoys a fixed point property, but in this case for formulas $\varphi(x)$ where x is *positive,* i.e. with the restriction that $\overline{x}$ may not appear in φ.

Observe how in the case of **GL**, we ask for x to be modalized in $\varphi(x)$ while in **Grz** and in the μ-calculus we want it to be positive. Positivity is typically required in order to avoid pathological cases. For example if $\Box\neg p$ were to have a definable fixed point ψ in **Grz**, then over **Grz** the following would hold $\psi \equiv \Box\neg\psi \equiv \neg\psi \wedge \Box\neg\psi$, a contradiction. Naturally, there is always a fixed point for any $\varphi(x)$ over **GL** as well when x is positive in $\varphi(x)$, however it is not necessarily unique up to equivalence.

▶ **Theorem 7.** *Given a formula $\varphi(x)$ in which x is positive, there is a formula ψ such that $\varphi(\psi) \equiv \psi$ over the class of* **Grz** *models. The size of ψ may be bounded by a $2^{O(|\varphi|^3)}$ function.*

This does not seem to have been stated in this form in the literature, but it is a consequence of Theorem 8 below, which states that the μ-calculus is no more expressive than modal logic over the classes of **GL** or **Grz** models. In order to make this precise, let us review the μ-calculus.

3.3 The μ-calculus

The μ-calculus is obtained from $\mathcal{L}_{\Diamond\Diamondsuit}$ by adding formula constructors $\mu x.\varphi$ and $\nu x.\varphi$, where x is positive in φ. We denote the resulting language by $\mathcal{L}^{\mu}_{\Diamond\Diamondsuit}$. For a model $\mathcal{A}$, a variable x and $X \subseteq |\mathcal{A}|$, let $\mathcal{A}_{[x/X]}$ be a model which is the same as $\mathcal{A}$ except that $V_{\mathcal{A}_{[x/X]}}(x) = X$. Then, $\|\mu x.\varphi\|_{\mathcal{A}}$ is the least fixed point of the map $X \mapsto \|\varphi\|_{\mathcal{A}_{[x/X]}}$; in other words, $\|\mu x.\varphi\|_{\mathcal{A}} = \|\varphi\|_{\mathcal{A}_{[x/\|\mu x.\varphi\|_{\mathcal{A}}]}}$, and every other set with this property contains $\|\mu x.\varphi\|_{\mathcal{A}}$. Syntactically, we obtain $\mu x.\varphi(x) \equiv \varphi(\mu x.\varphi(x))$. Similarly, $\|\nu x.\varphi\|_{\mathcal{A}}$ is the *greatest* fixed point of the map $X \mapsto \|\varphi\|_{\mathcal{A}_{[x/X]}}$. This definition is known to be sound due to the Knaster-Tarski theorem [24], which entails that monotone operators on a powerset always have least and greatest fixed points.

Sub-languages of the μ-calculus are denoted by indicating the modalities allowed, e.g. $\mathcal{L}^{\mu}_{\Diamond}$ allows the modalities $\Diamond, \Box$ but not $\Diamond, \Box$. Formula complexity is extended by setting

$$|\mu x.\varphi| = |\nu x.\varphi| = |\varphi| + 1.$$

Normally the μ-calculus provides a far-reaching extension of modal logic, but surprisingly this is no longer the case over **GL** [2] and **Grz** [9]. The bounds given are obtained from a separate construction by [12].

▶ **Theorem 8.** *Given a formula $\varphi \in \mathcal{L}^{\mu}_{\Diamond\Diamondsuit}$ of the μ-calculus, there are formulas $\varphi_{\mathbf{GL}}$ and $\varphi_{\mathbf{Grz}}$ of $\mathcal{L}_{\Diamond}$ such that $\varphi_{\mathbf{GL}} \equiv \varphi$ over* **GL** *and $\varphi_{\mathbf{Grz}} \equiv \varphi$ over* **Grz**. *The sizes of $\varphi_{\mathbf{GL}}$ and $\varphi_{\mathbf{Grz}}$ are of size $2^{O(|\varphi|^3)}$.*

Proof. In [12], an explicit translation from the μ-calculus into an extension of modal logic is given, with an operator $\blacklozenge^{\infty}(\varphi_0, \varphi_1, \ldots, \varphi_{n-1})$. Over both **GL** and **Grz**, this operator is equivalent to $\diamond \bigwedge \varphi_i$, and thus one obtains a translation of the μ-calculus into $\mathcal{L}_{\diamond}$. By examining the formulas involved, and essentially repeating the same calculation as in [12], we may compute an upper bound of $2^{O(|\varphi|^3)}$. ◀

Note that in general $\varphi_{\mathbf{GL}}$ and $\varphi_{\mathbf{Grz}}$ may be distinct. In view of these results, it may seem that over **GL** and **Grz**, the μ-calculus is merely a cosmetic extension of $\mathcal{L}_{\diamond}$. However, note that the fixed-point formulas provided by Theorems 6 and 7 are quite large. As we will see, this is unavoidable and thus the μ-calculus offers a substantial advantage when succinctness is taken into account. In order to simultaneously provide lower bounds for Theorems 6–8, in our succinctness results, we will work with formulas that are both positive and modalized.

4 Model equivalence games

In this section, we set up the model equivalence games tailored for a language with $\diamond$ and $\diamond$ interpreted over **GL** models. The game is based on sets of rooted **GL** models; that is, **GL** models $\mathcal{A}$ which are *generated* from some $a \in \mathcal{A}$ in the sense that $\mathcal{A} = a{\uparrow_{\mathcal{A}}} := \{b \in \mathcal{A} : a \sqsubseteq b\}$. Henceforth we simply call these *rooted models* and we will usually designate the root by writing the model as the pair $(\mathcal{A}, a)$. The following operations will be useful in describing the game.

▶ **Definition 9.** *Given a set of rooted models* $\mathbf{A}$*, we define:*

- $\Box\mathbf{A} := \{(b{\uparrow_{\mathcal{A}}}, b) : a \sqsubset_{\mathcal{A}} b$ *for some* $(\mathcal{A}, a) \in \mathbf{A}\}$*;*
- $\boxdot\mathbf{A} := \{(b{\uparrow_{\mathcal{A}}}, b) : a \sqsubseteq_{\mathcal{A}} b$ *for some* $(\mathcal{A}, a) \in \mathbf{A}\}$*;*
- *given* $f : \mathbf{A} \to \bigcup_{\mathcal{A} \in \mathbf{A}} \mathcal{A}$ *where* $f(\mathcal{A}) \in \Box\mathcal{A}$*, then* $\diamond_f \mathbf{A} := rng(f)$*;*
- *given* $f : \mathbf{A} \to \bigcup_{\mathcal{A} \in \mathbf{A}} \mathcal{A}$ *where* $f(\mathcal{A}) \in \boxdot\mathcal{A}$*, then* $\diamond_f \mathbf{A} := rng(f)$*.*

We write $\Box\mathcal{A}$ *for* $\Box\{\mathcal{A}\}$ *and* $\boxdot\mathcal{A}$ *for* $\boxdot\{\mathcal{A}\}$ *respectively. We also write* $\mathbf{A} \Vdash \varphi$ *to mean that* φ *holds in the root of every model* $\mathcal{A} \in \mathbf{A}$*.*

▶ **Definition 10.** *Let* $\mathbf{M}$ *be a class of rooted models and* φ *be a formula. The* $(\varphi, \mathbf{M})$ *model equivalence game* ($(\varphi, \mathbf{M})$-MEG*) is played by two players,* Hercules *and the* Hydra*, according to the following rules.*

SETTING UP THE PLAYING FIELD.
The Hydra's only move is in the initiation of the game by choosing two sets of models $\mathbf{A}, \mathbf{B} \subseteq \mathbf{M}$ *such that* $\mathbf{A} \Vdash \varphi$ *and* $\mathbf{B} \Vdash \neg\varphi$*.*

After that, Hercules constructs a finite game-tree T*. Each node* $\eta \in T$ *will be labelled with a pair* $(\mathbf{L}(\eta), \mathbf{R}(\eta))$ *of sets of rooted models, and a symbol that is either a literal or one of* $\{\wedge, \vee, \diamond, \Box, \diamond, \boxdot\}$*. We will usually write* $\mathbf{A}(\eta) \circ \mathbf{B}(\eta)$ *instead of* $(\mathbf{A}, \mathbf{B})$ *for pairs of sets of rooted models.*

At each step of the construction, a leaf η *can be either declared a* head *or a* stub *in accordance to the rules of the game. Once it has been declared a stub, no further moves can be played on it. The root* λ *of the tree is labelled as* $\mathbf{L}(\lambda) \circ \mathbf{R}(\lambda) = \mathbf{A} \circ \mathbf{B}$ *and declared a head.*

Afterwards, the game continues so long as there is at least one head. In each turn, Hercules chooses a head η *labelled by* $\mathbf{L} \circ \mathbf{R}$ *and plays one of the following moves.*

literal-move.
Hercules chooses a literal ι such that $\mathbf{L} \Vdash \iota$ and $\mathbf{R} \nVdash \iota$. The node η is declared a stub and labelled with the symbol ι.

$\vee$-move.
Hercules labels η with the symbol $\vee$ and chooses two sets $\mathbf{L}_1, \mathbf{L}_2 \subseteq \mathbf{L}$ such that $\mathbf{L} = \mathbf{L}_1 \cup \mathbf{L}_2$. Two new heads, labelled by $\mathbf{L}_1 \circ \mathbf{R}$ and $\mathbf{L}_2 \circ \mathbf{R}$, are added to the tree as children of η.

$\wedge$-move.
Analogous to a $\vee$-MOVE*, except that Hercules instead chooses $\mathbf{R}_1, \mathbf{R}_2 \subseteq \mathbf{R}$.*

$\Diamond$-move.
Hercules labels η with the symbol $\Diamond$ and chooses a function f, as in Definition 9, for which $\Diamond_f \mathbf{L}$ exists (if it does not exist i.e. for some $\mathcal{A} \in \mathbf{L}$ we have $\Box\mathcal{A} = \emptyset$, Hercules cannot play this move). We let $\mathbf{L}_1$ be $\Diamond_f \mathbf{L}$ and $\mathbf{R}_1$ to be $\Box\mathbf{R}$.[3] *A new head labelled by $\mathbf{L}_1 \circ \mathbf{R}_1$ is added as a child to η.*

$\Box$-move.
Analogous to a $\Diamond$-MOVE*, except that Hercules instead chooses a function f for which $\Diamond_f \mathbf{R}$ exists and the new head is labelled by $\Box\mathbf{L} \circ \Diamond_f \mathbf{R}$.*

$\diamondsuit$-move and $\boxdot$-move.
Analogous to $\Diamond$- *and* $\Box$-MOVES*, but with $\boxdot$ and $\diamondsuit$ in place of $\Box$ and $\Diamond$ respectively.*

*The $(\varphi, \mathbf{M})$-*MEG *game concludes when there are no heads. If the game-tree is finite (in size) and it has no heads, we call it* closed *and Hercules has won. We say that* Hercules has a winning strategy in n moves *in the $(\varphi, \mathbf{M})$-*MEG *if no matter how the Hydra sets up the playing field, the resulting game tree has at most n nodes and is closed.*

If there is an $\mathcal{L}_{\Diamond\diamondsuit}$ formula ψ equivalent to φ on $\mathbf{M}$, Hercules can read a winning strategy off of ψ for the $(\varphi, \mathbf{M})$-MEG. Conversely, if Hercules has a winning strategy then such a ψ can be read off of the game tree when Hydra plays optimally, i.e. always choosing as many rooted models as allowed. We thus obtain the following.

▶ **Theorem 11.** *Hercules has a winning strategy in n moves in the $(\varphi, \mathbf{M})$-*MEG *iff there is a $\mathcal{L}_{\Diamond\diamondsuit}$ formula ψ equivalent to φ on $\mathbf{M}$ such that $|\psi| \leq n$. (See e.g. [18])*

It moreover should be clear that Hercules cannot win if there are isomorphic models on the left and right, since there will be no formula distinguishing them.

▶ **Proposition 12.** *No closed game tree contains a node η such that there are $\mathcal{A} \in \mathbf{L}(\eta)$, $\mathcal{B} \in \mathbf{R}(\eta)$ that are isomorphic. (See e.g. [18])*

5 The playing field

We will use the model equivalence games to show that the μ-calculus with a *single* application of the least fixed point operator is exponentially more succinct than modal logic over the class of **GL** frames, even when equipped with *both* $\Diamond$ and $\diamondsuit$. Our proof will be based on an infinite sequence of formulas that convey the existence of a certain binary tree: the root is labelled by q^0_n, with two children labelled by q^0_{n-1} and q^1_{n-1}, respectively, and so on until we reach leaves labelled by q^0_0 and q^1_0. To maintain control over the tree structure, we use auxiliary variables p^k which "remember" the parent's label.

[3] In particular, if $\Box\mathcal{B} = \emptyset$ for some $\mathcal{B} \in \mathbf{R}$, then nothing is added to $\mathbf{R}_1$ for the rooted model $\mathcal{B}$.

▶ **Definition 13.** *For every $n \geq 0$, let the open formulas $\varphi_n^*(x)$ be defined as follows. First, for $n \in \mathbb{N}$ set $\theta_n(x)$ to be the formula*

$$\bigwedge_{j \leq 2} \left(q_{n+1}^j \to \bigwedge_{k \leq 2} \Diamond(q_n^k \wedge p^j \wedge x \wedge \neg q_{n+1}^j)\right)$$

and define $\varphi_n^*(x) = \bigwedge_{i \leq n} \theta_i(x)$. Then, for $n \geq 0$, let φ_n and $\overline{\varphi_n}$ be defined as:

$$\varphi_n := q_{n+1}^0 \wedge \mu x.\varphi_n^*(x);$$
$$\overline{\varphi_n} := q_{n+1}^1 \wedge \mu x.\varphi_n^*(x).$$

By the fixed-point theorem for **GL**, we know there are $\mathcal{L}_{\Diamond\Diamond}$ formulae $\psi_n \equiv \varphi_n$ over **GL** of size at most $2^{O(n \log(n))}$.[4] Due to the occurrence of $\neg q_{n+1}^j$, the formulae θ_n are equivalent to the formulae θ'_n obtained by substituting $\Diamond$ for $\Diamond$. As such, any lower bound results for the size of $\mathcal{L}_{\Diamond\Diamond}$ formulae equivalent to the formulae φ_n in **GL** will also produce succinctness results for $\mathcal{L}^{\mu}_{\Diamond}$, hence also for **Grz**.

Observe that the formulas $\varphi_n^*(x)$ are all positive over x and modalized. We therefore know by the fixed-point theorem for **GL** that their fixed-point is unique and since they are positive for x, their fixed point will also be equivalent to their greatest and least fixed point.

As promised, the above formulas will define a tree embedding property as this is a sufficient condition a rooted model should satisfy in order for some φ_n to hold in its root. If we are to be more precise, consider the following model $\mathcal{T}_n = \langle \mathcal{T}_n, \prec, V_{\mathcal{T}_n} \rangle$, where $\mathcal{T}_n$ is the set of binary sequences of length $\leq n+1$, rooted at the empty sequence $\langle\rangle$ and:

1. $V_{\mathcal{T}_n}(q_{n+1}^0) := \{\langle\rangle\}$;
2. $V_{\mathcal{T}_n}(q_i^j) = \{s \in \mathcal{T}_n : |s| := n+1-i \wedge s(|s|-1) = j\}$ for $i \leq n$;
3. $V_{\mathcal{T}_n}(p^j) = \{s \in \mathcal{T}_n : \exists k\ s := r \frown \langle k \rangle \wedge r \in V_{\mathcal{T}_n}(q_{n-|r|}^j)\}$;

where $s \frown r$ denotes the concatenation of the sequences s and r. Intuitively, q_i^0 is true on paths of length $n+1-i$ that go "left" on the last step, and q_i^1 holds instead when they go "right" on the last step. The truth value of p^j on the world mimics that of q_{i+1}^j on its parent. Note that the lower index of the variables goes "backward" from the root to the leaves.

A *model embedding* is a function $f : \mathcal{M} \to \mathcal{N}$ such that:

1. For every $a, b \in \mathcal{M}$, if $a \sqsubset_M b$ then $f(a) \sqsubset_N f(b)$;
2. For every $p \in \mathbb{P}(\mathcal{M})$ and $a \in \mathcal{M}$, $a \in V_{\mathcal{M}}(p)$ iff $f(a) \in V_{\mathcal{N}}(p)$;

where $\mathbb{P}(\mathcal{M})$ denotes the set of propositional variables occurring in the valuation of some world in $\mathcal{M}$; i.e. $\mathbb{P}(\mathcal{M}) := \{p \in \mathbb{P} : \exists v \in \mathcal{M} v \Vdash p\}$.

By construction, for every n, $\mathcal{T}_n \Vdash \varphi_n$. While we will mostly focus on φ_n in our inductive arguments, down the line we will need models satisfying $\overline{\varphi_n}$ at the root. Models of φ_n and $\overline{\varphi_n}$ vary on whether q_n^0 or q_n^1 holds at the root, i.e. in φ_n the tree starts on the "left" and in $\overline{\varphi_n}$ on the "right". Hence we use the following notational convention:

Given a rooted model $(\mathcal{A}, a)$, let $N^+(a)$ be the set of children of a, i.e. the set of direct successors of a. Define $\overline{\mathcal{A}} := (\mathcal{A}, \sqsubset_{\mathcal{A}}, V_{\overline{\mathcal{A}}})$ where for $\overline{j} = 1 - j$:

- The valuations of q_i^j and $q_i^{\overline{j}}$ are swapped at the root;
- The valuations of p^j and $p^{\overline{j}}$ are swapped at the children of a.

All unmentioned propositional variables will be evaluated the same as before.

Observe that in the case of the tree models we have defined, the symmetric counterpart $\overline{\mathcal{T}_n}$ satisfies $\overline{\varphi_n}$ at its root.

[4] Since the size of φ_n is linear in n, we substitute n for $|\varphi_n|$ in the above expression.

▶ **Lemma 14.** *Let $n \in \mathbb{N}$ and $(\mathcal{A}, a)$ be a rooted model such that for all $b \in \mathcal{A}$, there is at most one pair $\langle i, j\rangle$ such that $b \in V_{\mathcal{A}}(q_i^j)$, and at most one j such that $b \in V_{\mathcal{A}}(p^j)$.*

1. *The model $(\mathcal{A}, a)$ satisfies φ_n iff there is a model embedding $f : \mathcal{T}_n \to \mathcal{A}$ such that $f(\langle\rangle) = a$.*
2. *The model $(\mathcal{A}, a)$ satisfies $\overline{\varphi_n}$ iff there is a model embedding $f : \overline{\mathcal{T}_n} \to \mathcal{A}$ such that $f(\langle\rangle) = a$.*

Proof. We prove both items simultaneously by induction on n.

Left-to-Right. For $n = 0$ assume first that $(\mathcal{A}, a)$ is a model satisfying the assumptions of the Lemma and additionally $\mathcal{A}, a \Vdash \varphi_0$. Thus $\mathcal{A}, a \Vdash q_1^0$ and since $\mathcal{A}, a \Vdash \mu x.\varphi_0^*(x)$, then $\mathcal{A}, a \Vdash \varphi_0^*(\mu x.\varphi_0^*(x))$ and hence for $k \in 2$ there are $b_k \sqsupset a$ such that $\mathcal{A}, b_k \Vdash q_0^k \wedge p^0$. This naturally gives us the embedding from $\mathcal{T}_1$ into $\mathcal{A}$.
Assume that the induction step holds for n and we will show the equivalence for $n+1$. Let $\mathcal{A}, a \Vdash \varphi_{n+1}$, then $\mathcal{A}, a \Vdash q_{n+1}^0$ and $A, a \Vdash \varphi_{n+1}^*(\mu x.\varphi_{n+1}^*(x))$, so there are $b_k \sqsupset a$ for $k \in 2$ such that $\mathcal{A}, b_k \Vdash q_n^k \wedge p^j \wedge \mu x.\varphi_{n+1}^*(x)$ while also $\mathcal{A}, b_k \nVdash q_{n+1}^0$. Since $\mathcal{A}, b_k \nVdash q_{n+1}^0$, it follows that $\mathcal{A}, b_k \Vdash \varphi_{n+1}^*(x)$ iff $\mathcal{A}, b_k \Vdash \varphi_n^*(x)$. Therefore:

$$\begin{aligned}
&\mathcal{A}, b_k \Vdash \mu x.\varphi_{n+1}^*(x) \Leftrightarrow \\
&\mathcal{A}, b_k \Vdash \varphi_{n+1}^*(\mu x.\varphi_{n+1}^*(x)) \Leftrightarrow \\
&\mathcal{A}, b_k \Vdash \varphi_n^*(\mu x.\varphi_{n+1}^*(x)) \Rightarrow \\
&\mathcal{A}, b_k \Vdash \varphi_n^*(\mu x.\varphi_n^*(x)). \qquad (1)
\end{aligned}$$

The last implication holds by monotonicity. Thus, we can use the induction hypothesis for $b_0{\uparrow_{\mathcal{A}}}$ and $b_1{\uparrow_{\mathcal{A}}}$ to obtain the desired embedding. The case for $\overline{\varphi_n}$ is symmetrical.

Right-to-Left. Let $f : \mathcal{T}_0 \to \mathcal{A}$ be a model embedding; we just need to show that $\mathcal{A}, a \Vdash \varphi_0^*(\mu x.\varphi_0^*(x))$. By our assumption there will be elements $b_k \sqsupset a$ for $k \in 2$ such that $\mathcal{A}, b_k \Vdash q_0^k \wedge p^0$ and $\mathcal{A}, b_k \nVdash q_1^j$ for any $j \in 2$. Therefore $\mathcal{A}, b_k \Vdash \varphi_0^*(\bot)$ and since x occurs positively in φ_0^*, it follows that $\mathcal{A}, b_k \Vdash \varphi_0^*(\mu x.\varphi_0^*(x))$. Hence $\mathcal{A}, a \Vdash \varphi_0^*(\varphi_0^*(\mu x.\varphi_0^*(x)))$, and so $\mathcal{A}, a \Vdash \mu x.\varphi_0^*(x)$.
Assume now that the induction step holds for n and let $f : \mathcal{T}_{n+1} \to \mathcal{A}$ be a model embedding. Thus $\mathcal{A}, a \Vdash q_{n+1}^0$ and there are $b_k \sqsupset a$ for $k \leq 2$ and model embeddings $g_0 : \mathcal{T}_n \to b_0{\uparrow_{\mathcal{A}}}$, $g_1 : \overline{\mathcal{T}_n} \to b_1{\uparrow_{\mathcal{A}}}$. By the induction hypothesis, $\mathcal{A}, b_0 \Vdash \varphi_n$ and $\mathcal{A}, b_1 \Vdash \overline{\varphi_n}$. Item 2 is proved similarly. ◀

Our models will be designed so that they have a *critical branch* on which Hercules will be forced to play. This critical branch is described using a "successor" function which goes up the tree along said branch. This and other technical notions needed to describe Hercules' strategy are given by the following definition.

▶ **Definition 15.** *Given a rooted model $(\mathcal{A}, a)$ and a propositional variable p, we will denote by $\mathcal{A}(p)$ the model $\langle \mathcal{A}, \sqsubset_A, V'\rangle$ where V' is the same as $V_{\mathcal{A}}$ except that p will also hold in the root of the model. In the interest of distinguishing the root of $\mathcal{A}(p)$ from $\mathcal{A}(q)$ we will write them as $a(p)$ and $a(q)$, respectively (even though it's technically the same element).*

A model with successors *is a model $\mathcal{A}$ equipped with a partial function $S_{\mathcal{A}} \colon \mathcal{A} \to \mathcal{A}$ such that $S_{\mathcal{A}}(a)$ is always a child of a.*

If $\mathcal{A}$ is a rooted model with successors, the critical branch of $\mathcal{A}$ *is the maximal path $\vec{w} = (w_i)_{i \leq m}$ such that w_0 is the root of $\mathcal{A}$ and $w_{i+1} = S_{\mathcal{A}}(w_i)$ for all $i < m$; we say that m is the* critical height *of $\mathcal{A}$.*

We denote by $S[\mathcal{A}]$ the generated submodel of $S_{\mathcal{A}}(w_0)$ with $S_{\mathcal{A}}(w_0)$ as its root and its induced successor function being $S_{S[\mathcal{A}]}(w_0)$. For a natural number r, we define the rth iteration of $S_{\mathcal{A}}$ by induction so that $S^{(0)}[\mathcal{A}] := \mathcal{A}$, $S^{(0)}_{\mathcal{A}}(w) := w$ and on the inductive step $S^{(r+1)}[\mathcal{A}] := S[S^{(r)}[\mathcal{A}]]$, $S^{(r+1)}_{\mathcal{A}}(w) := S_{\mathcal{A}}(S^{(r)}_{\mathcal{A}}(w))$.

The partial function $S_{\mathcal{A}}$ will not be used in the semantics, but it will help us to describe Hercules' strategy. We will begin by defining sets of rooted models $\mathbf{A}^n$ and $\mathbf{B}^n$ recursively on n containing 2^{n+1} models each. Of those, the former 2^n will be used to prove our succinctness lower bound while the latter 2^n are auxiliary[5] and used solely in our recursive construction. The following definition is illustrated in Figures 1 and 2.

▶ **Definition 16.** *First, for $n = 0$ and for $i < 2$ we define $\mathcal{A}^0_i$ with domains the sequences s of length at most 1 on the natural numbers $\{0, 1, 2\}$. Then $\sqsubset_{\mathcal{A}^0_i}$ is the prefix relation and valuations $V_{\mathcal{A}^0_i}(p^i) = \{\langle k\rangle : k \leq 2\}$, $V_{\mathcal{A}^0_i}(q^i_1) = \{\langle\rangle\}$ and $V_{\mathcal{A}^0_i}(q^j_0) = \{\langle j\rangle\}$ for $j < 2$. Set $S_{\mathcal{A}^0_i}(\langle\rangle) = \langle 1 - i\rangle$ and $S_{\mathcal{A}^0_i}$ is undefined otherwise. The $\mathcal{B}^0_i$ have as domain the sequences s of length at most 1 in the natural numbers ≤ 1. Their relations are the prefix relations and the valuations are the following: $V_{\mathcal{B}^0_i}(p^i) = \{\langle k\rangle : k \leq 1\}$, $V_{\mathcal{B}^0_i}(q^i_1) = \{\langle\rangle\}$ and $V_{\mathcal{B}^0_i}(q^i_0) = \{\langle 0\rangle\}$. Set $S_{\mathcal{B}^0_i} = \{(\langle\rangle, \langle 1\rangle)\}$.*
For $2 \leq i < 4$, we let $\mathcal{A}^0_i := \overline{\mathcal{A}^0_{i-2}}$ and $\mathcal{B}^0_i := \overline{\mathcal{B}^0_{i-2}}$. Their successor functions remain the same. Now, given n, we will define the models $\mathcal{A}^{n+1}_i$ and $\mathcal{B}^{n+1}_i$ with a case distinction in i. In this paper, given models $\mathcal{A}$ and $\mathcal{B}$, we let $\mathcal{A} \amalg \mathcal{B}$ be the model with domain the disjoint union of $\mathcal{A}$ and $\mathcal{B}$,[6] accessibility relation $\sqsubset_{\mathcal{A}\amalg\mathcal{B}}$ being the disjoint union of the accessibility relations $\sqsubset_{\mathcal{A}}$ and $\sqsubset_{\mathcal{B}}$, and similarly the valuation. This is the set of all the elements in $\mathcal{A}$ and $\mathcal{B}$ with each element labelled by the set to which it belongs.

Case $i < 2^{n+1}$. *First, set $\mathcal{X}$ to be*

$$\Big(\coprod_{k=0}^{2^{n+1}-1} \mathcal{A}^n_k(p^1)\Big) \amalg \mathcal{A}^n_i(p^0) \amalg \overline{\mathcal{A}^n_i}(p^0) \amalg \mathcal{B}^n_i(p^0),$$

and construct $\mathcal{A}^{n+1}_i$ by adding a (fresh) irreflexive root a^{n+1}_i which is below all elements of $\mathcal{X}$ and satisfies q^0_{n+2}. We set

$$S_{\mathcal{A}^{n+1}_i} := S_{\mathcal{A}^n_i} \cup \{(a^{n+1}_i, a^n_i(p^0))\}.$$

The models $\mathcal{B}^{n+1}_i$ are defined similarly by setting

$$\mathcal{Y} = \Big(\coprod_{k=0}^{2^{n+1}-1} \mathcal{A}^n_k(p^1)\Big) \amalg \overline{\mathcal{A}^n_i}(p^0) \amalg \mathcal{B}^n_i(p^0),$$

and, as before, adding an irreflexive root b^{n+1}_i that satisfies q^0_{n+2}. Set

$$S_{\mathcal{B}^{n+1}_i} := S_{\mathcal{B}^n_i} \cup \{(b^{n+1}_i, b^n_i(p^0))\}.$$

Case $2^{n+1} \leq i < 2^{n+2}$. *Here we construct our auxiliary models where $\mathcal{A}^{n+1}_i = \overline{A^{n+1}_{i-2^{n+1}}}$ and $\mathcal{B}^{n+1}_i = \overline{B^{n+1}_{i-2^{n+1}}}$.*

We remark that the auxiliary models were only used to help us inductively construct the $\mathcal{A}^n_i$ and $\mathcal{B}^n_i$ models. Letting $\mathbf{A}^n = \{\mathcal{A}^n_i : i < 2^n\}$, $\mathbf{B}^n = \{\mathcal{B}^n_i : i < 2^n\}$ and $\mathbf{M}_n = \mathbf{A}^n \cup \mathbf{B}^n$, we will study the $(\varphi_n, \mathbf{M}_n)$-MEG where φ_n are the formulae in Definition 13.

[5] They are the "right" versions of the former models.
[6] Hence elements in the intersection $\mathcal{A} \cap \mathcal{B}$ will appear twice; once labelled by $\mathcal{A}$ and once labelled by $\mathcal{B}$.

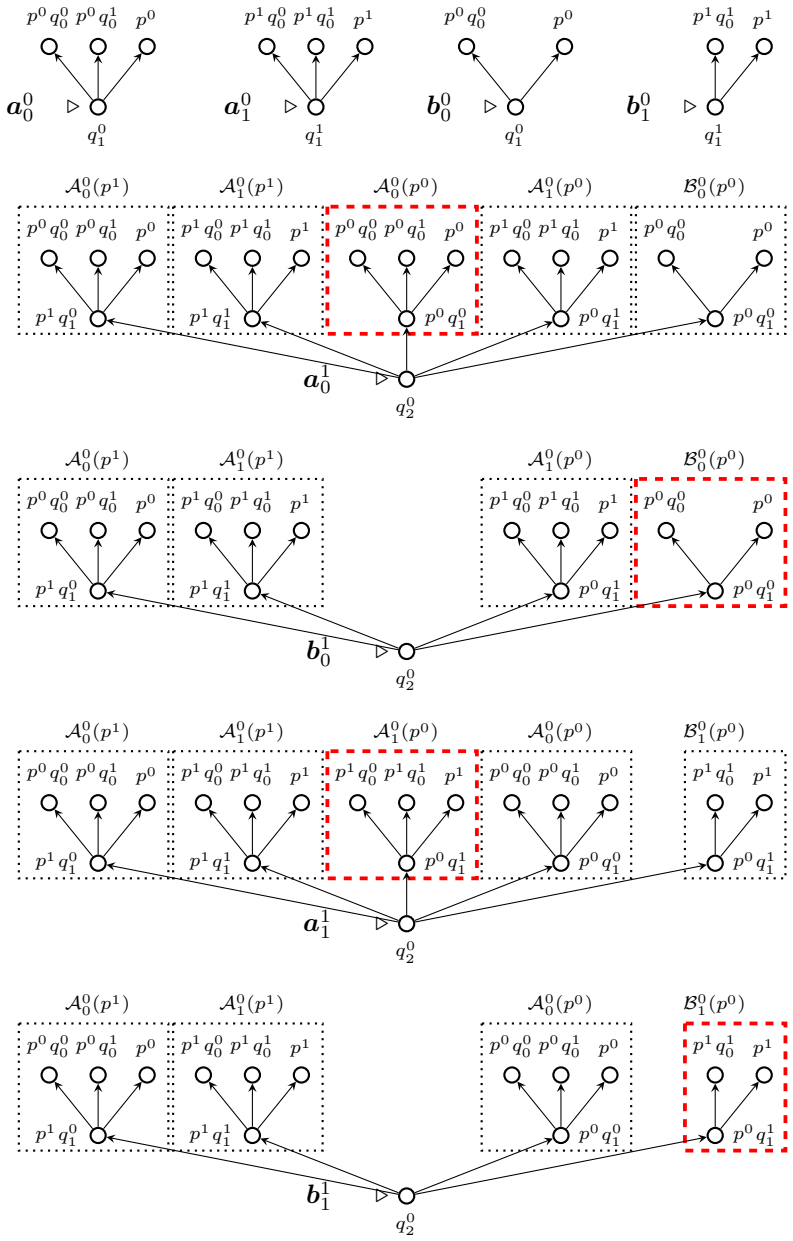


Figure 1 The figure illustrates models in $\mathbf{M}_0$ and $\mathbf{M}_1$. At the very top, we see the rooted models of $\mathbf{A}^0, \mathbf{B}^0$ as well as their auxiliary models. These are then used in the construction of the models of $\mathbf{A}^1, \mathbf{B}^1$ that we can see in the rest of the graphic.

Each copy of the smaller models being used in the construction is indicated by a box and a label. It is easy to see that $\mathcal{A}^0_0$ embeds $\mathcal{T}_0$ (with the image being all but the rightmost leaf) and $\mathcal{A}^0_1$ embeds $\overline{\mathcal{T}}_0$, but $\mathcal{B}^0_0$ and $\mathcal{B}^0_1$ do not, hence $\mathcal{A}^0_0$ satisfies φ_0 while $\mathcal{B}^0_0$ does not, and similarly $\mathcal{A}^0_1$ satisfies $\overline{\varphi}_0$ while $\mathcal{B}^0_1$ does not. A similar analysis shows that e.g. $\mathcal{A}^1_0$ satisfies φ_1 but $\mathcal{B}^1_0$ does not.

The successor function will point from the root towards the sub-model in the red, dashed boxes; for example, $S[\mathcal{A}^1_0] = \mathcal{A}^0_0(p^0)$. The models $S[\mathcal{A}^1_0]$ and $S[\mathcal{B}^1_0]$ are similar enough that Hercules is not able to tell them apart before reaching their topmost worlds. The remaining branches are there to make this task as difficult as possible.

6 The lower bound

In this section, we show that if Hydra sets up the playing field with $\mathbf{M}^n$, then Hercules cannot win the game in fewer than 2^n moves. This is our main technical result and will require several preparatory lemmas.

As mentioned, each model has a critical branch. More specifically, each $\mathcal{A}_i^{n+1}$ is very similar to the respective $\mathcal{B}_i^{n+1}$, and can only be distinguished by Hercules if he plays along the critical branch. The number i can be seen as coding a binary string simply by writing $i = e_{n+1}2^{n+1} + \ldots + e_0 2^0$ in binary. Then each e_{n+1-r} indicates whether the critical branch goes left or right at step i; note that this includes step 0, corresponding to the label of the root, i.e. the critical branch of $\mathcal{A}_i^{n+1}$ goes left first while that of $\overline{\mathcal{A}_i^{n+1}}$ goes right.

▶ **Lemma 17.** *For all n, $i < 2^{n+2}$ and $r < n+2$, the following hold:*
1. *$S^{(r)}[\mathcal{A}_i^{n+1}]$ is isomorphic to $\mathcal{A}_k^{n+1-r}(p^e)$;*
2. *$S^{(r)}[\mathcal{B}_i^{n+1}]$ is isomorphic to $\mathcal{B}_i^{n+1-r}(p^e)$;*

where $k \equiv i \mod 2^{n+2-r}$ with $k < 2^{n+2-r}$ and e is the digit $e_{n+2-r+1}$ in the binary expansion of i.

Proof. We will only prove Item 1 as the case for the $\mathcal{B}_i^{n+1}$ model is identical. The case for $r = 1$ is immediate from the definitions of the models and the S function. So assume that the lemma holds for $r < n+1$. By the induction hypothesis, $S^{(r+1)}[\mathcal{A}_i^{n+1}]$ is isomorphic to $S[\mathcal{A}_k^{n+1-r}(p^{e_{n+1-r}})]$, which by the induction hypothesis for $r = 1$, this is in turn isomorphic to $\mathcal{A}_k^{n+1-r}(p^{e_{n-r}})$. ◀

By construction, $\mathcal{T}_n$ embeds into $\mathcal{A}_i^n$ but not into $\mathcal{B}_i^n$, yielding the following.

▶ **Lemma 18.** *For all n and all $i < 2^n$, $\mathcal{A}_i^n \Vdash \varphi_n$ and $\mathcal{B}_i^n \Vdash \neg\varphi_n$.*

Proof. For this proof, we will extend our definition of $\overline{\cdot}$ into embeddings as follows: Given an embedding $f : \mathcal{A} \to \mathcal{B}$, we define $\overline{f} : \overline{\mathcal{A}} \to \overline{\mathcal{B}}$ to be such that $\overline{f}(a) = f(a)$ for all $a \in A$. Notice that $\overline{f}$ will still be an embedding if it preserves the root r and maps children of the root of A into children of the root of B (i.e. $f[N^+(r)] \subseteq N^+(f(r))$).

We will make use of Lemma 14. First, we show $\mathcal{A}_i^n \Vdash \varphi_n$ by proving the existence of embeddings $f_i^n : \mathcal{T}_n \to \mathcal{A}_i^n$ for $i < 2^n$ by induction on n. Notice that since $\mathcal{A}_i^n$ and $\mathcal{T}_n$ have the same depth, we will also obtain $f_i^n[N^+(\langle\rangle)] \subseteq N^+(f_i^n(\langle\rangle))$; thus, $\overline{f_i^n}$ will also be an embedding from $\overline{\mathcal{T}_n}$ to $\mathcal{A}_{i+2^n}^n = \overline{\mathcal{A}_i^n}$.
For $n = 0$, one needs only look at the definition of $\mathcal{A}_0^0$. Now assume that the statement holds for n, let $i < 2^{n+1}$ be arbitrary, $j \equiv i \mod 2^n$ and $f_j^n : \mathcal{T}_n \to \mathcal{A}_j^n$ be an embedding given from our induction hypothesis. We can then define $f_i^{n+1} : \mathcal{T}_{n+1} \to \mathcal{A}_i^{n+1}$ by mapping:
- $\langle\rangle$ to a_i^{n+1};
- $\langle 0\rangle \frown s$ to the point corresponding to $f_j^n(s)$ on the copy of $\mathcal{A}_j^n(p^0)$ in $\mathcal{A}_i^{n+1}$;
- $\langle 1\rangle \frown s$ to the point corresponding to $\overline{f_j^n}(s)$ on the copy of $\overline{\mathcal{A}_j^n}(p^0)$ in $\mathcal{A}_i^{n+1}$.

$\mathcal{A}_i^{n+1}$ satisfies the base conditions of Lemma 14 and thus our induction step holds for all $i < 2^{n+1}$ and so $\mathcal{A}_i^n \Vdash \varphi_n$.

We now show $\mathcal{B}_i^n \Vdash \neg\varphi_n$ by proving that there are no embeddings as in Lemma 14 $f : \mathcal{T}_n \to \mathcal{B}_i^n$ for all $i < 2^n$ by induction on n.
For $n = 0$, this is clear. Assume that the induction hypothesis holds for n, and suppose, for a contradiction, that there is an embedding $f : \mathcal{T}_n \to \mathcal{B}_i^{n+1}$ with $f(\langle\rangle) = b_i^{n+1}$ for some $i < 2^n$. As f is an embedding, $\{q_{n+1}^0, p^0\} \subseteq V_{\mathcal{B}_i^{n+1}}^{-1}\big(f(\langle 0\rangle)\big)$ and the only world that satisfies this condition in $\mathcal{B}_i^{n+1}$ is the root b_i^n of the $\mathcal{B}_i^n(p^0)$ part of the model. This implies the existence of a root-preserving embedding $f' : \mathcal{T}_n \to \mathcal{B}_i^n$ which contradicts our induction hypothesis. The case for $2^n \leq i < 2^{n+1}$ is done in a similar way. ◀

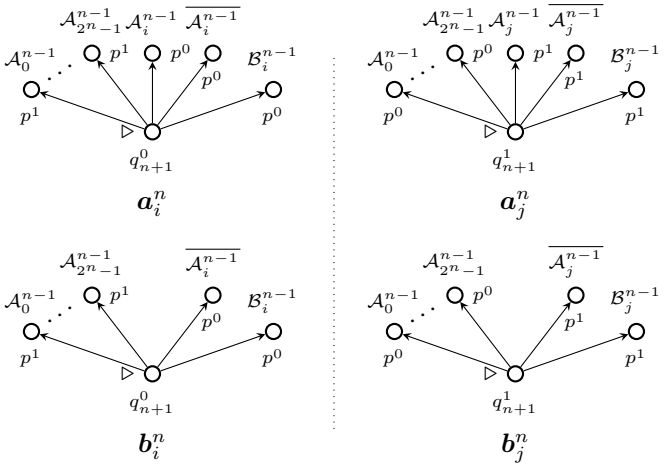


Figure 2 The inductive structure of the models for $n > 0$ with the models of $\mathcal{A}^n$ and $\mathcal{B}^n$ to the left and the auxiliary models to the right ($0 \leq i < 2^n \leq j < 2^{n+1}$).

Hence, Hydra can set up the playing field by placing the models $\mathcal{A}^n_i$ on the left and the models $\mathcal{B}^n_i$ on the right. In this case, Hercules requires exponentially many moves to win the game.

▶ **Definition 19.** *Suppose that $\mathcal{M}$, $\mathcal{N}$ are two finite rooted models with successors. We say that $r \in \mathbb{N}$* distinguishes *$\mathcal{M}$ and $\mathcal{N}$ if $S^{(r)}[\mathcal{M}]$ and $S^{(r)}[\mathcal{N}]$ differ on the truth of a propositional variable at their roots, but whenever $i < r$, then $S^{(i)}[\mathcal{M}]$ and $S^{(i)}[\mathcal{N}]$ agree on the truth of all propositional variables at their respective roots. We call r the* distinguishing value *of $\mathcal{M}$ and $\mathcal{N}$.*

Note that the distinguishing value of two models $\mathcal{M}, \mathcal{N}$ need not be defined, but when it is, it is unique. Moreover, the distinguishing values of the models we have constructed always exist.

▶ **Lemma 20.** *Fix $n \geq 1$ and $0 \leq i < j < 2^{n+1}$. Then, $\mathcal{A}^n_i$ and $\mathcal{A}^n_j$ are distinguished at some $r < n$, satisfying the following properties:*

(a) *If $i < 2^n$ and $2^n \leq j$, then $\mathcal{A}^n_i$ and $\mathcal{A}^n_j$ have distinguishing value 0.*

(b) *If $\mathcal{A}^n_i$ and $\mathcal{A}^n_j$ have distinguishing value r, then $\mathcal{A}^{n+1}_i$, $\mathcal{A}^{n+1}_j$ have distinguishing value $r+1$. The same holds for $\mathcal{A}^{n+1}_{2^{n+1}+i}$, $\mathcal{A}^{n+1}_{2^{n+1}+j}$.*

Proof. Item a is immediate since the roots of $\mathcal{A}^n_i$ and $\mathcal{A}^n_j$ are evaluated differently. For Item b, observe that $i, j < 2^{n+1}$ implies that $\mathcal{A}^{n+1}_i$ and $\mathcal{A}^{n+1}_j$ have roots with the same valuation. Thus, they are distinguished at $r+1$. Since $\mathcal{A}^{n+1}_{2^{n+1}+i} = \overline{\mathcal{A}^{n+1}_i}$ and $\mathcal{A}^{n+1}_{2^{n+1}+j} = \overline{\mathcal{A}^{n+1}_j}$, these are also distinguished at $r+1$. ◀

▶ **Lemma 21.** *Fix n and $i < 2^{n+1}$. Then, $\mathcal{A}^n_i$ and $\mathcal{B}^n_i$ are distinguished at $n+1$.*

Proof. By an easy induction on n. ◀

By *twins of height k* we mean a pair of the form $(S^{(k)}[\mathcal{A}^n_i], S^{(k)}[\mathcal{B}^n_i])$, where $i < 2^n$ and $k \leq n+1$. If $\mathbf{L}$ is a set of rooted models from $\mathbf{A}^n$ and $\mathbf{R}$ a set of rooted models from $\mathbf{B}^n$, we say that there are twins of height k in $\mathbf{L} \circ \mathbf{R}$ if there are twins $(S^{(k)}[\mathcal{A}^n_i], S^{(k)}[\mathcal{B}^n_i])$ such that $S^{(k)}[\mathcal{A}^n_i] \in \mathbf{L}$ and $S^{(k)}[\mathcal{B}^n_i] \in \mathbf{R}$.

We will study how pairs of the form $(S^{(k)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n])$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ affect the viability of the various modal moves for Hercules. This will place restrictions on the relationship between k and r. For example, we see that Item b below states that if $k < r$ then Hercules couldn't play any $\boxdot$ moves as Hydra can get from $S^{(k)}[\mathcal{A}_i^n]$ into a model isomorphic to any choice of Hercules in $\boxdot S^{(r)}[\mathcal{B}_i^n]$. Clearly, that also excludes any $\Box$ moves for Hercules, as two isomorphic models are of the same height and hence any isomorphic model the Hydra would produce for the corresponding $\boxdot$ move, would also show up in a $\Box$ move. In Lemma 22, all of the restrictions applying to a reflexive modality will not only just apply to the irreflexive modality, but also, they will apply even if we substitute $\leq$ for $<$.

At this point, let us fix $n \geq 0$ and assume that Hydra labels the root with $\mathbf{A}^n \circ \mathbf{B}^n$.

▶ **Lemma 22.** *For any node η in a closed game tree $(T, \prec)$ for the $(\varphi_n, \mathbf{GL})$-*MEG

(a) *If there are twins $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ with $k < n+1$ then no literal move was played in the node η.*

(b) *If there are $S^{(k)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n]$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ and $k < r$ then $\boxdot$ was not played in the node η. If $k \leq r$ then no $\Box$ move was played in η either.*

(c) *If there are twins $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ and a $\boxdot$ move was played in the node η, then $S^{(k)}[\mathcal{B}_i^n]$ was chosen.*

(d) *If there are twins $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ and a $\diamondsuit$ move was played in the node η, then either $S^{(k)}[\mathcal{A}_i^n]$ or $S^{(k+1)}[\mathcal{A}_i^n]$ was chosen. Hence, if a $\diamondsuit$ move was played in η, then $S^{(k+1)}[\mathcal{A}_i^n]$ was chosen.*

(e) *If there are two distinct twins $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ and $S^{(r)}[\mathcal{A}_j^n], S^{(r)}[\mathcal{B}_j^n]$ in $\mathbf{L}(\eta) \circ \mathbf{R}(\eta)$, then*

- **(i)** *if $r+1 < k$, then no $\diamondsuit$ or $\boxdot$ move was played in η. Similarly, if $r < k$, then no $\diamondsuit$ or $\Box$ move was played in η.*
- **(ii)** *If $\mathcal{A}_i^n, \mathcal{B}_j^n$ are distinguished at r and $r = k$ then no $\diamondsuit$ move was played in η. If instead $k = r+1$, then no $\diamondsuit$-move was played in η either.*

Proof. Item a is immediate by Lemma 21 as no literal move can be played if there are models $(\mathcal{A}, \mathcal{B}) \in \mathbf{L}(\eta) \circ \mathbf{R}(\eta)$ with distinguishing value $r > 0$.

For Item b, if $r = n+1$, then the statement is clear by the definition of the $\mathcal{B}_j^0$. Thus, assume $r < n+1$. Then observe that $S^{(r)}[\mathcal{B}_i^n] \subseteq \boxdot S^{(k)}[\mathcal{A}_i^n]$ and hence $\boxdot S^{(r)}[\mathcal{B}_i^n] \subseteq \boxdot S^{(k)}[\mathcal{A}_i^n]$. Hence, no matter where $S^{(r)}[\mathcal{B}_i^n]$ is mapped in $\boxdot S^{(r)}[\mathcal{B}_i^n]$ by Hercules, Hydra will include an isomorphic model in its response. Observe that if $k = r$ then we still have that $\Box S^{(k)}[\mathcal{B}_i^n] \subseteq \Box S^{(k)}[\mathcal{A}_i^n]$.

In Item c, since $\boxdot\mathcal{A} = \Box\mathcal{A} \cup \{\mathcal{A}\}$ for any model $\mathcal{A}$, we get that $\Box S^{(k)}[\mathcal{B}_i^n] \subseteq \boxdot S^{(k)}[\mathcal{A}_i^n]$ and thus only $S^{(k)}[\mathcal{B}_i^n]$ can be used by Hercules.

Moving into Item d, the cases for $k = n+1$ and for $n = 0$ are trivial; therefore, let $k < n+1$ and $0 < n$. By Lemma 17 it is sufficient to prove the statement for $k = 0$.[7] Now assume, aiming towards a contradiction, that neither of $\mathcal{A}_i^n$ and $S[\mathcal{A}_i^n]$ were chosen by Hercules. We will show that all of the remaining alternatives are isomorphic to some model in $\Box\mathcal{B}_i^n$. Let us assume that $i < 2^n$, then Hercules could not choose any model in $\boxdot\{\mathcal{A}_j^{n-1}(p^1), \overline{\mathcal{A}_i^{n-1}}(p^0), \mathcal{B}_i^{n-1}(p^0)\}$ as all those models belong by definition in $\mathcal{B}_i^n$. Finally, Hercules could not have chosen a model in $\Box\mathcal{A}_i^{n-1}(p^0)$ since $\Box\mathcal{A}_i^{n-1}(p^0) = \Box\mathcal{A}_i^{n-1}(p^1) \subseteq \boxdot\mathcal{B}_i^n$.

[7] Formally we should prove it for $\mathcal{A}_i^n(p^e)$, $\mathcal{B}_i^n(p^e)$, but the proof is otherwise identical.

In Item e-i, we can assume by Lemma 17 that $r = 0$ and $k \geq 2$, and we will show that $S^{(2)}[\mathcal{A}_i^n] \in \boxdot\mathcal{B}_j^n$. This will imply that $\boxdot S^{(2)}[\mathcal{A}_i^n] \subseteq \boxdot\mathcal{B}_j^n$, thus giving us $S^{(k)}[\mathcal{A}_i^n] \in \boxdot\mathcal{B}_j^n$. Since $i \neq j$, $\mathcal{A}_i^{n-1}(p^e)$ is one of the model branches of $\mathcal{B}_j^n$ by definition for some e. Then $S^{(2)}[\mathcal{A}_i^n] = S[\mathcal{A}_i^{n-1}(p^e)] \in \boxdot\mathcal{B}_j^n$.

Finally, we prove Item e-ii. The case for $k = n+1$ is trivial by the definition of the game. Since $r = k < n+1$ is the distinguishing value, by Lemma 17 it follows that $S^{(k)}[\mathcal{A}_i^n]$ is isomorphic to $\mathcal{A}_{i'}^{n-k}(p^e)$ and $S^{(k)}[\mathcal{A}_j^n]$ is isomorphic to $\mathcal{A}_{j'}^{n-k}(p^e)$ for some i, j, e. Thus we can without loss of generality assume that $r = k = 0$. We can assume without loss of generality that $i < 2^n \leq j$, then by definition $\mathcal{A}_i^{n-1} \in \Box\mathcal{B}_i^n$. ◀

▶ **Definition 23.** *In the closed game tree $(T, \prec)$, let $\Lambda(i)$ be the set of leaves η such that*

1. *for every $\eta' \preceq \eta$ there is $r \geq 0$ such that the twins $(S^{(r)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n])$ appear in $\mathbf{L}(\eta') \circ \mathbf{R}(\eta')$ and,*
2. *for every $\zeta \prec \eta$, every child σ of ζ with $\zeta \prec \sigma \preceq \eta$ and every other child σ' of ζ, if $S^{(r)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n]$ are in $\mathbf{L}(\sigma') \circ \mathbf{R}(\sigma')$ then $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ are in $\mathbf{L}(\sigma) \circ \mathbf{R}(\sigma)$ for some $k < r$.*

More informally, if $\eta \in \Lambda(i)$ then the path to η from the root is exactly the path that by *Condition* 2 "locally" minimises the height r of the $(S^{(r)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n])$ twins is chosen.

▶ **Lemma 24.** *For a closed $(\varphi_n, \mathbf{GL})$-MEG game tree $(T, \prec)$ in which the Hydra plays optimally, the following hold:*

(a) $\forall\, i < 2^n$, $\Lambda(i) \neq \emptyset$;

(b) $\forall\, i < 2^n\ \forall\, \eta \in \Lambda(i)\ \forall\, k \leq n+1$ *there is a $\zeta \preceq \eta$ such that k is least with the property that $S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]$ are in $\mathbf{L}(\zeta) \circ \mathbf{R}(\zeta)$;*

(c) *if $0 \leq i < j < 2^n$, then $\Lambda(i) \cap \Lambda(j) = \emptyset$.*

Proof. We will prove each Item by contradiction, starting with Item a. Assume otherwise and let ζ be a maximal node of $(T, \prec)$ for which Conditions 1 and 2 of Definition 23 hold. Let k be the least natural number such that $(S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]) \in \mathbf{L}(\zeta) \circ \mathbf{R}(\zeta)$. As ζ is not a leaf, Hercules has not played a literal move. If Hercules plays a $\vee$-move, then at least one of the two children of ζ, call it ζ', will have $S^{(k)}[\mathcal{A}_i^n] \in \mathbf{L}(\zeta')$. But since $\mathbf{R}(\zeta) = \mathbf{R}(\zeta')$, we get that $(S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n]) \in \mathbf{L}(\zeta') \circ \mathbf{R}(\zeta')$ and this is the least k with such property since $\mathbf{L}(\zeta') \subseteq \mathbf{L}(\zeta)$. The case for the $\wedge$-move comes contrary to our maximality assumption for ζ in the same way as the $\vee$-move case. If Hercules plays a $\Diamond$ or a $\diamondsuit$-move, then by Lemma 22 $(S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n])$ or $(S^{(k+1)}[\mathcal{A}_i^n], S^{(k+1)}[\mathcal{B}_i^n])$ will belong to $\mathbf{L}(\zeta') \circ \mathbf{R}(\zeta')$ with ζ' being the only child of ζ. This satisfies Condition 1, while Condition 2 holds trivially; this contradicts our minimality assumption for ζ. By Lemma 22, Hercules has not played a $\Box$-move, and this leaves only the $\boxdot$-move. In this case, Lemma 22 once more dictates that $(S^{(k)}[\mathcal{A}_i^n], S^{(k)}[\mathcal{B}_i^n])$ will belong in the only child of ζ. As with the $\diamondsuit$-case, this comes contrary to the minimality condition for ζ. As such, we have reached a contradiction and so $\forall i < 2^n$, $\Lambda(i) \neq \emptyset$.

Now for Item b, assume otherwise for some η and k and let $\zeta \preceq \eta$ be greatest with an $r < k$ where $(S^{(r)}[\mathcal{A}_i^n], S^{(r)}[\mathcal{B}_i^n]) \in \mathbf{L}(\zeta) \circ \mathbf{R}(\zeta)$. Clearly $\zeta \prec \eta$ as otherwise a literal move could be played, something only possible if $n+1 = r < k \leq n+1$. Thus there is a child $\sigma \preceq \eta$ of ζ. By Lemma 22 and by the definition of $\Lambda(i)$, no matter what move Hercules performs, $(S^{(s)}[\mathcal{A}_i^n], S^{(s)}[\mathcal{B}_i^n]) \in \mathbf{L}(\sigma) \circ \mathbf{R}(\sigma)$ for some $s \leq k$, contradicting either the maximality of ζ or our original assumption.

Finally we prove Item c. Assume the statement doesn't hold, then there are $i < j < 2^n$ such that $\eta \in \Lambda(i) \cap \Lambda(j)$ for some leaf $\eta \in T$. By Lemma 20 $\mathcal{A}^n_i$ and $\mathcal{A}^n_j$ have some distinguishing value $m \leq n$. Let $\zeta \prec \eta$ be maximal such that the least r and k with

$$(S^{(r)}[\mathcal{A}^n_i], S^{(r)}[\mathcal{B}^n_i]), (S^{(k)}[\mathcal{A}^n_j], S^{(k)}[\mathcal{B}^n_j]) \in \mathbf{L}(\zeta) \circ \mathbf{R}(\zeta) \tag{2}$$

are such that $r \leq m$ or $k \leq m$. Assume that $r = m \leq k$; it will always be the case that one of them will be m from the proof of Item b. Clearly no $\vee$ or $\wedge$-move could have been played at ζ, as then the child $\sigma \prec \eta$ of ζ would either invalidate the choice of ζ, or otherwise, σ would then be a witness of a failure of Condition 2 for η, thus invalidating the assumption $\eta \in \Lambda(i) \cap \Lambda(j)$. Similarly, no $\boxdot$-move could have been played either as it would contradict our choice of the node ζ. If a $\diamondsuit$-move was played at ζ then by Lemma 22 it has to be that $r = k$. Hence, the choice of ζ will be violated as (2) will also hold for its child $\sigma \preceq \eta$. Finally, the $\Diamond$ and $\Box$-moves are simpler cases of the $\diamondsuit$ and $\boxdot$-moves. ◀

It follows that any closed game tree has at least 2^n leaves, yielding our main technical result.

▶ **Proposition 25.** *For every $n \geq 1$, Hercules has no winning strategy of less than $2^n + 2$ moves on the $(\varphi_n, \mathbf{M}_n)$-*MEG*.*

Proof. Let $(T, \prec)$ be a closed game tree of $(\varphi_n, \mathbf{M}_n)$-MEG. By Lemma 24, the sets of leaves $\{\Lambda(i) : i < 2^n\}$ are non-empty and disjoint. Since each leaf represents a literal move being played, Hercules must have played at least 2^n literal moves. As there are at least 2^n leaves, at least one branching move must have been played. Furthermore, by the definition of $\Lambda(i)$, at least one modal rule must have been played. As such, Hercules must have played at least $2^n + 2$ moves. ◀

7 Succinctness

This section contains the main results of our study, first showing an exponential succinctness result in a wide range of Kripke frames. An example of such an application can be found in [12]. We will then examine some additional benefits we can obtain from the connection of the interpolation and the fixed point theorems of **GL**.

7.1 Succinctness of definable fixed points

Proposition 25 is a powerful tool for proving succinctness results. In general, succinctness results for a class of models apply to any larger class. Thus, our results apply not only to **GL** models, but also to a wide range of classes of Kripke models.

▶ **Theorem 26.** *Let $\mathbf{C}$ be any class of Kripke models containing all finite* **GL** *models or all finite* **Grz** *models. Then, there is a sequence of formulas $(\varphi_0(x), \varphi_1(x), \dots)$ which are both modalized and positive on x with $|\varphi_i(x)| = O(i)$ such that for any $i \in \mathbb{N}$ and any $\psi \in \mathcal{L}_{\Diamond\diamondsuit}$, if $\varphi_i(\psi) \equiv \psi$ over $\mathbf{C}$, then $|\psi| \geq 2^i$.*

Proof. The sequence consists of the formulae φ^*_i in Definition 13. Counting the symbols present in the formulas, we get, by a simple induction, $|\varphi^*_n| \leq 41 \cdot n$. Assume, towards a contradiction, that there is some $\psi_n \in \mathcal{L}_\Diamond$ such that $\psi \equiv \mu x.\varphi^*_n(x)$ over $\mathbf{C}$ and $|\psi_n| < 2^n$. But $\psi'_n := q^0_n \wedge \psi_n \equiv \varphi_n$ by Definition 13, making it an $\mathcal{L}_\Diamond$ equivalent to φ_n of size $< 2^n + 2$. However, by Theorem 11, this contradicts Proposition 25. ◀

▶ **Corollary 27.** *Let* $\mathbf{C}$ *be any class of Kripke models containing all finite* $\mathbf{GL}$ *models or all finite* $\mathbf{Grz}$ *models. Then, the languages* $\mathcal{L}^{\mu}_{\Diamond}$ *and* $\mathcal{L}^{\mu}_{\Diamond^{\Diamond}}$ *are exponentially more succinct than* $\mathcal{L}_{\Diamond\Diamond^{\Diamond}}$ *over* $\mathbf{C}$*. To be precise, for* $\mathcal{L}^{\mu} \in \{\mathcal{L}^{\mu}_{\Diamond}, \mathcal{L}^{\mu}_{\Diamond^{\Diamond}}\}$*, there is a sequence of* $\mathcal{L}^{\mu}$ *formulas* $(\varphi_0, \varphi_1, \ldots)$ *with* $|\varphi_i| = O(i)$ *such that for any* $i \in \mathbb{N}$ *and any* $\psi \in \mathcal{L}_{\Diamond\Diamond^{\Diamond}}$*, if* $\varphi_i \equiv \psi$ *over* $\mathbf{C}$*, then* $|\psi| \geq 2^i$*.*

Proof. Observe that if $\varphi \equiv \psi$ over $\mathbf{C}$, then it is also the case over $\mathbf{GL}$ or $\mathbf{Grz}$. As such, by Theorem 26, the sequence of $\mathcal{L}^{\mu}_{\Diamond}$ formulae φ_i of Definition 13 is exponentially more succinct than their $\mathcal{L}_{\Diamond\Diamond^{\Diamond}}$ counterparts. Finally, if we consider the sequence of $\mathcal{L}^{\mu}_{\Diamond^{\Diamond}}$ formulae ψ_i that are the same as φ_i if we were to substitute $\Diamond$ by $\Diamond^{\Diamond}$, we know that $\psi_i \equiv \varphi_i$. ◀

7.2 Size of interpolants

A somewhat surprising link to this study is that with the interpolation theorem. The interpolation has been studied in many logics and via both model theoretic and proof theoretic means [8, 15]. However, while proof theoretic proofs of the interpolation theorem give us bounds for the proof size, no good bounds on the interpolants can be immediately derived. The link in this case is primarily tied to one of the proofs of the fixed-point theorem which we will briefly present here. Let us first recall the interpolation theorem.

▶ **Theorem 28** (Craig interpolation for $\mathbf{GL}$)**.** *Let* φ *and* ψ *be such that* $\mathbf{GL} \vdash \varphi \to \psi$*. There exists some formula* σ *containing only variables occurring in both* φ *and* ψ *such that*

$$\mathbf{GL} \vdash \varphi \to \sigma \text{ and } \mathbf{GL} \vdash \sigma \to \psi.$$

Proof. See e.g. [8]. ◀

Interpolation then easily implies the definability theorem of Beth.

▶ **Theorem 29** (Beth's definability theorem for $\mathbf{GL}$ [5])**.** *For any* $\varphi(x)$ *and* y *different from* x*, if* $\mathbf{GL} \vdash \varphi(x) \wedge \varphi(y) \to (x \leftrightarrow y)$ *then there is some formula* ψ *containing only variables in* $\varphi(x)$ *excluding* x *such that* $\mathbf{GL} \vdash \varphi(x) \to (\psi \leftrightarrow x)$*.*

Proof. By the assumptions $\mathbf{GL} \vdash \varphi(x) \wedge x \to (\varphi(y) \to y)$, then ψ is the formula given by Craig's interpolation theorem. For more details, see e.g. [7]. ◀

The proof proceeds by one proving uniqueness of fixed points in the following sense by Bernardi.

▶ **Theorem 30** (Bernardi [3])**.** *Let* $\varphi(x)$ *be modalized in* x*. Then*

$$\mathbf{GL} \vdash \boxdot(\varphi(x) \leftrightarrow x) \wedge \boxdot(\varphi(y) \leftrightarrow y) \to (x \leftrightarrow y).$$

Proof. See [3, 4, 7]. ◀

Then by the Beth definability theorem we can find some appropriate ψ such that $\mathbf{GL} \vdash \boxdot(\varphi(x) \leftrightarrow x) \to (\psi \leftrightarrow x)$. As a result, succinctness results for the fixed-point theorem of $\mathbf{GL}$ can be directly applied to provide succinctness results on the size of the interpolants.

▶ **Corollary 31.** *There exist sequences of formulae* $(\varphi_0, \varphi_1, \ldots)$ *and* $(\psi_0, \psi_1, \ldots)$ *both of size* $|\varphi_i|, |\psi_i| \leq O(i)$ *and such that* $\mathbf{GL} \vdash \varphi_i \to \psi_i$ *for every* i*, while every interpolant* σ_i *of* φ_i *and* ψ_i *is of size* $|\sigma_i| = 2^{\Omega(i)}$*.*

Proof. The sequences consist of the formulae $\varphi_i(x) := \boxdot(\varphi_i^*(x) \leftrightarrow x) \wedge x$ and $\psi_i(y) := \boxdot(\varphi_i^*(y) \leftrightarrow y) \to y$. Then any interpolant σ_i of φ_i and ψ_i is necessarily a fixed point of $\varphi_i^*(x)$ which by Theorem 26 must have size at least exponential in i. ◀

8 Conclusion

We have shown that the μ-calculus with only one occurrence of a fixed point operator is exponentially more succinct than basic modal logic, even when equipped with a reflexive modality and even in the setting of **GL**, where fixed points are already definable. This yields a lower bound on the fixed point formulas provided by Theorems 6 and 7, hence providing the first lower bounds for these celebrated results. This places the μ-calculus over **GL** or **Grz** as a powerful formal system in terms of expressivity, despite the theoretical definability of fixed points.

There is a small gap between the upper and lower bound for the fixed-point theorem for **GL**, with the lower bound being $2^{\Omega(n)}$ and the upper $2^{O(n \log(n))}$; it is unclear which of the two is tighter. In contrast, for **Grz** we obtain a larger gap of $2^{\Omega(n)}$ vs. $2^{O(n^3)}$; in this case we believe that the upper bound can be significantly improved, which we plan to address in future work.

Our proof of succinctness of the interpolants is a rather Post Hoc expansion of our succinctness for the fixed-point results coming directly from the bibliography. As such it is restricted to interpolants over **GL**. As far as we know, a result of this form is new and hence, a lucrative open problem would be expanding the methods of model equivalence games to get succinctness lower bounds for interpolants over **S**4 or **K**4 frames as an example.

Finally, we note that our techniques provide lower bound on formula length but *not* on the number of subformulas, or equivalently, the size of dag-like representations of formulas. This is particularly relevant since issues such as complexity can be bounded with respect to the latter measure, which may in fact be much smaller. Finding lower bounds on the number of subformulas would require a non-trivial modification of the model equivalence game; we leave the development of such games and the question of whether the μ-calculus remains exponentially succinct over dag-like formulas as challenging avenues for future research.

References

1 M. Adler and N. Immerman. An $n!$ lower bound on formula size. *ACM Transactions on Computational Logic*, 4(3):296–314, 2003. doi:10.1145/772062.772064.

2 Gaëlle Fontaine Balder ten Cate and Tadeusz Litak. Some modal aspects of xpath. *Journal of Applied Non-Classical Logics*, 20(3):139–171, 2010. doi:10.3166/JANCL.20.139-171.

3 Claudio Bernardi. The fixed-point theorem for diagonalizable algebras. *Studia Logica*, 34(3):239–251, 1975. doi:10.1007/bf02125226.

4 Claudio Bernardi. The uniqueness of the fixed-point in every diagonalizable algebra. *Studia Logica*, 35(4):335–343, 1976. doi:10.1007/bf02123401.

5 E.W. Beth. On padoa's method in the theory of definition. *Indagationes Mathematicae (Proceedings)*, 56:330–339, 1953. doi:10.1016/S1385-7258(53)50042-3.

6 A. Blass. Infinitary combinatorics and modal logic. *Journal of Symbolic Logic*, 55(2):761–778, 1990. doi:10.2307/2274663.

7 G. S. Boolos. *The Logic of Provability*. Cambridge University Press, 1993.

8 A. Chagrov and M. Zakharyaschev. *Modal Logic*, volume 35 of *Oxford logic guides*. Oxford University Press, 1997.

9 A. Dawar and M. Otto. Modal characterisation theorems over special classes of frames. *Annals of Pure and Applied Logic*, 161:1–42, 2009. doi:10.1016/J.APAL.2009.04.002.

10 K. Eickmeyer, M. Elberfeld, and F. Harwath. Expressivity and succinctness of order-invariant logics on depth-bounded structures. In *Proceedings of MFCS 2014*, pages 256–266, 2014.

11 David Fernández-Duque and Petar Iliev. Succinctness in subsystems of the spatial μ-calculus. *FLAP*, 5(4):827–874, 2018. URL: https://www.collegepublications.co.uk/downloads/ifcolog00024.pdf.

12 David Fernández-Duque and Konstnatinos Papafilippou. The universal tangle for spatial reasoning. In Sarah Alice Gaggl, Maria Vanina Martinez, and Magdalena Ortiz, editors, *Logics in Artificial Intelligence - 18th European Conference, JELIA 2023, Dresden, Germany, September 20-22, 2023, Proceedings*, volume 14281 of *Lecture Notes in Computer Science*, pages 814–827. Springer, 2023. doi:10.1007/978-3-031-43619-2_55.

13 S. Figueira and D. Gorín. On the size of shortest modal descriptions. *Advances in Modal Logic*, 8:114–132, 2010.

14 T. French, W. van der Hoek, P. Iliev, and B. Kooi. Succinctness of epistemic languages. In T. Walsh, editor, *Proceedings of IJCAI*, pages 881–886, 2011.

15 Dov M. Gabbay and Larisa Maksimova. *Interpolation and Definability: Modal and Intuitionistic Logics*. Oxford University Press, May 2005. doi:10.1093/acprof:oso/9780198511748.001.0001.

16 Zachary Gleit and Warren Goldfarb. Characters and fixed-points in provability logic. *Notre Dame Journal of Formal Logic*, 31(1):26–36, 1989. doi:10.1305/ndjfl/1093635330.

17 M. Grohe and N. Schweikardt. The succinctness of first-order logic on linear orders. *Logical Methods in Computer Science*, 1:1–25, 2005.

18 L. Hella and M. Vilander. The succinctness of first-order logic over modal logic via a formula size game. In *Advances in Modal Logic*, volume 11, pages 401–419, 2016.

19 A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990. doi:10.1007/BF02122698.

20 Lisa Reidhaar-Olson. A new proof of the fixed-point theorem of probability logic. *Notre Dame J. Formal Log.*, 31:37–43, 1989. URL: https://api.semanticscholar.org/CorpusID:7685381, doi:10.1305/NDJFL/1093635331.

21 Giovanni Sambin. An effective fixed-point theorem in intuitionistic diagonalizable algebras. *Studia Logica*, 35(4):345–361, 1976. doi:10.1007/bf02123402.

22 Craig Smoryński. Modal logic and self-reference. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, pages 1–53. Springer Netherlands, Dordrecht, 2004. doi:10.1007/978-94-017-0466-3_1.

23 R. M. Solovay. Provability interpretations of modal logic. *Israel Journal of Mathematics*, 25:287–304, 1976.

24 A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.*, 5(2):285–309, 1955. URL: https://projecteuclid.org:443/euclid.pjm/1103044538.

25 W. van der Hoek, P. Iliev, and B. Kooi. On the relative succinctness of modal logics with union, intersection and quantification. In *Proceedings of AAMAS*, pages 341–348, 2014.

26 VII Soviet Symposium on Logic. *On modal "companions" of superintuitionistic logics*, Kiev, 1976. Russian.

The Complexity of Deciding Characteristic Formulae in Van Glabbeek's Branching-Time Spectrum

Luca Aceto
Department of Computer Science, Reykjavik University, Iceland
Gran Sasso Science Institute, L'Aquila, Italy

Antonis Achilleos
Department of Computer Science, Reykjavik University, Iceland

Aggeliki Chalki
Department of Computer Science, Reykjavik University, Iceland

Anna Ingólfsdóttir
Department of Computer Science, Reykjavik University, Iceland

Abstract

Characteristic formulae give a complete logical description of the behaviour of processes modulo some chosen notion of behavioural semantics. They allow one to reduce equivalence or preorder checking to model checking, and are exactly the formulae in the modal logics characterizing classic behavioural equivalences and preorders for which model checking can be reduced to equivalence or preorder checking.

This paper studies the complexity of determining whether a formula is characteristic for some process in each of the logics providing modal characterizations of the simulation-based semantics in van Glabbeek's branching-time spectrum. Since characteristic formulae in each of those logics are exactly the satisfiable and prime ones, this article presents complexity results for the satisfiability and primality problems, and investigates the boundary between modal logics for which those problems can be solved in polynomial time and those for which they become computationally hard.

Amongst other contributions, this article also studies the complexity of constructing characteristic formulae in the modal logics characterizing simulation-based semantics, both when such formulae are presented in explicit form and via systems of equations.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Complexity theory and logic

Keywords and phrases Characteristic formulae, prime formulae, bisimulation, simulation relations, modal logics, complexity theory, satisfiability

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.26

Related Version *Full Version*: `https://doi.org/10.48550/arXiv.2405.13697` [1]

Funding This work has been funded by the projects "Open Problems in the Equational Logic of Processes (OPEL)" (grant no. 196050), "Mode(l)s of Verification and Monitorability" (MoVeMnt) (grant no. 217987), and "Learning and Applying Probabilistic Systems" (grant no. 206574-051) of the Icelandic Research Fund.

Acknowledgements The authors thank the anonymous reviewers for comments that led to improvements in the paper. This paper is dedicated to the memory of Rance Cleaveland (1961–2024), who used characteristic formulae to compute behavioural relations, logically and efficiently.

1 Introduction

Several notions of behavioural relations have been proposed in concurrency theory to describe when one process is a suitable implementation of another. Many such relations have been catalogued by van Glabbeek in his seminal linear-time/branching-time spectrum [22], together with a variety of alternative ways of describing them including testing scenarios and axiom systems. To our mind, modal characterizations of behavioural equivalences and preorders are some of the most classic and pleasing results in concurrency theory – see, for instance, [25] for the seminal Hennessy-Milner theorem and [12, 16, 17, 22] for similar results for other relations in van Glabbeek's spectrum and other settings. By way of example, in their archetypal modal characterization of bisimilarity, Hennessy and Milner have shown in [25] that, under a mild finiteness condition, two processes are bisimilar if, and only if, they satisfy the same formulae in a multi-modal logic that is now often called Hennessy-Milner logic. Apart from its intrinsic theoretical interest, this seminal logical characterization of bisimilarity means that, when two processes are *not* bisimilar, there is always a formula that distinguishes between them. Such a formula describes a reason why the two processes are not bisimilar, provides useful debugging information and can be algorithmically constructed over finite processes – see, for instance, [8, 14] and [35], where Martens and Groote show that, in general, computing minimal distinguishing Hennessy-Milner formulae is NP-hard.

On the other hand, the Hennessy-Milner theorem seems to be less useful to show that two processes *are* bisimilar, since that would involve verifying that they satisfy the same formulae, and there are infinitely many of those. However, as shown in works such as [3, 6, 12, 23, 39], the logics that underlie classic modal characterization theorems for equivalences and preorders over processes allow one to express *characteristic formulae*. Intuitively, a characteristic formula $\chi(p)$ for a process p gives a complete logical characterization of the behaviour of p modulo the behavioural semantics of interest $\lesssim$, in the sense that any process is related to p with respect to $\lesssim$ if, and only if, it satisfies $\chi(p)$.[1] Since the formula $\chi(p)$ can be constructed from p, characteristic formulae reduce the problem of checking whether a process q is related to p by $\lesssim$ to a model checking problem, viz. whether q satisfies $\chi(p)$. See, for instance, the classic reference [15] for applications of this approach.

Characteristic formulae, thus, allow one to reduce equivalence and preorder checking to model checking. But what model checking problems can be reduced to equivalence/preorder checking ones? To the best of our knowledge, that question was first studied by Boudol and Larsen in [11] in the setting of modal refinement over modal transition systems. See [3, 4] for other contributions in that line of research. The aforementioned articles showed that characteristic formulae coincide with those that are *satisfiable* and *prime*. (A formula is prime if whenever it entails a disjunction $\varphi_1 \vee \varphi_2$, then it must entail φ_1 or φ_2.) Moreover, characteristic formulae with respect to bisimilarity coincide with the formulae that are satisfiable and *complete* [7]. (A modal formula is complete if, for each formula φ, it entails either φ or its negation.) The aforementioned results give semantic characterizations of the formulae that are characteristic within the logics that correspond to the behavioural semantics in van Glabbeek's spectrum. Those characterizations tell us for what logical specifications model checking can be reduced to equivalence or preorder checking. However,

[1] Formulae akin to characteristic ones first occurred in the study of equivalence of structures using first-order formulae up to some quantifier rank. See, for example, the survey paper [40] and the textbook [20]. The existence of formulae in first-order logic with counting that characterize graphs up to isomorphism has significantly contributed to the study of the complexity of the Graph Isomorphism problem – see, for instance, [13, 30].

given a specification expressed as a modal formula, can one decide whether that formula is characteristic and therefore can be model checked using algorithms for behavioural equivalences or preorders? And, if so, what is the complexity of checking whether a formula is characteristic? Perhaps surprisingly, those questions were not addressed in the literature until the recent papers [2, 7], where it is shown that, in the setting of the modal logics that characterize bisimilarity over natural classes of Kripke structures and labelled transition systems, the problem of checking whether a formula is characteristic for some process modulo bisimilarity is computationally hard and, typically, has the same complexity as validity checking, which is PSPACE-complete for Hennessy-Milner logic and EXP-complete for its extension with fixed-point operators [26, 33] and the μ-calculus [31].

The aforementioned hardness results for the logics characterizing bisimilarity tell us that deciding whether a formula is characteristic in bisimulation semantics is computationally hard. But what about the less expressive logics that characterize the coarser semantics in van Glabbeek's spectrum? And for what logics characterizing relations in the spectrum does computational hardness manifest itself? Finally, what is the complexity of computing a characteristic formula for a process?

The aim of this paper is to answer the aforementioned questions for some of the simulation-based semantics in the spectrum. In particular, we study the complexity of determining whether a formula is characteristic modulo the simulation [36], complete simulation and ready simulation preorders [10, 34], as well as the trace simulation and the n-nested simulation preorders [24]. Since characteristic formulae are exactly the satisfiable and prime ones for each behavioural relation in van Glabbeek's spectrum [3], the above-mentioned tasks naturally break down into studying the complexity of satisfiability and primality checking for formulae in the fragments of Hennessy-Milner logic that characterize those preorders. By using a reduction to the, seemingly unrelated, reachability problem in *alternating graphs*, as defined by Immerman in [28, Definition 3.24], we discover that both those problems are decidable in polynomial time for the simulation and the complete simulation preorders, as well as for the ready simulation preorder when the set of actions has constant size. On the other hand, when the set of actions is unbounded (that is, it is an input of the algorithmic problem at hand), the problems of checking satisfiability and primality for formulae in the logic characterizing the ready simulation preorder are NP-complete and coNP-complete respectively. We also show that deciding whether a formula is characteristic in that setting is US-hard [9] (that is, it is at least as hard as the problem of deciding whether a given Boolean formula has exactly one satisfying truth assignment) and belongs to DP, which is the class of languages that are the intersection of one language in NP and of one in coNP [38].[2] These negative results are in stark contrast with the positive results for the simulation and the complete simulation preorder, and indicate that augmenting the logic characterizing the simulation preorder with formulae that state that a process cannot perform a given action suffices to make satisfiability and primality checking computationally hard. In passing, we also prove that, in the presence of at least two actions, (1) for the logics characterizing the trace simulation and 2-nested simulation preorders, satisfiability and primality checking are NP-complete and coNP-hard respectively, and deciding whether a formula is characteristic is US-hard, (2) for the logic that characterizes the trace simulation preorder, deciding whether a formula is characteristic is fixed-parameter tractable [18], with the modal depth of the input formula as the parameter, when the size of the action set is a constant, and (3) deciding whether

[2] The class DP contains both NP and coNP, and is contained in the class of problems that can be solved in polynomial time with an NP oracle.

a formula is characteristic in the modal logic for the 3-nested simulation preorder [24] is PSPACE-hard. (The proof of the last result relies on "simulating" Ladner's reduction proving the PSPACE-hardness of satisfiability for modal logic [32] using the limited alternations of modal operators allowed by the logic for the 3-nested simulation preorder.)

We also study the complexity of computing characteristic formulae for finite, loop-free processes modulo the above-mentioned simulation semantics. To do so, we consider two different representations for formulae, namely an explicit form, where formulae are given by strings of symbols generated by their respective grammars, and a declarative form, where formulae are described by systems of equations. We prove that, even for the coarsest semantics we consider, such as the simulation and complete simulation preorders, computing the characteristic formula in explicit form for a finite, loop-free process cannot be done in polynomial time, unless $\mathsf{P} = \mathsf{NP}$. On the other hand, the characteristic formula for a process modulo the preorders we study, apart from the trace simulation preorder, can be computed in polynomial time if the output is given in declarative form. Intuitively, this is due to the fact that, unlike the explicit form, systems of equations allow for sharing of subformulae and there are formulae for which this sharing leads to an exponentially more concise representation. Finally, in sharp contrast to that result, we prove that, modulo the trace simulation preorder, even if characteristic formulae are always of polynomial declaration size and polynomial equational length, they cannot be efficiently computed unless $\mathsf{P} = \mathsf{NP}$. In passing, we remark that all the aforementioned lower and upper bounds hold also for finite processes with loops, provided that, as done in [6, 29, 39], we add greatest fixed points or systems of equations interpreted as greatest fixed points to the modal logics characterizing the semantics we study in this article.

We summarize our results in Table 2. We provide their proofs in the technical appendices of the full version of the paper [1].

2 Preliminaries

In this paper, we model processes as finite, loop-free *labelled transition systems* (LTS). A finite LTS is a triple $\mathcal{S} = (P, A, \longrightarrow)$, where P is a finite set of states (or processes), A is a finite, non-empty set of actions and $\longrightarrow \subseteq P \times A \times P$ is a transition relation. As usual, we use $p \xrightarrow{a} q$ instead of $(p, a, q) \in \longrightarrow$. For each $t \in A^*$, we write $p \xrightarrow{t} q$ to mean that there is a sequence of transitions labelled with t starting from p and ending at q. An LTS is *loop-free* iff $p \xrightarrow{t} p$ holds only when t is the empty trace ε. A process q is *reachable* from p if $p \xrightarrow{t} q$, for some $t \in A^*$. We define the *size* of an LTS $\mathcal{S} = (P, A, \longrightarrow)$, denoted by $|\mathcal{S}|$, to be $|P| + |\longrightarrow|$. The *size of a process* $p \in P$, denoted by $|p|$, is the cardinality of $\mathrm{reach}(p) = \{q \mid q \text{ is reachable from } p\}$ plus the cardinality of the set $\longrightarrow$ restricted to $\mathrm{reach}(p)$. We define the set of *initials* of p, denoted $I(p)$, as the set $\{a \in A \mid p \xrightarrow{a} p' \text{ for some } p' \in P\}$. We write $p \xrightarrow{a}$ if $a \in I(p)$, $p \overset{a}{\not\longrightarrow}$ if $a \notin I(p)$, and $p \not\longrightarrow$ if $I(p) = \emptyset$. A sequence of actions $t \in A^*$ is a trace of p if there is a q such that $p \xrightarrow{t} q$. We denote the set of traces of p by $\mathrm{traces}(p)$. The *depth* of a finite, loop-free process p, denoted by $\mathrm{depth}(p)$, is the length of a longest trace t of p.

In what follows, we shall often describe finite, loop-free processes using the fragment of Milner's CCS [37] given by the following grammar:

$$p ::= 0 \;\mid\; a.p \;\mid\; p + p,$$

where $a \in A$. For each action a and terms p, p', we write $p \xrightarrow{a} p'$ iff

(i) $p = a.p'$ or

(ii) $p = p_1 + p_2$, for some p_1, p_2, and $p_1 \xrightarrow{a} p'$ or $p_2 \xrightarrow{a} p'$ holds.

In this paper, we consider the following relations in van Glabbeek's spectrum: simulation, complete simulation, ready simulation, trace simulation, 2-nested simulation, 3-nested simulation, and bisimilarity. Their definitions are given below.

▶ **Definition 1** ([37, 22, 3]). *We define each of the following preorders as the largest binary relation over P that satisfies the corresponding condition.*

(a) Simulation preorder (S): $p \lesssim_S q \Leftrightarrow$ *for all* $p \xrightarrow{a} p'$ *there exists some* $q \xrightarrow{a} q'$ *such that* $p' \lesssim_S q'$.

(b) Complete simulation (CS): $p \lesssim_{CS} q \Leftrightarrow$
- **(i)** *for all* $p \xrightarrow{a} p'$ *there exists some* $q \xrightarrow{a} q'$ *such that* $p' \lesssim_{CS} q'$, *and*
- **(ii)** $I(p) = \emptyset$ *iff* $I(q) = \emptyset$.

(c) Ready simulation (RS): $p \lesssim_{RS} q \Leftrightarrow$
- **(i)** *for all* $p \xrightarrow{a} p'$ *there exists some* $q \xrightarrow{a} q'$ *such that* $p' \lesssim_{RS} q'$, *and*
- **(ii)** $I(p) = I(q)$.

(d) Trace simulation (TS): $p \lesssim_{TS} q \Leftrightarrow$
- **(i)** *for all* $p \xrightarrow{a} p'$ *there exists some* $q \xrightarrow{a} q'$ *such that* $p' \lesssim_{TS} q'$, *and*
- **(ii)** $\mathrm{traces}(p) = \mathrm{traces}(q)$.

(e) n-Nested simulation (nS), *where* $n \geq 1$, *is defined inductively as follows: The* 1*-nested simulation preorder* $\lesssim_{1S}$ *is* $\lesssim_S$, *and the* n*-nested simulation preorder* $\lesssim_{nS}$ *for* $n > 1$ *is the largest relation such that* $p \lesssim_{nS} q \Leftrightarrow$
- **(i)** *for all* $p \xrightarrow{a} p'$ *there exists some* $q \xrightarrow{a} q'$ *such that* $p' \lesssim_{nS} q'$, *and*
- **(ii)** $q \lesssim_{(n-1)S} p$.

(f) Bisimilarity (BS): $\lesssim_{BS}$ *is the largest symmetric relation satisfying the condition defining the simulation preorder.*

It is well-known that bisimilarity is an equivalence relation and all the other relations are preorders [22, 37]. We sometimes write $p \sim q$ instead of $p \lesssim_{BS}$. Moreover, we have that $\sim \subsetneq \lesssim_{3S} \subsetneq \lesssim_{2S} \subsetneq \lesssim_{TS} \subsetneq \lesssim_{RS} \subsetneq \lesssim_{CS} \subsetneq \lesssim_S$ – see [22].

▶ **Definition 2** (Kernels of the preorders). *For each* $X \in \{S, CS, RS, TS, 2S, 3S\}$, *the kernel* $\equiv_X$ *of* $\lesssim_X$ *is the equivalence relation defined thus: for every* $p, q \in P$, $p \equiv_X q$ *iff* $p \lesssim_X q$ *and* $q \lesssim_X p$.

Each relation $\lesssim_X$, where $X \in \{S, CS, RS, TS, 2S, 3S, BS\}$, is characterized by a fragment $\mathcal{L}_X$ of Hennessy-Milner logic, **HML**, defined as follows [22, 3].

▶ **Definition 3.** *For* $X \in \{S, CS, RS, TS, 2S, 3S, BS\}$, $\mathcal{L}_X$ *is defined by the corresponding grammar given below* ($a \in A$)*:*

(a) $\mathcal{L}_S$: $\varphi_S ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_S \wedge \varphi_S \mid \varphi_S \vee \varphi_S \mid \langle a \rangle \varphi_S$.

(b) $\mathcal{L}_{CS}$: $\varphi_{CS} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{CS} \wedge \varphi_{CS} \mid \varphi_{CS} \vee \varphi_{CS} \mid \langle a \rangle \varphi_{CS} \mid \mathbf{0}$, *where* $\mathbf{0} = \bigwedge_{a \in A} [a]\mathbf{ff}$.

(c) $\mathcal{L}_{RS}$: $\varphi_{RS} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{RS} \wedge \varphi_{RS} \mid \varphi_{RS} \vee \varphi_{RS} \mid \langle a \rangle \varphi_{RS} \mid [a]\mathbf{ff}$.

(d) $\mathcal{L}_{TS}$: $\varphi_{TS} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{TS} \wedge \varphi_{TS} \mid \varphi_{TS} \vee \varphi_{TS} \mid \langle a \rangle \varphi_{TS} \mid \psi_{TS}$, *where* $\psi_{TS} ::= \mathbf{ff} \mid [a]\psi_{TS}$.

(e) $\mathcal{L}_{2S}$: $\varphi_{2S} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{2S} \wedge \varphi_{2S} \mid \varphi_{2S} \vee \varphi_{2S} \mid \langle a \rangle \varphi_{2S} \mid \neg\varphi_S$.

(f) $\mathcal{L}_{3S}$: $\varphi_{3S} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{3S} \wedge \varphi_{3S} \mid \varphi_{3S} \vee \varphi_{3S} \mid \langle a \rangle \varphi_{3S} \mid \neg\varphi_{2S}$.

(g) **HML** ($\mathcal{L}_{BS}$): $\varphi_{BS} ::= \mathbf{tt} \mid \mathbf{ff} \mid \varphi_{BS} \wedge \varphi_{BS} \mid \varphi_{BS} \vee \varphi_{BS} \mid \langle a \rangle \varphi_{BS} \mid [a]\varphi_{BS} \mid \neg\varphi_{BS}$.

Note that the explicit use of negation in the grammar for $\mathcal{L}_{BS}$ is unnecessary. However, we included the negation operator explicitly so that $\mathcal{L}_{BS}$ extends syntactically each of the other modal logics presented in Definition 3.

Given a formula $\varphi \in \mathcal{L}_{BS}$, the *modal depth* of φ, denoted by $\mathrm{md}(\varphi)$, is the maximum nesting of modal operators in φ. (See [1, Appendix A] for the formal definition.)

Truth in an LTS $\mathcal{S} = (P, A, \longrightarrow)$ is defined via the satisfaction relation $\models$ as follows:

$p \models \mathbf{tt}$ and $p \not\models \mathbf{ff}$;

$p \models \neg\varphi$ iff $p \not\models \varphi$;

$p \models \varphi \wedge \psi$ iff both $p \models \varphi$ and $p \models \psi$;

$p \models \varphi \vee \psi$ iff $p \models \varphi$ or $p \models \psi$;

$p \models \langle a \rangle \varphi$ iff there is some $p \xrightarrow{a} q$ such that $q \models \varphi$;

$p \models [a]\varphi$ iff for all $p \xrightarrow{a} q$ it holds that $q \models \varphi$.

If $p \models \varphi$, we say that φ is true, or satisfied, in p. If φ is satisfied in every process in every LTS, we say that φ is valid. Formula φ_1 entails φ_2, denoted by $\varphi_1 \models \varphi_2$, if every process that satisfies φ_1 also satisfies φ_2. Moreover, φ_1 and φ_2 are logically equivalent, denoted by $\varphi_1 \equiv \varphi_2$, if $\varphi_1 \models \varphi_2$ and $\varphi_2 \models \varphi_1$. A formula φ is *satisfiable* if there is a process that satisfies φ. Finally, $\mathrm{Sub}(\varphi)$ denotes the set of subformulae of formula φ.

For $\mathcal{L} \subseteq \mathcal{L}_{BS}$, we define the dual fragment of $\mathcal{L}$ to be $\overline{\mathcal{L}} = \{\varphi \mid \neg\varphi \in \mathcal{L}\}$, where $\neg\mathbf{tt} = \mathbf{ff}$, $\neg\mathbf{ff} = \mathbf{tt}$, $\neg(\varphi \wedge \psi) = \neg\varphi \vee \neg\psi$, $\neg(\varphi \vee \psi) = \neg\varphi \wedge \neg\psi$, $\neg[a]\varphi = \langle a \rangle \neg\varphi$, $\neg\langle a \rangle \varphi = [a]\neg\varphi$, and $\neg\neg\varphi = \varphi$. It is not hard to see that $p \models \neg\varphi$ iff $p \not\models \varphi$, for every process p. Given a process p, we define $\mathcal{L}(p) = \{\varphi \in \mathcal{L} \mid p \models \varphi\}$. A simplification of the Hennessy-Milner theorem gives a modal characterization of bisimilarity over finite processes. An analogous result is true for every preorder examined in this paper.

▶ **Theorem 4** (Hennessy-Milner theorem [25]). *For all processes p, q in a finite LTS, $p \sim q$ iff $\mathcal{L}_{BS}(p) = \mathcal{L}_{BS}(q)$.*

▶ **Proposition 5** ([22, 3]). *Let $X \in \{S, CS, RS, TS, 2S, 3S\}$. Then $p \lesssim_X q$ iff $\mathcal{L}_X(p) \subseteq \mathcal{L}_X(q)$, for all $p, q \in P$.*

▶ Remark 6. Neither **ff** nor disjunction are needed in several of the modal characterizations presented in the above result. The reason for adding those constructs to all the logics is that doing so makes our subsequent results more general and uniform. For example, having **ff** and disjunction in all logics allows us to provide algorithms that determine whether a formula in a logic $\mathcal{L}$ is prime with respect to a sublogic.

▶ **Definition 7** ([11, 4]). *Let $\mathcal{L} \subseteq \mathcal{L}_{BS}$. A formula $\varphi \in \mathcal{L}_{BS}$ is* prime in $\mathcal{L}$ *if for all $\varphi_1, \varphi_2 \in \mathcal{L}$, $\varphi \models \varphi_1 \vee \varphi_2$ implies $\varphi \models \varphi_1$ or $\varphi \models \varphi_2$.*

When the logic $\mathcal{L}$ is clear from the context, we say that φ is prime. Note that every unsatisfiable formula is trivially prime in $\mathcal{L}$, for every $\mathcal{L}$.

▶ **Example 8.** The formula $\langle a \rangle \mathbf{tt}$ is prime in $\mathcal{L}_S$. Indeed, let $\varphi_1, \varphi_2 \in \mathcal{L}_S$ and assume that $\langle a \rangle \mathbf{tt} \models \varphi_1 \vee \varphi_2$. Since $a.\mathsf{0} \models \langle a \rangle \mathbf{tt}$, without loss of generality, we have that $a.\mathsf{0} \models \varphi_1$. We claim that $\langle a \rangle \mathbf{tt} \models \varphi_1$. To see this, let p be some process such that $p \models \langle a \rangle \mathbf{tt}$ – that is, a process such that $p \xrightarrow{a} p'$ for some p'. It is easy to see that $a.\mathsf{0} \lesssim_S p$. Since $a.\mathsf{0} \models \varphi_1$, Proposition 5 yields that $p \models \varphi_1$, proving our claim and the primality of $\langle a \rangle \mathbf{tt}$. On the other hand, the formula $\langle a \rangle \mathbf{tt} \vee \langle b \rangle \mathbf{tt}$ is not prime in $\mathcal{L}_S$. Indeed, $\langle a \rangle \mathbf{tt} \vee \langle b \rangle \mathbf{tt} \models \langle a \rangle \mathbf{tt} \vee \langle b \rangle \mathbf{tt}$, but neither $\langle a \rangle \mathbf{tt} \vee \langle b \rangle \mathbf{tt} \models \langle a \rangle \mathbf{tt}$ nor $\langle a \rangle \mathbf{tt} \vee \langle b \rangle \mathbf{tt} \models \langle b \rangle \mathbf{tt}$ hold.

The definition of a characteristic formula within logic $\mathcal{L}$ is given next.

▶ **Definition 9** ([5, 23, 39]). *Let $\mathcal{L} \subseteq \mathcal{L}_{BS}$. A formula $\varphi \in \mathcal{L}$ is* characteristic for $p \in P$ within $\mathcal{L}$ *iff, for all $q \in P$, it holds that $q \models \varphi \Leftrightarrow \mathcal{L}(p) \subseteq \mathcal{L}(q)$. We denote by $\chi(p)$ the unique characteristic formula for p with respect to logical equivalence.*

▶ Remark 10. Let $X \in \{S, CS, RS, TS, 2S, 3S, BS\}$. In light of Theorem 4 and Proposition 5, a formula $\varphi \in \mathcal{L}_X$ is characteristic for p within $\mathcal{L}_X$ iff, for all $q \in P$, it holds that $q \models \varphi \Leftrightarrow p \lesssim_X q$. This property is often used as an alternative definition of characteristic formula for process p modulo $\lesssim_X$. In what follows, we shall employ the two definitions interchangeably.

In [3, Table 1 and Theorem 5], Aceto, Della Monica, Fabregas, and Ingólfsdóttir presented characteristic formulae for each of the semantics we consider in this paper, and showed that characteristic formulae are exactly the satisfiable and prime ones.

▶ **Proposition 11** ([3]). *For every $X \in \{S, CS, RS, TS, 2S\}$, $\varphi \in \mathcal{L}_X$ is characteristic for some process within $\mathcal{L}_X$ iff φ is satisfiable and prime in $\mathcal{L}_X$.*

▶ Remark 12. Proposition 11 is the only result we use from [3] and we employ it as a "black box". The (non-trivial) methods used in the proof of that result given in that reference do not play any role in our technical developments.

We note, in passing, that the article [3] does not deal explicitly with $3S$. However, its results apply to all the n-nested simulation preorders.

We can also consider characteristic formulae modulo equivalence relations as follows.

▶ **Definition 13.** *Let $X \in \{S, CS, RS, TS, 2S, 3S, BS\}$. A formula $\varphi \in \mathcal{L}_X$ is characteristic for $p \in P$ modulo $\equiv_X$ iff for all $q \in P$, it holds that $q \models \varphi \Leftrightarrow \mathcal{L}_X(p) = \mathcal{L}_X(q)$.*[3]

When studying the complexity of finding a characteristic formula for some process p with respect to the behavioural relations we have introduced above, we will need some way of measuring the size of the resulting formula as a function of $|p|$. A formula in $\mathcal{L}_X$, where $X \in \{S, CS, RS, TS, 2S, 3S, BS\}$, can be given in *explicit form* as in Definition 3 or by means of a system of equations. In the latter case, we say that the formula is given in *declarative form*. For example, formula $\phi = \langle a\rangle(\langle a\rangle \mathbf{tt} \wedge \langle b\rangle \mathbf{tt}) \wedge \langle b\rangle(\langle a\rangle \mathbf{tt} \wedge \langle b\rangle \mathbf{tt})$ can be represented by the equations $\phi = \langle a\rangle\phi_1 \wedge \langle b\rangle\phi_1$ and $\phi_1 = \langle a\rangle \mathbf{tt} \wedge \langle b\rangle \mathbf{tt}$. We define:

- the *size* of formula φ, denoted by $|\varphi|$, to be the number of symbols that appear in the explicit form of φ,
- the *declaration size* of formula φ, denoted by $\mathrm{decl}(\varphi)$, to be the number of equations that are used in the declarative form of φ, and
- the *equational length* of formula φ, denoted by $\mathrm{eqlen}(\varphi)$, to be the maximum number of symbols that appear in an equation in the declarative form of φ.

For example, for the aforementioned formula ϕ, we have that $|\phi| = 13$, $\mathrm{decl}(\phi) = 2$, and $\mathrm{eqlen}(\phi) = 5$. Note that $\mathrm{decl}(\varphi) \leq |\mathrm{Sub}(\varphi)| \leq |\varphi|$, for each φ.

3 The complexity of deciding characteristic formulae modulo preorders

In this section, we address the complexity of deciding whether formulae in $\mathcal{L}_S$, $\mathcal{L}_{CS}$, $\mathcal{L}_{RS}$, $\mathcal{L}_{TS}$, $\mathcal{L}_{2S}$, and $\mathcal{L}_{3S}$ are characteristic. Since characteristic formulae in those logics are exactly the satisfiable and prime ones [3, Theorem 5], we study the complexity of checking satisfiability and primality separately in Subsections 3.1 and 3.2.

[3] The above definition can also be phrased as follows: A formula $\varphi \in \mathcal{L}_X$ is characteristic for p modulo $\equiv_X$ iff, for all $q \in P$, it holds that $q \models \varphi \Leftrightarrow p \equiv_X q$. This version of the definition is used, in the setting of bisimilarity, in references such as [2, 29].

3.1 The complexity of satisfiability

To address the complexity of the satisfiability problem in $\mathcal{L}_S$, $\mathcal{L}_{CS}$, or $\mathcal{L}_{RS}$, we associate a set $I(\varphi) \subseteq 2^A$ to every formula $\varphi \in \mathcal{L}_{RS}$. Intuitively, $I(\varphi)$ describes all possible sets of initial actions that a process p can have, when $p \models \varphi$.

▶ **Definition 14.** *Let $\varphi \in \mathcal{L}_{RS}$. We define $I(\varphi)$ inductively as follows:*

(a) $I(\mathbf{tt}) = 2^A$,

(b) $I(\mathbf{ff}) = \emptyset$,

(c) $I([a]\mathbf{ff}) = \{X \mid X \subseteq A \text{ and } a \notin X\}$,

(d) $I(\langle a\rangle\varphi) = \begin{cases} \emptyset, & \text{if } I(\varphi) = \emptyset, \\ \{X \mid X \subseteq A \text{ and } a \in X\}, & \text{otherwise} \end{cases}$

(e) $I(\varphi_1 \vee \varphi_2) = I(\varphi_1) \cup I(\varphi_2)$,

(f) $I(\varphi_1 \wedge \varphi_2) = I(\varphi_1) \cap I(\varphi_2)$.

Note that $I(\mathbf{0}) = \{\emptyset\}$.

▶ **Lemma 15.** *For every $\varphi \in \mathcal{L}_{RS}$, the following statements hold:*

(a) *for every $S \subseteq A$, $S \in I(\varphi)$ iff there is a process p such that $I(p) = S$ and $p \models \varphi$.*

(b) *φ is unsatisfiable iff $I(\varphi) = \emptyset$.*

When the number of actions is constant, $I(\varphi)$ can be computed in linear time for every $\varphi \in \mathcal{L}_{RS}$. For $\mathcal{L}_{CS}$, we need even less information; indeed, it is sufficient to define $I(\varphi)$ so that it encodes whether φ is unsatisfiable, or is satisfied only in deadlocked states (that is, states with an empty set of initial actions), or is satisfied only in processes that are not deadlocked, or is satisfied both in some deadlocked and non-deadlocked states. This information can be computed in linear time for every $\varphi \in \mathcal{L}_{CS}$, regardless of the size of the action set.

▶ **Corollary 16.**

(a) *Satisfiability of formulae in $\mathcal{L}_{CS}$ and $\mathcal{L}_S$ is decidable in linear time.*

(b) *Let $|A| = k$, where $k \geq 1$ is a constant. Satisfiability of formulae in $\mathcal{L}_{RS}$ is decidable in linear time.*

On the other hand, if we can use an unbounded number of actions, the duality of $\langle a\rangle$ and $[a]$ can be employed to define a polynomial-time reduction from SAT, the satisfiability problem for propositional logic, to satisfiability in $\mathcal{L}_{RS}$. Moreover, if we are allowed to nest $[a]$ modalities ($a \in A$) and have at least two actions, we can encode n propositional literals using formulae of $\log n$ size and reduce SAT to satisfiability in $\mathcal{L}_{TS}$ in polynomial time. Finally, satisfiability in $\mathcal{L}_{2S}$ is in NP, which can be shown by an appropriate tableau construction.

▶ **Proposition 17.** *Let either $X = RS$ and $|A|$ be unbounded or $X \in \{TS, 2S\}$ and $|A| > 1$. Satisfiability of formulae in $\mathcal{L}_X$ is NP-complete.*

Deciding satisfiability of formulae in $\overline{\mathcal{L}}_{2S}$ when $|A| > 1$, turns out to be PSPACE-complete. (A proof is provided in [1, Appendix B.6].) This means that satisfiability of $\mathcal{L}_{3S}$ is also PSPACE-complete, since $\overline{\mathcal{L}}_{2S} \subseteq \mathcal{L}_{3S}$.

▶ **Proposition 18.** *Let $|A| > 1$. Satisfiability of formulae in $\mathcal{L}_{3S}$ is PSPACE-complete.*

3.2 The complexity of primality

We now study the complexity of checking whether a formula is prime in the logics that characterize some of the relations in Definition 1.

Table 1 Rules for the simulation preorder. If $|_\forall$ is displayed in the conclusion of a rule, then the rule is called universal. Otherwise, it is called existential.

$$\frac{\varphi_1 \vee \varphi_2, \varphi \Rightarrow \psi}{\varphi_1, \varphi \Rightarrow \psi \;|_\forall\; \varphi_2, \varphi \Rightarrow \psi}\ (\mathrm{L}\vee_1) \qquad \frac{\varphi, \varphi_1 \vee \varphi_2 \Rightarrow \psi}{\varphi_1, \varphi \Rightarrow \psi \;|_\forall\; \varphi_2, \varphi \Rightarrow \psi}\ (\mathrm{L}\vee_2)$$

$$\frac{\varphi_1 \wedge \varphi_2, \varphi \Rightarrow \langle a\rangle\psi}{\varphi_1, \varphi \Rightarrow \langle a\rangle\psi \;|_\exists\; \varphi_2, \varphi \Rightarrow \langle a\rangle\psi}\ (\mathrm{L}\wedge_1) \qquad \frac{\varphi, \varphi_1 \wedge \varphi_2 \Rightarrow \langle a\rangle\psi}{\varphi_1, \varphi \Rightarrow \langle a\rangle\psi \;|_\exists\; \varphi_2, \varphi \Rightarrow \langle a\rangle\psi}\ (\mathrm{L}\wedge_2)$$

$$\frac{\varphi_1, \varphi_2 \Rightarrow \psi_1 \wedge \psi_2}{\varphi_1, \varphi_2 \Rightarrow \psi_1 \;|_\forall\; \varphi_1, \varphi_2 \Rightarrow \psi_2}\ (\mathrm{R}\wedge) \qquad \frac{\varphi_1, \varphi_2 \Rightarrow \psi_1 \vee \psi_2}{\varphi_1, \varphi_2 \Rightarrow \psi_1 \;|_\exists\; \varphi_1, \varphi_2 \Rightarrow \psi_2}\ (\mathrm{R}\vee)$$

$$\frac{\langle a\rangle\varphi_1, \langle a\rangle\varphi_2 \Rightarrow \langle a\rangle\psi}{\varphi_1, \varphi_2 \Rightarrow \psi}\ (\diamond) \qquad \frac{\varphi_1, \varphi_2 \Rightarrow \mathbf{tt}}{\textsc{True}}\ (\mathrm{tt})$$

Primality in $\mathcal{L}_S$. Unsatisfiable formulae are trivially prime. Note also that in the case that $|A| = 1$, all satisfiable formulae in $\mathcal{L}_S$ are prime. To address the problem for any action set, for every satisfiable formula $\varphi \in \mathcal{L}_S$ we can efficiently compute a logically equivalent formula φ' given by the grammar $\varphi ::= \mathbf{tt} \mid \langle a\rangle\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi$. We examine the complexity of deciding primality of such formulae.

▶ **Proposition 19.** *Let $\varphi \in \mathcal{L}_S$ such that $\mathbf{ff} \notin \mathrm{Sub}(\varphi)$. Deciding whether φ is prime is in* P.

Proof. We describe algorithm $\mathrm{Prime_S}$ that, on input φ, decides primality of φ. $\mathrm{Prime_S}$ constructs a rooted directed acyclic graph, denoted by G_φ, from the formula φ as follows. Every vertex of the graph is either of the form $\varphi_1, \varphi_2 \Rightarrow \psi$ – where φ_1, φ_2 and ψ are sub-formulae of φ – , or True. The algorithm starts from vertex $x = (\varphi, \varphi \Rightarrow \varphi)$ and applies some rule in Table 1 to x in top-down fashion to generate one or two new vertices that are given at the bottom of the rule. These vertices are the children of x and the vertex x is labelled with either $\exists$ or $\forall$, depending on which one is displayed at the bottom of the applied rule. If x has only one child, $\mathrm{Prime_S}$ labels it with $\exists$. The algorithm recursively continues this procedure on the children of x. If no rule can be applied on a vertex, then this vertex has no outgoing edges. For the sake of clarity and consistency, we assume that right rules, i.e. (R$\vee$) and (R$\wedge$), are applied before the left ones, i.e. (L$\vee_i$) and (L$\wedge_i$), $i = 1, 2$, by the algorithm. The graph generated in this way is an *alternating graph*, as defined by Immerman in [28, Definition 3.24] (see also [1, Appendix A]). In G_φ, the source vertex s is $\varphi, \varphi \Rightarrow \varphi$, and the target vertex t is True. Algorithm $\mathrm{Prime_S}$ solves the problem Reach_a on input G_φ, where Reach_a is Reachability on alternating graphs and is defined in [28, pp. 53–54]. It accepts φ iff Reach_a accepts G_φ. Intuitively, the source vertex $(\varphi, \varphi \Rightarrow \varphi)$ can reach the target vertex True in the alternating graph G_φ exactly when for each pair of disjuncts ψ_1 and ψ_2 in the disjunctive normal form of φ there is a disjunct ψ_3 in the disjunctive normal form of φ that is entailed by both ψ_1 and ψ_2. It turns out that this is a necessary and sufficient condition for the primality of φ. For example, consider the formula $\langle a\rangle\mathbf{tt} \vee \langle b\rangle\mathbf{tt}$. There is no disjunct of $\langle a\rangle\mathbf{tt} \vee \langle b\rangle\mathbf{tt}$ that is entailed by both $\langle a\rangle\mathbf{tt}$ and $\langle b\rangle\mathbf{tt}$. This is because that formula is not prime, as we observed in Example 8. On the other hand, the formula $\langle a\rangle\mathbf{tt} \vee \langle a\rangle\langle b\rangle\mathbf{tt}$ is prime since each of its disjuncts entails $\langle a\rangle\mathbf{tt}$. The full technical details are included in [1, Appendix C.1]. Note that graph G_φ is of polynomial size and there is a linear-time algorithm solving Reach_a [28]. ◀

Primality in $\mathcal{L}_{CS}$. Note that, in the case of $\mathcal{L}_{CS}$, the rules in Table 1 do not work any more because, unlike $\mathcal{L}_S$, the logic $\mathcal{L}_{CS}$ can express some "negative information" about the behaviour of processes. For example, let $A = \{a\}$ and $\varphi = \langle a\rangle\mathbf{tt}$. Then, Prime_S accepts φ, even though φ is not prime in $\mathcal{L}_{CS}$. Indeed, $\varphi \models \langle a\rangle\langle a\rangle\mathbf{tt} \vee \langle a\rangle\mathbf{0}$, but $\varphi \not\models \langle a\rangle\langle a\rangle\mathbf{tt}$ and $\varphi \not\models \langle a\rangle\mathbf{0}$. However, we can overcome this problem as described in the proof sketch of Proposition 20 below.

▶ **Proposition 20.** *Let $\varphi \in \mathcal{L}_{CS}$ be a formula such that every $\psi \in \text{Sub}(\varphi)$ is satisfiable. Deciding whether φ is prime is in* P.

Proof. Consider the algorithm that first computes the formula $\varphi^\diamond$ by applying rule $\langle a\rangle\mathbf{tt} \to_\diamond \mathbf{tt}$, and rules $\mathbf{tt} \vee \psi \to_{\text{tt}} \mathbf{tt}$ and $\mathbf{tt} \wedge \psi \to_{\text{tt}} \psi$ modulo commutativity on φ. It holds that φ is prime iff $\varphi^\diamond$ is prime and $\varphi^\diamond \models \varphi$. Next, the algorithm decides primality of $\varphi^\diamond$ by solving reachability on a graph constructed as in the case of simulation using the rules in Table 1, where rule (tt) is replaced by rule (0), whose premise is $\mathbf{0}, \mathbf{0} \Rightarrow \mathbf{0}$ and whose conclusion is True. To verify $\varphi^\diamond \models \varphi$, the algorithm computes a process p for which $\varphi^\diamond$ is characteristic within $\mathcal{L}_{CS}$ and checks whether $p \models \varphi$. In fact, the algorithm has also a preprocessing phase during which it applies a set of rules on φ and obtains an equivalent formula with several desirable properties. See [1, Appendix C.2] for full details. ◀

Primality in $\mathcal{L}_{RS}$. The presence of formulae of the form $[a]\mathbf{ff}$ in $\mathcal{L}_{RS}$ means that a prime formula $\varphi \in \mathcal{L}_{RS}$ has at least to describe which actions are necessary or forbidden for any process that satisfies φ. For example, let $A = \{a, b\}$. Then, $\langle a\rangle\mathbf{0}$ is not prime, since $\langle a\rangle\mathbf{0} \models (\langle a\rangle\mathbf{0} \wedge [b]\mathbf{ff}) \vee (\langle a\rangle\mathbf{0} \wedge \langle b\rangle\mathbf{tt})$, and $\langle a\rangle\mathbf{0}$ entails neither $\langle a\rangle\mathbf{0} \wedge [b]\mathbf{ff}$ nor $\langle a\rangle\mathbf{0} \wedge \langle b\rangle\mathbf{tt}$. Intuitively, we call a formula φ *saturated* if φ describes exactly which actions label the outgoing edges of any process p such that $p \models \varphi$. Formally, φ is saturated iff $I(\varphi)$ is a singleton.

If the action set is bounded by a constant, given φ, we can efficiently construct a formula φ^s such that (1) φ^s is saturated and for every $\langle a\rangle\varphi' \in \text{Sub}(\varphi^s)$, φ' is saturated, (2) φ is prime iff φ^s is prime and $\varphi^s \models \varphi$, and (3) primality of φ^s can be efficiently reduced to $\text{Reach}_a(G_{\varphi^s})$.

▶ **Proposition 21.** *Let $|A| = k$, where $k \geq 1$ is a constant, and $\varphi \in \mathcal{L}_{RS}$ be such that if $\psi \in \text{Sub}(\varphi)$ is unsatisfiable, then $\psi = \mathbf{ff}$ and ψ occurs in the scope of some $[a]$. Deciding whether φ is prime is in* P.

As the following result indicates, primality checking for formulae in $\mathcal{L}_{RS}$ becomes computationally hard when $|A|$ is not a constant.

▶ **Proposition 22.** *Let $|A|$ be unbounded. Deciding primality of formulae in $\mathcal{L}_{RS}$ is* coNP*-complete.*

Proof. We give a polynomial-time reduction from Sat to deciding whether a formula in $\mathcal{L}_{RS}$ is not prime. Let φ be a propositional formula over $x_0, \ldots, x_{n-1}$. We set $\varphi' = (\varphi \wedge \neg x_n) \vee (x_n \wedge \bigwedge_{i=1}^{n-1} \neg x_i)$ and φ'' to be φ' where x_i is substituted with $\langle a_i\rangle\mathbf{0}$ and $\neg x_i$ with $[a_i]\mathbf{ff}$, where $A = \{a_0, \ldots, a_n\}$. As φ'' is satisfied in $a_n.\mathtt{0}$, it is a satisfiable formula, and so φ'' is prime in $\mathcal{L}_{RS}$ iff φ'' is characteristic within $\mathcal{L}_{RS}$. We show that φ is satisfiable iff φ'' is not characteristic within $\mathcal{L}_{RS}$. Let φ be satisfiable and let s denote a satisfying assignment of φ. Consider $p_1, p_2 \in P$ such that:

- $p_1 \xrightarrow{a_i} \mathtt{0}$ iff $s(x_i) = \text{true}$, for $0 \leq i \leq n-1$, and $p_1 \overset{a_n}{\not\to}$, and
- $p_2 \xrightarrow{a_n} \mathtt{0}$ and $p_2 \overset{a}{\not\to}$ for every $a \in A \setminus \{a_n\}$.

It holds that $p_i \models \varphi''$, $i = 1, 2$, $p_1 \not\leq_{RS} p_2$, and $p_2 \not\leq_{RS} p_1$. Suppose that there is a process q, such that φ'' is characteristic for q within $\mathcal{L}_{RS}$. If $q \xrightarrow{a_n}$, then $q \not\leq_{RS} p_1$. On the other hand, if $q \overset{a_n}{\not\to}$, then $q \not\leq_{RS} p_2$. So, both cases lead to a contradiction, which means that φ'' is not characteristic within $\mathcal{L}_{RS}$. For the converse implication, assume that φ is unsatisfiable. This implies that there is no process satisfying the first disjunct of φ''. Thus, φ'' is characteristic for p_2, described above, within $\mathcal{L}_{RS}$.

Proving the matching upper bound is non-trivial. There is a coNP algorithm that uses properties of prime formulae and rules of Table 1, carefully adjusted to the case of ready simulation. We describe the algorithm and prove its correctness in [1, Appendix C.3.2]. ◀

Primality in $\mathcal{L}_{TS}$. If we have more than one action, a propositional literal can be encoded by using the restricted nesting of modal operators that is allowed by the grammar for $\mathcal{L}_{TS}$. This observation is the crux of the proof of the following result.

▶ **Proposition 23.** *Let $|A| > 1$. Deciding primality of formulae in $\mathcal{L}_{TS}$ is* coNP*-hard.*

Proof. Let $A = \{0, 1\}$. The proof follows the steps of the proof of Proposition 22. The initial and basic idea is that given an instance φ of SAT over $x_1, \ldots, x_n$, every x_i is substituted with $[\overline{b_{i1}}]\mathbf{ff} \wedge \langle b_{i1}\rangle([\overline{b_{i2}}]\mathbf{ff} \wedge \langle b_{i2}\rangle(\ldots([\overline{b_{ik}}]\mathbf{ff} \wedge \langle b_{ik}\rangle\mathbf{0})\ldots))$ and $\neg x_i$ with $[b_{i1}][b_{i2}]\ldots[b_{ik}]\mathbf{ff}$, where $b_{i1}\ldots b_{ik}$ is the binary representation of i and $\overline{b} = 0$, if $b = 1$, and $\overline{b} = 1$, if $b = 0$. For more technical details, see [1, Appendix C.4.1]. ◀

In contrast to the case for $\mathcal{L}_{RS}$, bounding the size of the action set is not sufficient for deciding primality of formulae in $\mathcal{L}_{TS}$ in polynomial time. However, we show that both satisfiability and primality become efficiently solvable if we bound both $|A|$ and the modal depth of the input formula.

▶ **Proposition 24.** *Let $|A| = k$ and $\varphi \in \mathcal{L}_{TS}$ with $\mathrm{md}(\varphi) = d$, where $k, d \geq 1$ are constants. Then, there is an algorithm that decides whether φ is satisfiable and prime in linear time.*

Proof. It is necessary and sufficient to check that there is a process p with $\mathrm{depth}(p) \leq d$ such that (1) $p \models \varphi$ and (2) for every q with $\mathrm{depth}(q) \leq d + 1$, if $q \models \varphi$ then $p \lesssim_{TS} q$. Since k and d are considered to be constants, there is an algorithm that does so and requires linear time in $|\varphi|$. In particular, the algorithm runs in $\mathcal{O}(2^{2k^{d+1}} \cdot k^{d+1} \cdot |\varphi|)$. ◀

To classify the problem of deciding whether formulae in $\mathcal{L}_{TS}$ are characteristic when $|A|$ is bounded, let us briefly introduce fixed-parameter tractable problems – see, for instance, [19, 21] for textbook accounts of this topic. Let $L \subseteq \Sigma^* \times \Sigma^*$ be a parameterized problem. We denote by L_y the associated fixed-parameter problem $L_y = \{x \mid (x, y) \in L\}$, where y is the parameter. Then, $L \in$ FPT (or L is fixed-parameter tractable) if there are a constant α and an algorithm to determine if (x, y) is in L in time $f(|y|) \cdot |x|^\alpha$, where f is a computable function [18].

▶ **Corollary 25.** *Let $|A| = k$, where $k \geq 1$ is a constant. The problems of deciding whether formulae in $\mathcal{L}_{TS}$ are satisfiable, prime, and characteristic are in* FPT*, with the modal depth of the input formula as the parameter.*

We note that the coNP-hardness argument from Proposition 23 applies also to logics that include $\mathcal{L}_{TS}$. Since $\mathcal{L}_{TS} \subseteq \mathcal{L}_{2S}$, the coNP-hardness of deciding primality of formulae in $\mathcal{L}_{TS}$ with $|A| > 1$ implies the same lower bound for deciding primality of formulae in $\mathcal{L}_{2S}$ when $|A| > 1$. Next, we show that in $\mathcal{L}_{3S}$ with $|A| > 1$ the problem becomes PSPACE-hard.

Primality in $\mathcal{L}_{3S}$. Let $|A| > 1$. PSPACE-hardness of $\mathcal{L}_{3S}$-satisfiability implies PSPACE-hardness of $\overline{\mathcal{L}}_{3S}$-validity. Along the lines of the proof of [2, Theorem 26], we prove the following result.

▶ **Proposition 26.** *Let $|A| > 1$. Deciding prime formulae within $\mathcal{L}_{3S}$ is* PSPACE*-hard.*

▶ Remark 27. Note that primality within $\mathcal{L}_{BS}$ coincides with primality modulo $\sim$. In [2], primality modulo $\sim$ is called completeness and it is shown to be decidable in PSPACE. However, the algorithm used in [2] does not immediately imply that primality within $\mathcal{L}_{3S}$ is in PSPACE.

Interestingly, PSPACE-hardness of $\mathcal{L}_{2S}$-validity implies the following theorem.

▶ **Theorem 28.** *Let $X \in \{CS, RS, TS, 2S, 3S\}$ and $|A| > 1$. The problem of deciding whether a formula in $\overline{\mathcal{L}}_{2S}$ is prime in $\mathcal{L}_X$ is* PSPACE*-hard.*

Proof. We reduce $\mathcal{L}_{2S}$-validity to this problem. Let $\varphi \in \mathcal{L}_{2S}$. The reduction will return a formula φ', such that φ is $\mathcal{L}_{2S}$-valid if and only if φ' is prime in $\mathcal{L}_X$. If $\mathtt{0} \not\models \varphi$, then let $\varphi' = \mathbf{tt}$; in this case, φ is not valid and $\mathbf{tt}$ is not prime in $\mathcal{L}_X$. Otherwise, let $\varphi' = \mathbf{0} \vee \neg\varphi$. If φ is valid, then $\varphi' \equiv \mathbf{0}$ and therefore φ' is prime in $\mathcal{L}_X$. On the other hand, if φ is not valid, then there is some process $p \models \neg\varphi$. From $\mathtt{0} \models \varphi$, it holds that $p \xrightarrow{a}$. Then, $\varphi' \models \mathbf{0} \vee \bigvee_{a\in A}\langle a\rangle\mathbf{tt}$, but $\varphi' \not\models \mathbf{0}$ and $\varphi' \not\models \bigvee_{a\in A}\langle a\rangle\mathbf{tt}$. Since $\mathbf{0} \vee \bigvee_{a\in A}\langle a\rangle\mathbf{tt} \in \mathcal{L}_{CS}$, φ' is not prime in $\mathcal{L}_X$, where $X \in \{CS, RS, TS, 2S, 3S\}$. ◀

Theorem 28 shows that when deciding primality in $\mathcal{L}_X$, if we allow the input to be in a logic $\mathcal{L}$ that is more expressive than $\mathcal{L}_X$, the computational complexity of the problem can increase. It is then reasonable to constrain the input of the problem to be in $\mathcal{L}_X$ in order to obtain tractable problems as in the case of $\mathcal{L}_S$ and $\mathcal{L}_{CS}$.

Before we give our main result summarizing the complexity of deciding characteristic formulae, we introduce two classes that play an important role in pinpointing the complexity of deciding characteristic formulae within $\mathcal{L}_{RS}$, $\mathcal{L}_{TS}$, and $\mathcal{L}_{2S}$. The first class is $\mathsf{DP} = \{L_1 \cap L_2 \mid L_1 \in \mathsf{NP} \text{ and } L_2 \in \mathsf{coNP}\}$ [38] and the second one is US [9], which is defined thus: A language $L \in \mathsf{US}$ iff there is a non-deterministic Turing machine T such that, for every instance x of L, $x \in L$ iff T has exactly one accepting path. The problem UniqueSat, viz. the problem of deciding whether a given Boolean formula has exactly one satisfying truth assignment, is US-complete and $\mathsf{US} \subseteq \mathsf{DP}$ [9].

▶ **Theorem 29.**

(a) *Deciding characteristic formulae within $\mathcal{L}_S$, $\mathcal{L}_{CS}$, or $\mathcal{L}_{RS}$ with a bounded action set is in* P*.*

(b) *Deciding characteristic formulae within $\mathcal{L}_{RS}$ with an unbounded action set is* US*-hard and belongs to* DP*.*

(c) *Deciding characteristic formulae within $\mathcal{L}_{TS}$ or $\mathcal{L}_{2S}$ is* US*-hard.*

(d) *Deciding characteristic formulae within $\mathcal{L}_{3S}$ is* PSPACE*-hard.*

4 Finding characteristic formulae: The gap between trace simulation and the other preorders

Let $X \in \{S, CS\}$ or $X = RS$ and $|A|$ is bounded by a constant. The complexity of finding characteristic formulae within $\mathcal{L}_X$ depends on the representation of the output. If the characteristic formula has to be given in explicit form, then the following result holds.

Table 2 The complexity of deciding satisfiability and primality, and of finding characteristic formulae for different logics. Finding$_{decl}$ (resp. Finding$_{expl}$) denotes the problem of finding the characteristic formula for a given finite loop-free process, when the output is given in declarative (resp. explicit) form. Superscripts $= k$, $> k$, and > 1 mean that the action set is bounded by a constant, unbounded, and has more than one action, respectively. FP is the class of functions computable in polynomial time. All the results shown in white cells have been proven in this paper, whereas results in light gray are from [2].

	$\mathcal{L}_S$	$\mathcal{L}_{CS}$	$\mathcal{L}^{=k}_{RS}$	$\mathcal{L}^{>k}_{RS}$	$\mathcal{L}^{>1}_{TS}$	$\mathcal{L}^{>1}_{2S}$	$\mathcal{L}^{>1}_{3S}$	$\mathcal{L}_{BS}$
Satisfiability	P	P	P	NP-comp.	NP-comp.	NP-comp.	PSPACE-comp.	PSPACE-comp.
Primality	P	P	P	coNP-comp.	coNP-hard	coNP-hard	PSPACE-hard	PSPACE-comp.
Finding$_{decl}$	FP	FP	FP	FP	NP-hard	FP	FP	FP
Finding$_{expl}$	NP-hard							

▶ **Proposition 30.** *Let $X \in \{S, CS\}$ or $X = RS$ and $|A|$ is bounded by a constant. If finding the characteristic formula within $\mathcal{L}_X$ for a given finite loop-free process can be done in polynomial time when the output is given in explicit form, then* P = NP.

Proof. If the assumption of the proposition is true, the results of this paper allow us to decide trace equivalence of two finite loop-free processes in polynomial time. (For details, the reader can see [1, Appendix E.1].) Since trace equivalence for such processes is coNP-complete [27, Theorem 2.7(1)], this implies that P = NP. ◀

However, if output formulae are given in declarative form, then finding characteristic formulae within $\mathcal{L}_X$, where $X \in \{nS, CS, RS, BS\}$, $n \geq 1$, can be done in polynomial time.

▶ **Proposition 31.** *For every $X \in \{nS, CS, RS, BS\}$, where $n \geq 1$, there is a polynomial-time algorithm that, given a finite loop-free process p, outputs a formula in declarative form that is characteristic for p within $\mathcal{L}_X$.*

Proof. The proof relies on inductive definitions of characteristic formulae within $\mathcal{L}_X$, where $X \in \{S, CS, RS, 2S, BS\}$, given in [29, 6], and within $\mathcal{L}_{nS}$, $n \geq 3$, given in [1, Appendix E.1]. These definitions guarantee that there are polynomial-time recursive procedures which construct characteristic formulae within $\mathcal{L}_X$. We prove the proposition for $X = 2S$ below.

Given a finite loop-free process p, the characteristic formula for p within $\mathcal{L}_{2S}$ is defined as follows: $\chi_{2S}(p) = \bar{\chi}_S(p) \wedge \bigwedge_{a \in A} \bigwedge_{p \xrightarrow{a} p'} \langle a \rangle \chi_{2S}(p')$, where $\bar{\chi}_S(p) = \bigwedge_{a \in A} [a] \bigvee_{p \xrightarrow{a} p'} \bar{\chi}_S(p')$. Consider the algorithm that recursively constructs $\chi_{2S}(p)$. The algorithm has to construct $\chi_{2S}(p')$ and $\bar{\chi}_S(p')$ for every $p' \in \text{reach}(p)$, yielding a linear number of equations. Moreover, for every $p' \in \text{reach}(p)$, $\bar{\chi}_S(p')$ is of linear size in $|p'|$. If $p' = \mathtt{0}$, then $\bar{\chi}_S(p') = \bigwedge_{a \in A} [a]\mathbf{ff}$. Otherwise, $|\bar{\chi}_S(p')| = \mathcal{O}(|\{p'' \mid p' \xrightarrow{a} p''\}| + |A|)$, where $|A|$ is added because for every $a \in A$ such that $p' \overset{a}{\not\rightarrow}$, $[a]\mathbf{ff}$ is a conjunct of $\bar{\chi}_S(p')$. Note that for every p'', if $\bar{\chi}_S(p'')$ occurs in $\bar{\chi}_S(p')$, it is considered to add 1 to the size of $\bar{\chi}_S(p')$. Therefore, $|\bar{\chi}_S(p')|$ is of linear size in $|p'|$. Using a similar argument, we can show that $\chi_{2S}(p')$ is of linear size. Thus, the algorithm constructs a linear number of equations, each of which is of linear size in $|p|$. The proofs for $X \in \{nS, CS, RS, BS\}$, $n \neq 2$, are analogous. ◀

▶ Remark 32. Note that the recursive procedures given in [29, 6] and [1, Appendix E.1] provide characteristic formulae for finite processes with loops provided that we enrich the syntax of our logics by adding greatest fixed points. See, for example, [6]. Consequently, constructing characteristic formulae for finite processes within $\mathcal{L}_X$, $X \in \{nS, CS, RS, BS\}$, $n \geq 1$, can be done in polynomial time.

We now present the complexity gap between finding characteristic formulae for preorders CS, RS, BS, and nS, $n \geq 1$, and the same search problem for preorder TS. In the former case, there are characteristic formulae with both declaration size and equational length that are polynomial in the size of the processes they characterize, and they can be efficiently computed. On the contrary, for TS, even if characteristic formulae are always of polynomial declaration size and polynomial equational length, they *cannot* be efficiently computed unless $\mathsf{P} = \mathsf{NP}$.

▶ **Proposition 33.** *Assume that for every finite loop-free process p, there is a characteristic formula within $\mathcal{L}_{TS}$ for p, denoted by $\chi(p)$, such that both* $\mathrm{decl}(\chi(p))$ *and* $\mathrm{eqlen}(\chi(p))$ *are in $\mathcal{O}(|p|^k)$ for some $k \in \mathbb{N}$. Given a finite loop-free process p, if $\chi(p)$ can be computed in polynomial time, then* $\mathsf{P} = \mathsf{NP}$.

Next, we prove that we do not expect that a finite loop-free process p has always a short characteristic formula within $\mathcal{L}_{TS}$ when this is combined with a second condition. To show that statement, we need the following lemma.

▶ **Lemma 34.** *For every finite p and q,* $\mathrm{traces}(p) = \mathrm{traces}(q)$ *iff $p \lesssim_{TS} p+q$ and $q \lesssim_{TS} p+q$.*

Proof. If $\mathrm{traces}(p) = \mathrm{traces}(q)$, then $p \lesssim_{TS} p + q$. Indeed, for every $p \xrightarrow{a} p'$, it holds that $p + q \xrightarrow{a} p'$ and, trivially, $p' \lesssim_{TS} p'$. Moreover, $\mathrm{traces}(p + q) = \mathrm{traces}(p) \cup \mathrm{traces}(q) = \mathrm{traces}(p)$. Symmetrically, $q \lesssim_{TS} p + q$. Conversely, if $p \lesssim_{TS} p + q$ and $q \lesssim_{TS} p + q$, then $\mathrm{traces}(p + q) = \mathrm{traces}(p) = \mathrm{traces}(q)$, and we are done. ◀

▶ **Proposition 35.** *Assume that the following two conditions hold:*

1. *For every finite loop-free process p, there is a characteristic formula within $\mathcal{L}_{TS}$ for p, denoted by $\chi(p)$, such that both* $\mathrm{decl}(\chi(p))$ *and* $\mathrm{eqlen}(\chi(p))$ *are in $\mathcal{O}(|p|^k)$ for some $k \in \mathbb{N}$.*
2. *Given a finite loop-free process p and a formula φ in declarative form, deciding whether φ is characteristic for p within $\mathcal{L}_{TS}$ is in* NP.

Then $\mathsf{NP} = \mathsf{coNP}$.

Proof. We describe an NP algorithm $\mathcal{A}$ that decides non-membership in Sat and makes use of conditions 1 and 2 of the proposition. Let ϕ be an input CNF formula to Sat. Algorithm $\mathcal{A}$ computes the DNF formula $\neg\phi$ for which it needs to decide DNF-Tautology. Then, $\mathcal{A}$ reduces DNF-Tautology to deciding trace equivalence of processes p_0 and q constructed as described in the proof of [27, Theorem 2.7(1)]. $\mathcal{A}$ can decide if $\mathrm{traces}(p_0) = \mathrm{traces}(q)$ by checking $p_0 \lesssim_{TS} p_0 + q$ and $q \lesssim_{TS} p_0 + q$ because of Lemma 34. Finally, $\mathcal{A}$ reduces $p_0 \lesssim_{TS} p_0+q$ (resp. $q \lesssim_{TS} p_0+q$) to model checking: it needs to check whether $p_0+q \models \chi(p_0)$ (resp. $p_0 + q \models \chi(q)$). To this end, $\mathcal{A}$ guesses two formulae φ_{p_0} and φ_q in declarative form of polynomial declaration size and equational length, and two witnesses that verify that φ_{p_0} and φ_q are characteristic within $\mathcal{L}_{TS}$ for p_0 and q, respectively. This can be done due to conditions 1 and 2. $\mathcal{A}$ rejects the input iff both $p_0+q \models \chi(p_0)$ and $p_0+q \models \chi(q)$ are true. ◀

5 A note on deciding characteristic formulae modulo equivalence relations

So far, we have studied the complexity of algorithmic problems related to characteristic formulae in the modal logics that characterize the simulation-based preorders in van Glabbeek's spectrum. As shown in [3], those logics are powerful enough to describe characteristic formulae for each finite, loop-free process up to the preorder they characterize. It is therefore natural to wonder whether they can also express characteristic formulae modulo the kernels of those preorders. The following result indicates that the logics $\mathcal{L}_X$, where $X \in \{S, CS, RS\}$, have very weak expressive power when it comes to defining characteristic formulae modulo $\equiv_X$.

▶ **Proposition 36.** *No formula in $\mathcal{L}_S$ is characteristic for some process p with respect to $\equiv_S$. For $X \in \{CS, RS\}$, a formula φ is characteristic for some process p with respect to $\equiv_X$ iff it is logically equivalent to $\bigwedge_{a\in A}[a]\mathbf{ff}$.*

Proof. Assume, towards contradiction, that there is a formula φ_c^S in $\mathcal{L}_S$ that is characteristic for some process p with respect to $\equiv_S$. Let ℓ be the depth of p and $a \in A$. Define process $q = p + a^{\ell+1}\mathsf{0}$ – that is, q is a copy of p with an additional path that has exactly $\ell + 1$ a-transitions. It is easy to see that $p \lesssim_S q$, but $q \not\lesssim_S p$. Since $p \models \varphi_c^S$, it holds that $q \models \varphi_c^S$. However, $q \not\equiv_S p$, which contradicts our assumption that φ_c^S is characteristic for p with respect to $\equiv_S$. For $X \in \{CS, RS\}$, note that a formula φ is logically equivalent to $\bigwedge_{a\in A}[a]\mathbf{ff}$ iff it is satisfied only by processes without outgoing transitions, and so it is characteristic for any such process modulo $\equiv_X$. To prove that no formula is characteristic for some process p with positive depth modulo $\equiv_{CS}$ or $\equiv_{RS}$, a similar argument to the one for $\equiv_S$ can be used. For $\equiv_{RS}$, the action a should be chosen such that $p \xrightarrow{a} p'$ for some p'. ◀

For TS and $2S$, there are non-trivial characteristic formulae modulo $\equiv_{TS}$ and $\equiv_{2S}$, respectively. For example, if $A = \{a, b\}$, the formula $\varphi_a = \langle a\rangle([a]\mathbf{ff} \wedge [b]\mathbf{ff}) \wedge [b]\mathbf{ff} \wedge [a][a]\mathbf{ff} \wedge [a][b]\mathbf{ff}$ is satisfied only by processes that are equivalent, modulo those equivalences, to process $p_a = a.\mathsf{0}$ that has a single transition labelled with a. Thus, φ_a is characteristic for p_a modulo both $\equiv_{TS}$ and $\equiv_{2S}$. We can use the following theorem as a tool to prove hardness of deciding characteristic formulae modulo some equivalence relation. Theorem 37 below is an extension of [2, Theorem 26], so that it holds for every X such that a characteristic formula modulo $\equiv_X$ exists, namely $X \in \{CS, RS, TS, 2S, 3S, BS\}$.

▶ **Theorem 37.** *Let $X \in \{CS, RS, TS, 2S, 3S, BS\}$. Validity in $\overline{\mathcal{L}}_X$ reduces in polynomial time to deciding characteristic formulae with respect to $\equiv_X$.*

Note that, from the results of Subsection 3.1, validity in $\overline{\mathcal{L}}_{RS}$ with an unbounded action set, $\overline{\mathcal{L}}_{TS}$ with $|A| > 1$, and $\overline{\mathcal{L}}_{2S}$ with $|A| > 1$ is coNP-complete, whereas validity in $\overline{\mathcal{L}}_{3S}$ with $|A| > 1$ is PSPACE-complete. Consequently, from Theorem 37, deciding whether a formula is characteristic modulo $\equiv_{RS}$ with an unbounded action set, $\equiv_{TS}$ with $|A| > 1$, and $\equiv_{2S}$ with $|A| > 1$ is coNP-hard. That problem is PSPACE-hard modulo $\equiv_{3S}$ with $|A| > 1$.

6 Conclusions

In this paper, we studied the complexity of determining whether a formula is characteristic for some finite, loop-free process in each of the logics providing modal characterizations of the simulation-based semantics in van Glabbeek's branching-time spectrum [22]. Since, as shown in [3], characteristic formulae in each of those logics are exactly the satisfiable and prime ones, we gave complexity results for the satisfiability and primality problems,

and investigated the boundary between logics for which those problems can be solved in polynomial time and those for which they become computationally hard. Our results show that computational hardness already manifests itself in ready simulation semantics [10, 34] when the size of the action set is not a constant. Indeed, in that setting, the mere addition of formulae of the form $[a]\mathbf{ff}$ to the logic that characterizes the simulation preorder yields a logic whose satisfiability and primality problems are NP-hard and coNP-hard respectively. Moreover, we show that deciding primality in the logic characterizing 3-nested simulation is PSPACE-hard in the presence of at least two actions.

Amongst others, we also studied the complexity of constructing characteristic formulae in each of the logics we consider, both when such formulae are presented in explicit form and in declarative form. In particular, one of our results identifies a sharp difference between trace simulation and the other semantics when it comes to constructing characteristic formulae. For all the semantics apart from trace simulation, there are characteristic formulae that have declaration size and equational length that are polynomial in the size of the processes they characterize and they can be efficiently computed. On the contrary, for trace simulation, even if characteristic formulae are always of polynomial declaration size and polynomial equational length, they *cannot* be efficiently computed, unless $\mathsf{P} = \mathsf{NP}$.

Our results are summarized in Table 2 and open several avenues for future research that we are currently pursuing. First of all, the precise complexity of primality checking is still open for the logics characterizing the n-nested simulation semantics. We conjecture that checking primality in $\mathcal{L}_{2S}$ is coNP-complete and that PSPACE-completeness holds for n-nested simulation when $n \geq 3$. Next, we plan to study the complexity of deciding whether formulae are characteristic in the extensions of the modal logics we have considered in this article with greatest fixed points. Indeed, in those extended languages, one can define characteristic formulae for finite processes. It is known that deciding whether a formula is characteristic is PSPACE-complete for **HML**, but becomes EXP-complete for its extension with fixed-point operators – see reference [2]. It would be interesting to see whether similar results hold for the other logics. Finally, building on the work presented in [3], we plan to study the complexity of the algorithmic questions considered in this article for (some of) the linear-time semantics in van Glabbeek's spectrum.

References

1 Luca Aceto, Antonis Achilleos, Aggeliki Chalki, and Anna Ingólfsdóttir. The complexity of deciding characteristic formulae in van glabbeek's branching-time spectrum. *CoRR*, abs/2405.13697, 2024. doi:10.48550/arXiv.2405.13697.

2 Luca Aceto, Antonis Achilleos, Adrian Francalanza, and Anna Ingólfsdóttir. The complexity of identifying characteristic formulae. *J. Log. Algebraic Methods Program.*, 112:100529, 2020. doi:10.1016/j.jlamp.2020.100529.

3 Luca Aceto, Dario Della Monica, Ignacio Fábregas, and Anna Ingólfsdóttir. When are prime formulae characteristic? *Theor. Comput. Sci.*, 777:3–31, 2019. doi:10.1016/j.tcs.2018.12.004.

4 Luca Aceto, Ignacio Fábregas, David de Frutos-Escrig, Anna Ingólfsdóttir, and Miguel Palomino. Graphical representation of covariant-contravariant modal formulae. In Bas Luttik and Frank Valencia, editors, *Proceedings 18th International Workshop on Expressiveness in Concurrency, EXPRESS 2011, Aachen, Germany, 5th September 2011*, volume 64 of *EPTCS*, pages 1–15, 2011. doi:10.4204/EPTCS.64.1.

5 Luca Aceto, Anna Ingólfsdóttir, Kim Guldstrand Larsen, and Jiri Srba. *Reactive Systems: Modelling, Specification and Verification*. Cambridge University Press, USA, 2007.

6 Luca Aceto, Anna Ingólfsdóttir, Paul Blain Levy, and Joshua Sack. Characteristic formulae for fixed-point semantics: a general framework. *Math. Struct. Comput. Sci.*, 22(2):125–173, 2012. doi:10.1017/S0960129511000375.

7 Antonis Achilleos. The completeness problem for modal logic. In *Proc. of Logical Foundations of Computer Science - International Symposium, LFCS 2018*, volume 10703 of *Lecture Notes in Computer Science*, pages 1–21. Springer, 2018. doi:10.1007/978-3-319-72056-2_1.

8 Benjamin Bisping, David N. Jansen, and Uwe Nestmann. Deciding all behavioral equivalences at once: A game for linear-time-branching-time spectroscopy. *Log. Methods Comput. Sci.*, 18(3), 2022. doi:10.46298/lmcs-18(3:19)2022.

9 Andreas Blass and Yuri Gurevich. On the unique satisfiability problem. *Inf. Control.*, 55(1-3):80–88, 1982. doi:10.1016/S0019-9958(82)90439-9.

10 Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995. doi:10.1145/200836.200876.

11 Gérard Boudol and Kim Guldstrand Larsen. Graphical versus logical specifications. *Theor. Comput. Sci.*, 106(1):3–20, 1992. doi:10.1016/0304-3975(92)90276-L.

12 Michael C. Browne, Edmund M. Clarke, and Orna Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Theor. Comput. Sci.*, 59:115–131, 1988. doi:10.1016/0304-3975(88)90098-9.

13 Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

14 Rance Cleaveland. On automatically explaining bisimulation inequivalence. In Edmund M. Clarke and Robert P. Kurshan, editors, *Computer Aided Verification, 2nd International Workshop, CAV '90*, volume 531 of *Lecture Notes in Computer Science*, pages 364–372. Springer, 1990. doi:10.1007/BFb0023750.

15 Rance Cleaveland and Bernhard Steffen. Computing behavioural relations, logically. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *Automata, Languages and Programming, 18th International Colloquium, ICALP91, Madrid, Spain, July 8-12, 1991, Proceedings*, volume 510 of *Lecture Notes in Computer Science*, pages 127–138. Springer, 1991. doi:10.1007/3-540-54233-7_129.

16 David de Frutos-Escrig, Carlos Gregorio-Rodríguez, Miguel Palomino, and David Romero-Hernández. Unifying the linear time-branching time spectrum of process semantics. *Log. Methods Comput. Sci.*, 9(2), 2013. doi:10.2168/LMCS-9(2:11)2013.

17 Rocco De Nicola and Frits W. Vaandrager. Three logics for branching bisimulation. *J. ACM*, 42(2):458–487, 1995. doi:10.1145/201019.201032.

18 Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *SIAM J. Comput.*, 24(4):873–921, 1995. doi:10.1137/S0097539792228228.

19 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

20 Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Mathematical logic (2. ed.)*. Undergraduate texts in mathematics. Springer, 1994.

21 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

22 Rob J. van Glabbeek. The linear time - branching time spectrum I. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, pages 3–99. North-Holland / Elsevier, 2001. doi:10.1016/b978-044482830-9/50019-9.

23 Susanne Graf and Joseph Sifakis. A modal characterization of observational congruence on finite terms of CCS. *Inf. Control.*, 68(1-3):125–145, 1986. doi:10.1016/S0019-9958(86)80031-6.

24 Jan Friso Groote and Frits W. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Inf. Comput.*, 100(2):202–260, 1992. doi:10.1016/0890-5401(92)90013-6.

25 Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985. doi:10.1145/2455.2460.

26 Sören Holmström. A refinement calculus for specifications in Hennessy-Milner logic with recursion. *Formal Aspects Comput.*, 1(3):242–272, 1989. doi:10.1007/BF01887208.

27 Harry B. Hunt III, Daniel J. Rosenkrantz, and Thomas G. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *J. Comput. Syst. Sci.*, 12(2):222–268, 1976. doi:10.1016/S0022-0000(76)80038-4.

28 Neil Immerman. *Descriptive Complexity*. Springer, 1999. doi:10.1007/978-1-4612-0539-5.

29 Anna Ingolfsdottir, Jens Christian Godskesen, and Michael Zeeberg. Fra Hennessy-Milner logik til CCS-processer. Master's thesis, Aalborg University, 1987. In Danish.

30 Sandra Kiefer, Pascal Schweitzer, and Erkal Selman. Graphs identified by logics with counting. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald Sannella, editors, *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part I*, volume 9234 of *Lecture Notes in Computer Science*, pages 319–330. Springer, 2015. doi:10.1007/978-3-662-48057-1_25.

31 Dexter Kozen. Results on the propositional μ-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983. doi:10.1016/0304-3975(82)90125-6.

32 Richard E. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM J. Comput.*, 6(3):467–480, 1977. doi:10.1137/0206033.

33 Kim Guldstrand Larsen. Proof systems for satisfiability in Hennessy-Milner logic with recursion. *Theor. Comput. Sci.*, 72(2&3):265–288, 1990. doi:10.1016/0304-3975(90)90038-J.

34 Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991. doi:10.1016/0890-5401(91)90030-6.

35 Jan Martens and Jan Friso Groote. Computing minimal distinguishing Hennessy-Milner formulas is NP-hard, but variants are tractable. In Guillermo A. Pérez and Jean-François Raskin, editors, *34th International Conference on Concurrency Theory, CONCUR 2023*, volume 279 of *LIPIcs*, pages 32:1–32:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CONCUR.2023.32.

36 Robin Milner. An algebraic definition of simulation between programs. In D. C. Cooper, editor, *Proceedings of the 2nd International Joint Conference on Artificial Intelligence, IJCAI 1971*, pages 481–489. William Kaufmann, 1971. URL: http://ijcai.org/Proceedings/71/Papers/044.pdf.

37 Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.

38 Christos H. Papadimitriou and Mihalis Yannakakis. The complexity of facets (and some facets of complexity). *J. Comput. Syst. Sci.*, 28(2):244–259, 1984. doi:10.1016/0022-0000(84)90068-0.

39 Bernhard Steffen and Anna Ingólfsdóttir. Characteristic formulae for processes with divergence. *Inf. Comput.*, 110(1):149–163, 1994. doi:10.1006/inco.1994.1028.

40 Wolfgang Thomas. On the Ehrenfeucht-Fraïssé game in theoretical computer science. In Marie-Claude Gaudel and Jean-Pierre Jouannaud, editors, *TAPSOFT'93: Theory and Practice of Software Development, International Joint Conference CAAP/FASE*, volume 668 of *Lecture Notes in Computer Science*, pages 559–568. Springer, 1993. doi:10.1007/3-540-56610-4_89.

A Complete Diagrammatic Calculus for Automata Simulation

Thibaut Antoine
ENS Rennes, France

Robin Piedeleu
University College London, UK

Alexandra Silva
Cornell University, Ithaca, NY, USA

Fabio Zanasi
University College London, UK

Abstract

We give a sound and complete (in)equational theory for simulation of finite state automata. Our approach uses a string diagrammatic calculus to represent automata and a functorial semantics to capture simulation in a compositional way. Interestingly, in contrast to other approaches based on regular expressions, fixpoints are a derived notion in our calculus and the resulting axiomatisation is finitary.

2012 ACM Subject Classification Theory of computation

Keywords and phrases finite-state automata, simulation, string diagrams, axiomatisation

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.27

Funding Fabio Zanasi acknowledges support from EPSRC EP/V002376/1, MIUR P2022HXNSC (PRIN 2022 PNRR - Next Generation EU), and ARIA Safeguarded AI TA1.1 grant n.8777242. The other authors acknowledge support from ERC grant Autoprobe (no. 101002697), ARIA Safeguarded AI program, and EPSRC Standard Grant CLeVer (EP/S028641/1).

Acknowledgements Fabio Zanasi conducted part of this research while affiliated with the University of Bologna, Italy.

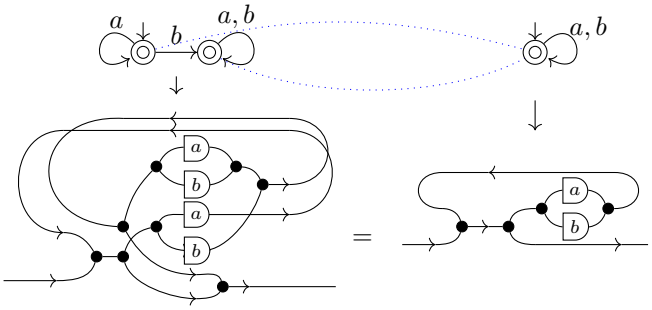
1 Introduction

Non-deterministic automata are basic models of computation which play a central role in formal verification – for instance, the Kripke frames used in the semantics of modal and temporal logics are non-deterministic automata. In verification, one often tries to answer a question of the form: does state s of the model satisfy φ? This question can be rephrased in terms of trace containment – do all the valid traces starting from s satisfy φ? To effectively answer it, formulae can be compiled to models which are then compared to the original model, using *simulation* for trace containment [2] (or, if one requires a finer semantics, bisimulation).

Another approach to reason about model behaviour is algebraic: one can design a language into which both the models and the formulae can be compiled, and reason equationally about their relationship in this common language. A very successful example of this approach is Kleene algebra and extensions thereof [13, 15, 23, 10]. In the algebraic approach, it is important that one has sufficient axioms to reason about the equivalence or refinement of behaviours. This amounts to providing a *sound and complete* axiomatization of the intermediary language with respect to model behaviour. Famously, Kozen was the first to provide a sound and complete algebraic axiomatization of language-equivalence of regular expressions [14]. The main challenge to overcome was the handling of loops (or Kleene star), a challenge which reappears in extensions of Kleene algebra and in other process calculi.

This paper builds on recent work connecting language theory and string diagrams [20] to offer a novel perspective on non-deterministic finite-state automata (NFA) and simulation. Remarkably, [20] provides a *finitary* axiomatization of NFA up to language equivalence. The key point is that automata are represented in a more modular syntax of string diagrams, in which loops/fixpoints are a derived notion. Moreover, the proposed diagrammatic language is more expressive, exploiting the underlying lattice structure of languages. This allows the authors to *(i)* encode not only the NFA themselves, but relations between the states of NFA, as well as *(ii)* an equational proof that these relations satisfy the defining properties of simulations [20]. The payoff is that it becomes possible to reason purely equationally (or inequationally, since we allow inequalities between diagrams) about fixpoints, a feature that is provably impossible to achieve in a traditional syntax [22].

Below, we give an example of a (bi)simulation (in blue) between two NFA, with their diagrammatic representation below. These correspond to $a^*(b(a+b)^*+1)$ and $(a+b)^*$; proving that they are (bi)similar in a traditional syntax requires fixpoint axioms. Our axiomatisation on the other hand will allow us to derive this fact using purely (in)equational reasoning on the diagrams themselves (Example 38).



Contributions. Our main contribution is a complete axiomatisation of NFA simulation. Contrary to language equivalence, the algebraic study of simulation has not been explored in depth in previous work, with the notable exception of Frendrup and Jensen's work on CCS expressions modulo similarity [9]. Given the clear importance of simulation for program refinement and bisimulation, it is worthwhile to develop different techniques to study it. Here, we offer a novel perspective on simulation through a string diagrammatic calculus with the same syntax as the one in [20], but a new semantics, capturing similarity instead of language equivalence. Importantly, the added expressiveness of our calculus allows the complete axiomatisation to be *finite*. Moreover, on the road to completeness, we prove several results about the algebraic structure of automata up to simulation, in particular that they form a lattice (Section 2.2, Theorem 5), and characterise fixed-points of systems of linear inequalities (Theorem 8).

Outline. After introducing some preliminary material in Section 2, we recall the diagrammatic calculus in which we encode NFA, together with a functorial (that is, compositional) semantics expressed in terms of simulation, rather than language-equivalence (Section 3). In Section 4 we provide constructions to translate between NFA and string diagrams. The main technical result appears in Section 5: we prove that the calculus is sound and complete to reason about simulation of NFA. We discuss related work and future research in Section 6, including a discussion of how our approach could extend to NFA modulo bisimilarity.

2 Background: automata and simulation

In this section we recall the necessary background regarding automata, simulation, as well as some basic properties and algebraic operations first introduced by Milner [18]. Then, we characterise the solutions of certain systems of equations (or rather, inequations, in this case), a result which will be instrumental in showing the soundness of our diagrammatic representation of NFA in later sections.

In what follows, we fix some finite alphabet Σ, over which all NFA will be defined and write $\mathcal{P}_f(\Sigma)$ for the set of finite subsets of Σ.

2.1 Automata and their algebraic operations

We recall here the notion of automaton with which we will work throughout.

▶ **Definition 1** (Nondeterministic Finite Automaton). *A nondeterministic finite automaton (NFA) A over Σ is a quadruple (Q, q_0, F, α) where Q is a finite set of states, $q_0 \in Q$ is the initial state of A, $F \subseteq Q$ is a set of final states (also seen as a function $Q \to \{0,1\}$), and $\alpha \subseteq Q \times \Sigma \times Q$ is the transition relation. We will denote $q \xrightarrow{a}_A q'$ for $(q, a, q') \in \alpha$, and will omit the sub-scripted A in case it is clear in context. Note that we do not allow ϵ-transitions.*

A path in A is a sequence $(q_{i_0}, \dots, q_{i_k}) \in Q^{k+1}$ where $q_{i_j} \to_A q_{i_{j+1}}$, denoted $q_{i_0} \rightsquigarrow_A q_{i_k}$.

We will use the following standard operations on NFA extensively: sum, (synchronous) product and composition, including prefixing. Rigorous definitions can be found in Appendix A. Let $A = (Q, q_0, F, \alpha)$ and $B = (S, s_0, G, \beta)$ be two NFA.

Sum. $A + B$ is defined by taking $Q \cup S \cup \{t_0\}$ where t_0 is a new state which can transition to the set of states reachable from q_0 and s_0.

Product. $A \times B$ is defined by taking the $Q \times S$ as set of states and allowing transitions $(q, s) \xrightarrow{a}_{A\times B} (q', s')$ iff $q \xrightarrow{a}_A q'$ and $s \xrightarrow{a}_B s'$.

Composition. $A.B$ is defined by attaching a copy of B to every final state in A and keeping as final states those of B.

Prefix. $a.A$ is the composition of $(\{q_0, q_1\}, q_0, \{q_1\}, \{(q_0, a, q_1)\})$ and A. We will also write $S.A$ for $S \in \mathcal{P}_f(\Sigma)$ as a shorthand for $\left(\sum_{a\in S} a\right).A = \sum_{a\in S} a.A$.

Finally, we define three special NFA to which we will often refer:

$$\bot = (\{q_0\}, q_0, 0, \varnothing) \qquad 1 = (\{q_0\}, q_0, 1, \varnothing) \qquad \top = (\{q_0\}, q_0, 1, \{(q_0, a, q_0) \mid a \in \Sigma\})$$

2.2 The simulation lattice

We now define (strong) *simulation* of a NFA by another, the central concept of this paper.

▶ **Definition 2** (Simulation Relation, Simulation preorder). *Let $A = (Q, q_0, F, \alpha), B = (S, s_0, G, \beta)$ be two NFAs. A simulation relation from A to B is a binary relation $R \subseteq Q \times S$ such that (1) $(q_0, s_0) \in R$; (2) if $(q, s) \in R$, then $F(q) \leq G(s)$; (3) if $(q, s) \in R$ and $q \xrightarrow{a}_A q'$, then there exists a state $s' \in S$ such that $s \xrightarrow{a}_B s'$ and $(q', s') \in R$.*

Note that condition (3) can also be phrased in terms of relational composition as $R^{-1}; \to_A \subseteq \to_B; R^{-1}$. In this case, we say that A is simulated by B (or that B simulates A) and we write $A \preceq B$, or $A \preceq_R B$ to specify the simulation relation.

It immediately follows that $\preceq$ is a preorder on NFAs: for all A, B, C NFAs we have $A \preceq_{\mathrm{id}} A$, as well as $A \preceq_R B$ and $B \preceq_{R'} C$ implies $A \preceq_{R;R'} C$.

▶ **Definition 3** (Similarity partial order). *If A and B are two NFA such that $A \preceq B$ and $B \preceq A$, we say that they are* similar *and write $A \simeq B$.*

We call Ω the set of all NFA modulo similarity. Given two equivalence classes $X = [A]$, $Y = [B]$, we write $X \leq Y$ if $A \preceq B$.

It is a standard fact about preorders that $\leq$ forms a partial order over Ω. Moreover, all previously defined operations are monotone with respect to it.

▶ **Proposition 4** (Monotony of composition). *Let $A = (Q, q_0, F, \alpha)$ and $B = (S, s_0, G, \beta)$ be two NFAs and let $A' = (Q', q'_0, F', \alpha')$, $B' = (S', s'_0, G', \beta')$ such that $A \preceq A'$, $B \preceq B'$. Then we have $A.B \preceq A'.B'$. In particular, prefixing is monotone with respect to $\preceq$.*

Proof. Assume in particular that $A \preceq_{R_A} A'$, $B \preceq_{R_B} B'$. Let $R = R_A \cup \{((i,s),(j,s')) \mid (s,s') \in R_B, (q_i, q'_j) \in R_A, F(q_i) = 1\}$. We will show that $A.B \preceq_R A'.B'$.

- $R_A \subseteq R$, thus $(q_0, q'_0) \in R$.
- Let $(t,t') \in R$, and $t \xrightarrow{a}_{A.B} u$. The interesting case happens when $t = q_i$ with $F(q_i) = 1$ and $u = (i,s)$ with $s_0 \xrightarrow{a}_B s$. Since $B \preceq_{R_B} B'$, $(s_0, s'_0) \in R_B$ and there is a $s' \in S'$ such that $s'_0 \xrightarrow{a}_B s'$ and $(s,s') \in R_B$.
  Moreover, since $t = q_i \in Q$ and $(t,t') \in R$, $t' = q'_j \in Q'$; and we also have $F(q_i) = 1 \leq F'(q'_j)$ thus q'_j is final in A'. By definition of $A'.B'$ this means that $q'_j \xrightarrow{a}_{A'.B'} (j,s')$, and by definition of R we also have $((i,s),(j,s')) \in R$, which concludes the proof. ◀

Proposition 4 allows us to lift prefixing to equivalence classes of NFA modulo similarity as a monotone operation: let $a.[A] = [a.A]$ for $[A] \in \Omega$.

The second key result of this subsection is the existence of a lattice structure on Ω. We first state the necessary properties for NFA before lifting the sum and product to Ω.

▶ **Theorem 5** (Bounded lattice operations). *Let A, B be two NFAs. We have:*

1. $\bot \preceq A \preceq \top$
2. $A \times B \preceq A, B$ *and if* $C \preceq A, B$ *then* $C \preceq A \times B$ *(that is, product of NFAs acts as a meet)*
3. $A, B \preceq A + B$ *and if* $A, B \preceq C$ *then* $A + B \preceq C$ *(that is, sum of NFAs acts as a join)*

Proof. We give the simulation relation for each:

1. The full relation suffices for each inequality.
2. $A \times B \preceq A$ through the projection, *i.e.*, the relation $\{((q,s),q) \mid q \in Q\}$. The same goes for B symmetrically. Moreover if $C \preceq_{R_A} A$ and $C \preceq_{R_B} B$ then $C \preceq_R A \times B$ with $R = \{(t,(q,s)) \mid (t,q) \in R_A \text{ and } (t,s) \in R_B\}$.
3. The simulation relations for $A, B \preceq A + B$ are given by the injections. If $A \preceq_{R_A} C$ and $B \preceq_{R_B} C$, then $A + B \preceq_R C$ with $R = \{(p_0, t_0)\} \cup \{(q,t) \mid (q,t) \in R_A\} \cup \{(s,t) \mid (s,t) \in R_B\}$, if p_0 and t_0 are the initial state of $A + B$ and C respectively. ◀

Direct consequences of this theorem are the monotony of $+$ and $\times$, their commutativity and unitality (with $\bot$ and $\top$ respectively). The monotony of $+$ and $\times$ also allows us to lift the sum and product to the set of NFA modulo similarity: for $[A], [B] \in \Omega$, let $[A] + [B] = [A + B]$ and $[A] \times [B] = [A \times B]$; finally we let $\bot$ and $\top$ denote $[\bot]$ and $[\top]$ respectively.

▶ **Corollary 6.** *Ω is a lattice, with $+$ as join, $\times$ as meet, $\bot$ as bottom, and $\top$ as top.*

In what follows, we will often use NFA to denote their equivalence class modulo similarity.

2.3 Systems of linear inequalities

In this section we introduce systems of linear inequalities and characterise their (least) solutions. This is not only an original result of independent interest, but one we will use later to show the soundness of the representation of NFA in our diagrammatic calculus.

▶ **Definition 7** (System of linear inequations). *Let $n \in \mathbb{N}$ and $K_0, \dots, K_n \in \Omega$, and for all $0 \leq i, j \leq n$, let $d_{i,j} \in \mathcal{P}_f(\Sigma)$. These define a system of $n+1$ inequations:*

$$(E) : \left\{ K_i + \sum_{j=0}^{n} d_{i,j}.X_j \leq X_i \right\}_{0 \leq i \leq n}$$

where $X_0, \dots, X_n$ are variables taking value in Ω. Using matrix multiplication and vector notations, we will write (E) as $\mathbf{K} + \mathbf{DX} \leq \mathbf{X}$.

The following theorem is the key result of this section: given a system of inequalities, we construct the (equivalence class of) NFA that is its unique smallest solution. Frendrup and Jensen prove a similar result [9, Theorem 7] for their syntax directly, but our methods are different and of independent interest.

▶ **Theorem 8.** *Let $\mathbf{D}$ be a matrix of coefficients in $\mathcal{P}_f(\Sigma)$.*

1. *For every vector $\mathbf{K}$, the system $\mathbf{K} + \mathbf{DX} \leq \mathbf{X}$ has a unique smallest solution $S(\mathbf{K})$.*
2. *Let $\mathbf{F} \in \{0,1\}^{n+1}$, K be a NFA and $A = (Q, q_0, \mathbf{F}, \alpha)$ where $Q = \{q_i\}_{0 \leq i \leq n}$, and $q_i \xrightarrow{a} q_j$ iff $a \in d_{i,j}$. Let $\mathbf{A} = (A_0, \dots, A_n)$ where $A_i = (Q, q_i, \mathbf{F}, \alpha)$. Then $S(\mathbf{F}.K) = \mathbf{A}.K$.*

Proof.

1. By monotony of prefixing and summing, $\mathbf{X} \mapsto \mathbf{K} + \mathbf{DX}$ is monotonous; by the Knaster-Tarski theorem it has a unique least fixed point, which is the solution we are looking for.
2. One can see easily that $\mathbf{A}.K$ is a solution. We show that it is the smallest. For that sake, let $\mathbf{X} = (X_0, \dots, X_n)$ be a solution of the system. In all that follows we write $x_0^{(i)}$ for the initial state of X_i. For each $0 \leq i \leq n$, let $Y_i = F_i.K + \sum_{j=0}^{n} d_{i,j}.X_j$, s_i be its initial state, and R_i be the simulation relation from Y_i to X_i. We now fix a i, and build a simulation relation S from $A_i.K$ to X_i.
   First, for all $0 \leq j \leq n$ we define P_j as follows:

$$P_j = \bigcup_{q_{i_0} \to_{A_i} \cdots \to_{A_i} q_{i_k}} R_{i_{k-1}}; \cdots; R_{i_0}$$

   where $i_0 = i$ and $i_k = j$. It is a simulation relation by union and composition. Then let

$$S' = \bigcup_{q_j \text{ s.t. } q_i \rightsquigarrow_{A_i} q_j} \{(q_j, x) \mid (x_0^{(j)}, x) \in P_j\}$$

$$S'' = \bigcup_{\substack{q_j \text{ s.t. } q_i \rightsquigarrow_{A_i} q_j \\ F(q_j)=1}} \{((j,t), x) \mid \exists x' \in X_j : (t, x') \in R_j \text{ and } (x', x) \in P_j\}$$

   and $S = \{(q_i, x_0^{(i)})\} \cup S' \cup S''$.
   Note that S'' is well defined since if q_j is final, then $K \preceq_{R_j} X_j$. We show that S is a simulation relation.

- By definition, $(q_i, x_0^{(i)}) \in S$
- Let $(q_j, x) \in S$, and $q_j \xrightarrow{a} q_k$. By definition, this means $(q_j, x) \in S'$, i.e. $(x_0^{(j)}, x) \in P_j$. By definition of R_j, we have $(s_j, x_0^{(j)}) \in R_j$. Combined to the existence of a path $q_i \rightsquigarrow q_j \to q_k$, this implies $(s_j, x) \in P_k$. Moreover, we know by construction of A that $s_j \xrightarrow{a} x_0^{(k)}$ since $q_j \xrightarrow{a} q_k$. Therefore, P_k being a simulation relation, there is a y such that $x \xrightarrow{a} y$ and $(x_0^{(k)}, y) \in P_k$ i.e. $(q_k, y) \in S'$.
- Let q_j be an accessible final state in A and $(q_j, x) \in S$ with $q_j \xrightarrow{a} (j, t)$, meaning $t_0 \xrightarrow{a}_K t$. Moreover since q_j is final, $K \preceq_{R_j} X_j$ and thus $(t_0, x_0^{(j)}) \in R_j$. By simulation there is a $x' \in X_j$ such that $x_0^{(j)} \xrightarrow{a} x'$ and $(t, x') \in R_j$. Moreover $(q_j, x) \in S$ thus $(x_0^{(j)}, x) \in P_j$. By simulation, there is a y with $(x', y) \in P_j$ such that $x \xrightarrow{a} y$, which concludes this case.
- The proof in the case $(j, t) \xrightarrow{a} (j, t')$ is direct by applying the simulation property. ◀

3 Syntax and semantics

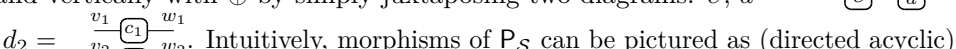
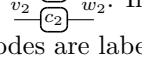
In this section, we define a diagrammatic calculus in which we can encode NFA in a natural way. The syntax has appeared in previous work [20], but the semantics is new, reflecting the focus of this work on simulation, rather than language-equivalence. We will then equip the same syntax with an (in)equational theory which we will show in Section 5 fully axiomatises the intended semantics. We proceed in three steps:

- In Section 3.1, we define our syntax as a free coloured *prop* on a number of generators, using string diagrams to depict its morphisms. For an introduction to string diagrammatic syntax, we refer the reader to [21].
- In Section 3.2, we formalise the intended semantics as a *symmetric monoidal functor* into the symmetric monoidal category (SMC) of monotone relations with the Cartesian product.
- Finally, we equip the syntax with a partial order which is sound for the intended semantics. This order is given by a finite number of (in)equalities of the same type.

3.1 Syntax

We build on a line of research that has sought to give a formal treatment of graphical models of computation of varying expressive power within the unifying language of symmetric monoidal categories. More specifically, we rely on the notion of coloured product and permutations category (prop), a mathematical structure which generalises standard multisorted algebraic theories [5]. Formally, a *prop* is a strict symmetric monoidal category (SMC) whose objects are words of a finite alphabet and where the monoidal product $\oplus$ on objects is given by concatenation. Equivalently, it is a strict SMC whose objects are all monoidal products of a finite number of generating objects.

Our syntax is a prop $\mathsf{P}_{\mathcal{S}}$, freely generated from a *signature* $\mathcal{S} = (G, M)$: a pair of a finite set of objects G and a set M of morphisms $g : v \to w$, where v and w are words over G. There are two ways of describing the morphisms of the prop $\mathsf{P}_{\mathcal{S}}$ concretely. As terms of (G^*, G^*)-sorted syntax whose constants are elements of M and whose operations are the usual categorical composition $(-);(-) : \mathsf{P}_{\mathcal{S}}(u, v) \times \mathsf{P}_{\mathcal{S}}(v, w) \to \mathsf{P}_{\mathcal{S}}(u, w)$ and the monoidal product $(-) \oplus (-) : \mathsf{P}_{\mathcal{S}}(v_1, w_1) \times \mathsf{P}_{\mathcal{S}}(v_2, w_2) \to \mathsf{P}_{\mathcal{S}}(v_1 v_2, w_1 w_2)$, quotiented by the laws of SMCs. However, this quotient is cumbersome and unintuitive to work with.

This is why we prefer a different representation: with their two forms of composition, monoidal categories admit a natural two-dimensional notation of *string diagrams*. The idea is that a morphism $c : v \to w$ of P_S is better represented as a box with $|v|$ ordered wires labelled by the elements of v on the left and $|w|$ wires labelled by the elements of w on the right. We can compose these diagrams in two different ways: horizontally with ; by connecting the right wires of the first diagram to the left wires of the second (when the types match), and vertically with $\oplus$ by simply juxtaposing two diagrams: $c ; d =$  and $d_1 \oplus d_2 =$  . Intuitively, morphisms of P_S can be pictured as (directed acyclic) graphs whose nodes are labelled by elements of M. The symmetry $\sigma_{a,b} : ab \to ba$ is drawn as a wire crossing  which swaps the a-and b-wires, and the unit for $\oplus$, $id_0 : 0 \to 0$, as the empty diagram  (we use 0 to denote the empty word). With this representation the laws of SMCs become diagrammatic tautologies.

In this work, we start with the same diagrammatic syntax as [20], which we call Syn_Σ. It is the free two-coloured prop freely generated by the objects and morphisms below.

- Two generating objects, ▶ (right) and ◀ (left), whose identities we will depict respectively as  and .
- The following generating morphisms:

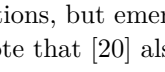
(1)

(2)

Morphisms of Syn_Σ are thus vertical and horizontal compositions of these generators, potentially including wire crossings. The direction of the arrows on the generators' wires denotes their type: for example,  represents an operation of type ▶→▶▶, while  has type ◀▶→ ε (where ε denotes the empty list, *i.e.*, the unit for the monoidal product).

As in [20], we will use the generators (1) to represent NFA:  , , allow us to encode states by gathering incoming and outgoing transitions, while the transitions themselves are encoded with  . The remaining two generators,  and , allow us to form feedback loops, with which we can encode iteration, as we will see in more details in Section 4. The , , play another important role: they will allow us to construct simulation relations directly in our syntax.

The white generators  , , in (2) are not used to build automata-diagrams, but will allow us to show that the diagrammatic simulation relations we construct do satisfy the required properties, and thereby allow us to prove that one automaton simulates another using purely (in)equational reasoning. As we will see next, these are not artificial syntactic operations, but emerge naturally out of the chosen semantics.

Note that [20] also contained dual generators  , which are not needed here.

3.2 Semantics

In this section, we explain how to interpret the diagrams of the previous section as relations. We will formalise the intended semantics as a SMC, and the interpretation as a symmetric monoidal functor from the syntax to the semantics.

Contrary to [20], the relations in our target semantics are not between languages, but between elements of Ω, that is, equivalence classes of NFAs *up to similarity*. As in [20], each generator is interpreted as a *monotone* relation between posets.

▶ **Definition 9** (Monotone relation). *Given two posets $(X, \leq_X)$ and $(Y, \leq_Y)$, a relation $R : X \to Y$ is* monotone *whenever for $(x, y) \in R$, if $x' \leq_X x$, $y \leq_Y y'$ then $(x', y') \in R$.*

▶ **Proposition 10** (SMC of monotone relations). *Posets and monotone relations form a SMC* $\mathsf{Prof}_{\mathbb{B}}$ *with composition given by relational composition, where the identity for an object* $(X, \leq_X)$ *is the order relation* $\leq_X$ *itself, and with monoidal product the product of posets.*

Moreover, monotone relations of the same type can be ordered by inclusion, making $\mathsf{Prof}_{\mathbb{B}}$ *into an order-enriched SMC.*

The SMC $\mathsf{Prof}_{\mathbb{B}}$ of monotone relations has also appeared in the literature under the name of Boolean(-enriched) [8, 20] and relational [11] profunctors, or weakening relations [19].

Since Syn_Σ is freely-generated, to define a symmetric monoidal functor $[\![\cdot]\!] : \mathsf{Syn}_\Sigma \to \mathsf{Prof}_{\mathbb{B}}$, it is sufficient to specify the image of each generating object and morphism.

▶ **Definition 11** (Semantics). *Let* $[\![\cdot]\!]$ *be the following mapping.*

- *For the generating objects, let* $[\![\blacktriangleright]\!] = (\Omega, \geq)$ *and* $[\![\blacktriangleleft]\!] = (\Omega, \leq)$*. By Definition 9, this fixes the interpretation of the corresponding identities to be the order relations themselves:*

$$[\![\longrightarrow\!\!\!\!\rightarrow\!\!\!\!\longrightarrow]\!] = \{(X, Y) \mid Y \leq X\} \qquad [\![\longrightarrow\!\!\!\!\leftarrow\!\!\!\!\longrightarrow]\!] = \{(X, Y) \mid X \leq Y\}$$

- *We map generating morphisms to the following relations:*

$$\left[\!\!\left[\rightarrow\!\bullet\!\begin{smallmatrix}\rightarrow\\\rightarrow\end{smallmatrix}\right]\!\!\right] = \left\{\big(X, (Y_1, Y_2)\big) \mid Y_i \leq X,\, i = 1, 2\right\} \qquad [\![\rightarrow\!\bullet]\!] = \{(X, \bullet) \mid X \in \Omega\}$$

$$\left[\!\!\left[\begin{smallmatrix}\rightarrow\\\rightarrow\end{smallmatrix}\!\bullet\!\rightarrow\right]\!\!\right] = \left\{\big((X_1, X_2), Y\big) \mid Y \leq X_i,\, i = 1, 2\right\} \qquad [\![\bullet\!\rightarrow\!\!\!\!\longrightarrow]\!] = \{(\bullet, Y) \mid Y \in \Omega\}$$

$$\left[\!\!\left[\begin{smallmatrix}\rightarrow\\\leftarrow\end{smallmatrix}\right]\!\!\right] = \{(\bullet, (Y_1, Y_2)) \mid Y_2 \leq Y_1\} \qquad \left[\!\!\left[\begin{smallmatrix}\leftarrow\\\rightarrow\end{smallmatrix}\right]\!\!\right] = \{((X_1, X_2), \bullet) \mid X_1 \leq X_2\}$$

$$[\![\longrightarrow\!\boxed{a}\!\longrightarrow]\!] = \{(X, Y) \mid a.Y \leq X\} \qquad (a \in \Sigma)$$

$$\left[\!\!\left[\rightarrow\!\circ\!\begin{smallmatrix}\rightarrow\\\rightarrow\end{smallmatrix}\right]\!\!\right] = \left\{\big(X, (Y_1, Y_2)\big) \mid Y_1 \times Y_2 \leq X, \right\} \qquad [\![\rightarrow\!\circ]\!] = \{(\top, \bullet)\}$$

A few remarks about the semantics are in order.

- The order relation is built into the identity wires $\longrightarrow\!\!\!\!\rightarrow\!\!\!\!\longrightarrow$ and $\longrightarrow\!\!\!\!\leftarrow\!\!\!\!\longrightarrow$. The two directions represent the identities on Ω ordered by $\geq := \{(X, Y) : Y \leq X\}$ and $\leq$ respectively. This is the opposite of the convention in [20], and is imposed by the use of prefixes (as opposed to suffixes in [20]) to interpret the atomic actions $\longrightarrow\!\boxed{a}\!\longrightarrow$, *cf.* last item of this list.
- The black primitives are standard monotone relations associated with any poset. One can see them as copy, delete, and their duals, relative to the relevant partial order relation. Similarly, the wire-bending primitives are interpreted as relations that exist for any poset, making use of the fact that our semantic category is compact-closed [12].
- The white generators are interpreted as the meet and top of the lattice structure obtained in Corollary 6. Contrary to [20], we do not need generators for the join of the lattice – this is one of the distinguishing features of language equivalence and simulation.
- The action of $\longrightarrow\!\boxed{a}\!\longrightarrow$ for each letter $a \in \Sigma$, relates $a.Y$ to any NFA X simulating it.

▶ **Proposition 12.** $[\![\cdot]\!]$ *extends to a symmetric monoidal functor* $\mathsf{Syn}_\Sigma \to \mathsf{Prof}_{\mathbb{B}}$*.*

Proof. Since Syn_Σ is freely generated, we just need to check that each generator is a *monotone* relation. The only non-immediate case is the monotony of $[\![\longrightarrow\!\boxed{a}\!\longrightarrow]\!]$, which follows from Proposition 4. ◀

This means that we can compute the semantics of arbitrary diagrams in a fully compositional way, using the following rules for composition and monoidal product:

$$[\![c\,;\, d]\!] = \left[\!\!\left[\overset{u}{\longrightarrow}\boxed{c}\overset{v}{\longrightarrow}\boxed{d}\overset{w}{\longrightarrow}\right]\!\!\right] = \left\{(\mathbf{X}, \mathbf{Z}) \mid \exists \mathbf{Y}\, (\mathbf{X}, \mathbf{Y}) \in [\![\longrightarrow\!\boxed{c}\!\longrightarrow]\!]\,, (\mathbf{Y}, \mathbf{Z}) \in \left[\!\!\left[\longrightarrow\!\boxed{d}\!\longrightarrow\right]\!\!\right]\right\}$$

$$[\![c_1 \oplus c_2]\!] = \left[\!\!\left[\begin{smallmatrix}v_1\\v_2\end{smallmatrix}\!\!\begin{smallmatrix}\boxed{c_1}\\\boxed{c_2}\end{smallmatrix}\!\!\begin{smallmatrix}w_1\\w_2\end{smallmatrix}\right]\!\!\right] = \left\{\big((\mathbf{X}_1, \mathbf{X}_2), (\mathbf{Y}_1, \mathbf{Y}_2)\big) \mid (\mathbf{X}_i, \mathbf{Y}_i) \in [\![\longrightarrow\!\boxed{c_i}\!\longrightarrow]\!]\,,\, i = 1, 2\right\}$$

3.3 Inequational theory

In Figure 1 below we introduce MDA, the theory of *Milner Diagram Algebra*, a set of axioms which will prove complete for simulation of NFA (once we have shown how to encode them into the diagrammatic syntax). Some background on symmetric monoidal (inequality) theories can be found in Appendix B.

Unsurprisingly MDA is close to the existing diagrammatic axiomatisation of NFA *up to language equivalence* of [20, Section 3] and we use the same naming scheme to highlight the similarities and differences between the two. There are two main differences: the lack of  , and  and (E1-E2) which only hold *laxly* here, witnessing the simulation $a.b+a.c \preceq a.(b+c)$. This is akin to how regular expressions up to (bi)similarity do not satisfy left-distributivity of composition over the sum, but still satisfy right-distributivity (E3-E4).

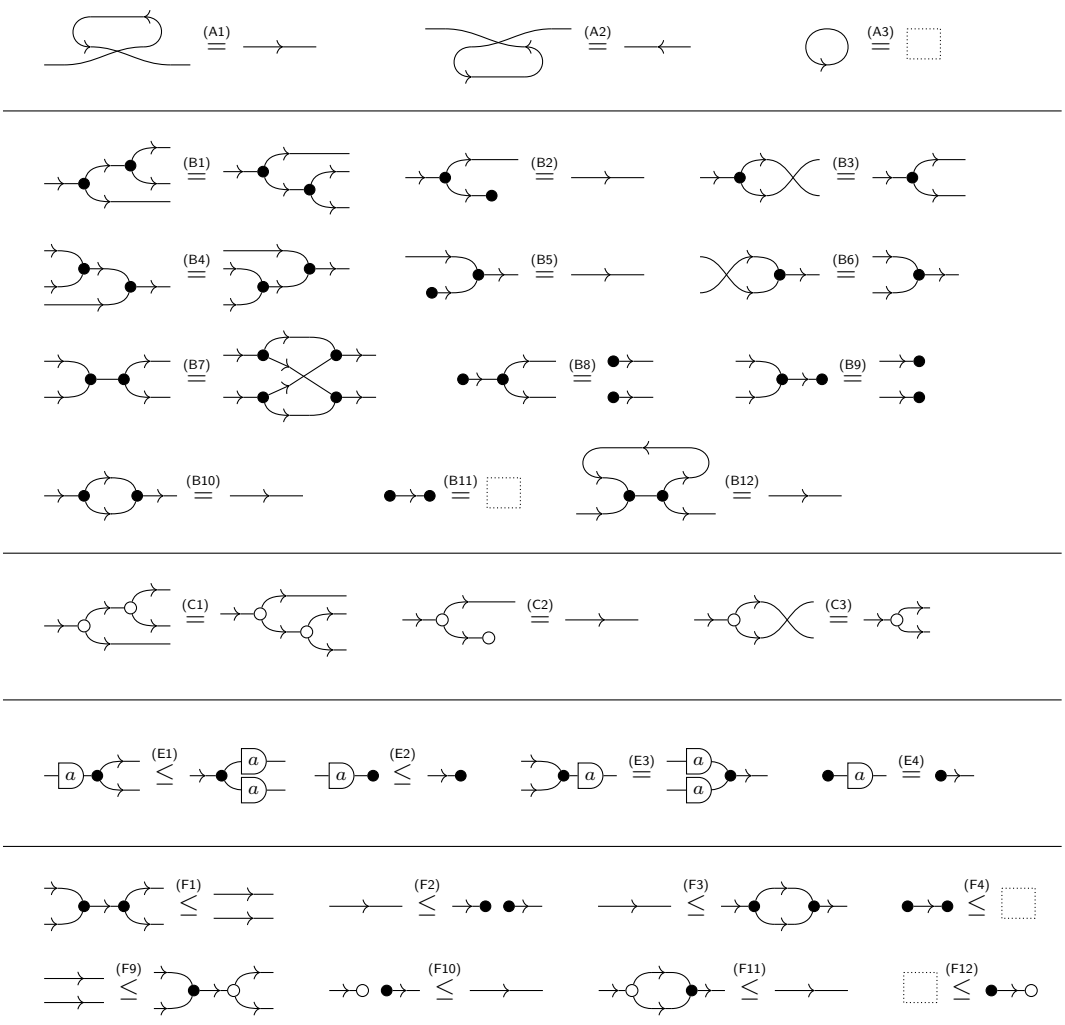
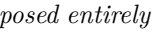


Figure 1 Theory of Milner Diagram Algebra (MDA).

▶ Remark 13. Note that this theory is not minimal: for example, the lax distributivity axiom (E1) can be proven using (F1), (B10) and (E3). A similar property holds for (E2). However, we have preferred to keep them since they highlight the main difference with previous work on language equivalence/inclusion [20].

As we will explain more thoroughly in Section 4, we are interested in the properties of diagrams that are closely related to NFA. We identify these in the following definition.

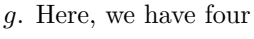
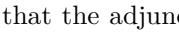
▶ **Definition 14** (Automaton-diagram). *We call* automaton-diagram *any diagram of* Syn_Σ *composed entirely of generators from* (1)*, namely*  $(a \in \Sigma)$ *In other words, automata-diagrams are the morphisms of the sub-prop freely generated by the morphisms of* (1)*. We call this sub-prop* Aut_Σ*.*

We can now state the main result of this paper.

▶ **Theorem 15** (Soundness and Completeness). *For any two automata-diagrams* $c, d : \blacktriangleright \to \blacktriangleright$*,*

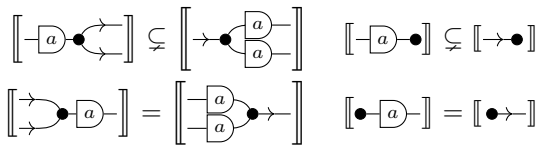
$[\![c]\!] \subseteq [\![d]\!]$ *if and only if* $c \leq_{MDA} d$*.*

The *completeness* of MDA is the most involved and will be the subject of Section 5. Its *soundness* on the other hand is straightforward and is a matter of verifying that each axiom holds in the semantics. Except for axioms of the E block (see Proposition 16), the soundness of the remaining (in)equalities follows from properties which have been proven earlier:

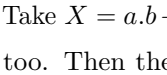
- The A and B blocks encode equalities that hold for any poset. They imply that our category of interest is compact-closed (A) and that every object is equipped with a (bi)commutative bimonoid (B1-B11), a common structure in diagrammatic calculi [16].
- The E block encodes the interaction of the atomic actions with the simulation order. As we already stated, here lies the main difference with [20] – (E1-E2) now only hold laxly.
- The C and F blocks encode the lattice structure of $(\Omega, \leq)$, and all (in)equalities follow from the existence of meets. The F block in particular encodes a number of adjunctions in the following 2-categorical sense: two morphisms $f : X \to Y$ and $g : Y \to X$ are *adjoints* if $\mathrm{id}_X \leq f\,;\, g$ and $g\,;\, f \leq \mathrm{id}_Y$. We write $f \dashv g$ and say that f is left adjoint to g. Here, we have four adjunctions:  and  Note that the adjunctions involving are the key defining adjunctions of Cartesian bicategories [6]. The remaining adjunctions hold whenever the supporting poset is a semi-lattice (has binary meets and top), which is the case for simulation.

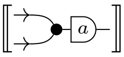
We prove here the soundness of axioms (E1-E4) and that the inequalities corresponding to axioms (E1-E2) are strict.

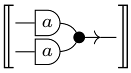
▶ **Proposition 16.** *We have the following (in-)equalities:*

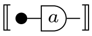
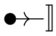


Proof.

1. The proof of the inclusion is straightforward, but the fact that it is strict is more interesting. Take $X = a.b + a.c \in \Omega$. Then $(X, (b, c)) \in$  . Suppose $(X, (b, c)) \in$ too. Then there is a $Y \in \Omega$ such that $a.y \leq x$, and $b, c \leq Y$. By the join property of $+$, we have $b + c \leq Y$ and therefore $a.(b + c) \leq X$. However $a.b + a.c$ does not simulate $a.(b + c)$ since there is no state in $a.b + a.c$ having a b-labelled out-transition as well as a c-labelled one. Thus $(X, (b, c)) \notin$.
2. $[\![\to \bullet]\!] = \{(X, \bullet) : X \in \Omega\} \supseteq \{(a.X, \bullet) : X \in \Omega\} = [\![a \to \bullet]\!]$, and it is clear that for all $X \in \Omega$, $0 \neq a.X$ because 0 has no transitions.

3. We have $[\![$  $]\!] = \{((X_1, X_2), Y) : a.Y \leq X_1, X_2\}$ and

 $= \left\{ ((X_1, X_2), Y) : \exists U_1, U_2, \begin{array}{r} Y \leq U_1, U_2 \\ a.U_1 \leq X_1 \\ a.U_2 \leq X_2 \end{array} \right\}$

   The "$\supseteq$" inclusion follows from the monotony of prefixing, which gives us $a.Y \leq a.U_i \leq X_i$ for $i = 1, 2$. For the other inclusion, choosing $U_1 = U_2 = Y$ gives the desired result.
4. $[\![$  $]\!] = \{(\bullet, Y) : \exists X (a.Y \leq X)\}$ is clearly contained in $\{(\bullet, Y) : Y \in \Omega\} = [\![$  $]\!]$; for the other direction, we can just set $X = a.Y$. ◀

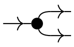
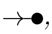
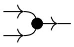
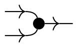
4 Automata-diagrams

This section aims to explain and justify our representation of NFA using string diagrams. Our encoding follows that of [20] – the aim of this section is not merely to reproduce it, but to show that the same syntax is able to represent automata uniformly for different semantics, and to define the notions we will use in the proof of completeness.

- In Section 4.1, we first show that the same encoding is sound for the simulation semantics, in the sense that the diagram c_A which represents a given NFA A has the appropriate semantics, *i.e.*, $[\![c_A]\!] = \{(X, Y) : A.Y \leq X\}$. This is Theorem 23 below.
- In Section 4.2, we prove a converse result: any automaton-diagram $\blacktriangleright \to \blacktriangleright$ can be thought of as a NFA, *i.e.*, for any automaton diagram c, there exists some NFA A_c such that $[\![c]\!] = \{(X, Y) : A_c.Y \leq X\}$. This is Corollary 27 below.

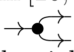
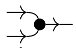
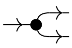
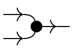
4.1 From automata to string diagrams...

First, we show how to encode relations into our calculus. We will use these to represent the initial and final states of NFA, and simulation relations in the proof of completeness.

▶ **Definition 17** (Relation-diagram). *A* relation-diagram *is a diagram of* Syn_Σ *composed entirely of*  ,  ,  ,  . *We call* RelDiag *the corresponding sub-prop. A relation-diagram is* functional *when it is composed only of*  ,  .

In what follows, we call *block* of a certain subset of the generators of Syn_Σ, any diagram made up entirely of the prescribed generators using vertical and horizontal composition (and, possibly, identities and wire crossings).

▶ **Proposition 18** (Encoding relations). *Given two finite sets* $Q = \{q_i\}_{1 \leq i \leq n}$ *and* $S = \{s_j\}_{1 \leq j \leq m}$, *and some relation* $R \subseteq Q \times S$, *there exists a relation diagram* $d_R : \blacktriangleright^n \to \blacktriangleright^m$ *such that* $[\![d_R]\!] = \{(\mathbf{X}, \mathbf{Y}) \mid X_i \geq Y_j$ *if* $(q_i, s_j) \in R\}$.

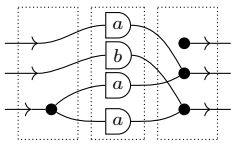
Proof. We reproduce the construction from [20, Section 4]. The relation-diagram d_R $\blacktriangleright^n \to \blacktriangleright^m$ is formed of two blocks: first, a block of  ,  , followed by a block of  ,  . The left i-th port of d_R is connected to the the right j-th port through an identity wire $\longrightarrow$ precisely when $(q_i, s_j) \in R$; we use  to accommodate multiple outgoing connections from a single left port (or none, with ), and  for multiple incoming connections into a right port (or none, with ).

Finally, it is straightforward to see that we have constructed d_R so that its semantics is precisely $\{(\mathbf{X}, \mathbf{Y}) \mid X_i \geq Y_j$ if $(q_i, s_j) \in R\}$. ◀

We now show how to encode the transition relation of any given automaton.

▶ **Definition 19** (Matrix-diagram). *A* matrix-diagram *is a diagram* $\blacktriangleright^n \to \blacktriangleright^m$ *that factors as a composition of a block of*  , , *another of* *for* $a \in \Sigma$, *and a last one of* , , *such that any path from a left to right port encounters exactly one* .

The intuition behind the condition of the previous definition are to *(i)* disallow multiple transitions for the same letter between the same two states, and *(ii)* disallow ϵ-transitions since our definition of NFA does not allow them. For example, the following is is a well-formed matrix-diagram (with the three blocks highlighted), encoding the transition relation $\{(q_0, a, q_1), (q_1, b, q_2), (q_2, a, q_1), (q_2, a, q_2)\}$.

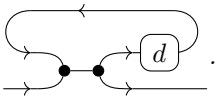


Proposition 20 shows that we can encode any transition relation as a matrix-diagram with the desired semantics.

▶ **Proposition 20** (Encoding transition-relations). *Given a set* $Q = \{q_i\}_{1 \leq i \leq n}$ *and transition relation* $\alpha \subseteq Q \times \Sigma \times Q$, *write* $\mathbf{D}$ *for the corresponding* $n \times n$ *matrix of coefficients in* $\mathcal{P}_f(\Sigma)$. *There exists a matrix-diagram* $d_\alpha : \blacktriangleright^n \to \blacktriangleright^n$ *such that* $[\![d_\alpha]\!] = \{(\mathbf{X}, \mathbf{Y}) : \mathbf{DY} \leq \mathbf{X}\}$.

Proof. Again, the idea is already present in [20, Section 4.2]. Let d_α be the matrix-diagram where the i-th input wire is linked to the j-th output wire via whenever $q_i \xrightarrow{a}_A q_j$. By construction, $[\![d_\alpha]\!] = \{(\mathbf{X}, \mathbf{Y}) : \mathbf{DY} \leq \mathbf{X}\}$, as we wanted. ◀

The last ingredient of the correspondence between automata-diagrams and NFA is iteration. As explained above, and  allow us to form the diagrammatic counterpart of iteration, aka the Kleene star, defined as follows.

▶ **Definition 21** (Star). *For a diagram* $d : \blacktriangleright^n \to \blacktriangleright^n$, *let* :=  .

▶ **Proposition 22** (Stars are least fixpoints). *If* $d : \blacktriangleright^n \to \blacktriangleright^n$ *is a matrix-diagram whose corresponding matrix is* $\mathbf{D}$, *then* $[\![d^*]\!] = \{(\mathbf{X}, \mathbf{Y}) : S(\mathbf{Y}) \leq \mathbf{X}\}$, *where* $S(\mathbf{Y})$ *is the (unique) least solution of the system* $\mathbf{Y} + \mathbf{DX} \leq \mathbf{X}$.

Proof. The semantics of d^* is given by $\{(\mathbf{X}, \mathbf{Y}) : \mathbf{Y} + \mathbf{DX} \leq \mathbf{X}\}$, which is equal to $\{(\mathbf{X}, \mathbf{Y}) : S(\mathbf{Y}) \leq \mathbf{X}\}$ by Theorem 8. ◀

▶ **Theorem 23** (Encoding of NFA). *Let* $A = (Q, q_0, F, \alpha)$ *be an automaton. There exists an automaton-diagram* $c_A : \blacktriangleright \to \blacktriangleright$ *such that* $[\![c_A]\!] = \{(X, Y) : A.Y \leq X\}$.

Proof. We represent automata as in [20, Section 4.2]. First, fix some ordering of $Q = \{q_i\}_{1 \leq i \leq n}$. Then

- $e : \blacktriangleright \to \blacktriangleright^n$ is the relation-diagram encoding the singleton $\{(q_0, \bullet)\}$, using Proposition 18;
- $f : \blacktriangleright^n \to \blacktriangleright$ is the relation-diagram encoding the $\{(q_i, \bullet) : q_i \in F\}$, using Proposition 18;
- $d : \blacktriangleright^n \to \blacktriangleright^n$ is the matrix-diagram representing α, using Proposition 20. It is such that $[\![d]\!] = \{(\mathbf{X}, \mathbf{Y}) : \mathbf{DY} \leq \mathbf{X}\}$.

Then, let $c_A = e; d^*; f$ where d^* is defined as in Definition 21. By Proposition 22, $[\![d^*]\!] = \{(\mathbf{X}, \mathbf{Y}) : S(\mathbf{Y}) \leq \mathbf{X}\}$. Then we have:

$$[\![c_A]\!] = \left\{ (X, Y) : \exists \mathbf{X}, \mathbf{Y}, \begin{array}{c} \forall i, q_i \in F \Rightarrow \mathbf{Y}_i \leq Y \\ S(\mathbf{Y}) \leq \mathbf{X} \\ \mathbf{X}_0 \leq X \end{array} \right\}$$

We want to show that $[\![c_A]\!]$ is equal to $\{(X, Y) : A.Y \leq X\}$ by double-inclusion.

(⊆) Let $X, Y \in \Omega$ be such that $A.Y \leq X$. Then take $\mathbf{Y} = \mathbf{F}.Y$ and $\mathbf{X} = S(\mathbf{Y})$. By definition, if $q_i \in F$ then $\mathbf{Y}_i \leq Y$, and $S(\mathbf{Y}) \leq \mathbf{X}$.
Moreover $S(\mathbf{Y}) = S(\mathbf{F}.Y) = \mathbf{A}.Y$ by Theorem 8. Thus $\mathbf{X}_0 = (S(\mathbf{Y}))_0 = (\mathbf{A}.Y)_0 = A_0.Y = A.Y \leq X$, and therefore $(X, Y) \in [\![c_A]\!]$.

(⊇) Let $(X, Y) \in [\![c_A]\!]$ and $\mathbf{X}, \mathbf{Y}$ be two vectors of behaviours witnessing this membership. To conclude the proof of the second inclusion, we have $A.Y = (\mathbf{A}.Y)_0 = (S(\mathbf{F}.Y))_0 \leq (S(\mathbf{Y}))_0 \leq \mathbf{X}_0 \leq X$ where the third step is our hypothesis.

Finally, we need to show that this representation is independent of the choice of ordering of Q. This is the case, because any other choice of total order induces a permutations of the wires. In other words, given another ordering, from which we obtain two other relation-diagrams e' and f' encoding initial and final states, and a matrix-diagram d' encoding the transition relation of A, there exists some permutation $\pi : \blacktriangleright^n \to \blacktriangleright^n$ such that $d' = \pi\,;\, d\,;\, \pi^{-1}$, $e' = e\,;\, \pi^{-1}$, and $f' = \pi\,;\, f$. A simple diagrammatic derivation shows that $d'^* = \pi\,;\, d^*\,;\, \pi^{-1}$ and therefore $e'\,;\, d'^*\,;\, f' = e\,;\, d\,;\, f$. ◀

As a result, by soundness, an inequality between two automata-diagrams implies the existence of a simulation between the corresponding NFA *in the reverse order*:

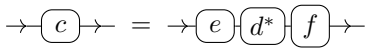
▶ **Corollary 24.** *Let A, B be two NFA and c_A, c_B the corresponding automaton-diagram obtained from Theorem 23. If $c_A \leq c_B$, then $B \leq A$.*

Proof. The proof is straightforward: by soundness, $c_A \leq c_B$ means $[\![c_A]\!] \subseteq [\![c_B]\!]$. Since $A.1 = A \leq A$, we have $(A, 1) \in [\![c_A]\!]$. Thus $(A, 1) \in [\![c_B]\!]$, *i.e.*, $1.B = B \leq A$. ◀

4.2 ...and back

Theorem 23 shows how to encode a given NFA as a string diagram into our diagrammatic syntax. Conversely, given a $\blacktriangleright \to \blacktriangleright$ automaton-diagram, we can extract the NFA it represents, by rewriting it into a form which mimics the encoding of NFA of the previous section – this is what we call a *representation*.

▶ **Definition 25** (Representation). *For a diagram $c : \blacktriangleright \to \blacktriangleright$, a* representation *is a triple (e, d, f) of a matrix-diagram, $d : \blacktriangleright^\ell \to \blacktriangleright^\ell$, one functional relation-diagram $e : \blacktriangleright \to \blacktriangleright^\ell$, and one relation-diagram $f : \blacktriangleright^\ell \to \blacktriangleright$, such that*



The intuition is that d represents the transition relation of the associated automaton, e the initial state, and f its final states.

▶ **Proposition 26.** *Any automaton-diagram $\blacktriangleright \to \blacktriangleright$ has a representation.*

Proof. The proof is the same as [20, Proposition 4.7]. All axioms used in this proof are in MDA – crucially, it does not use left-distributivity, but only right-distributivity (E3-E4). ◀

From here it is easy to extract a diagrammatic representation of its initial state, final states, and transition relations.

▶ **Corollary 27** (NFA from automaton-diagram). *Given an automaton-diagram $c : \blacktriangleright \to \blacktriangleright$, there exists a NFA A such that $[\![c]\!] = \{(X, Y) \mid A.Y \leq X\}$.*

Proof. First, by Proposition 26, we can find a representation (e, d, f) of c. We construct $A = (Q, q_0, F, \alpha)$. First, if $d : \blacktriangleright^n \to \blacktriangleright^n$, let $Q = \{1, \ldots, n\}$. Then let q_0 be the only i such that $(\bullet, i) \in \mathfrak{R}(e)$ (remember that $\mathfrak{R}(e)$ is a $n \times 1$ Boolean matrix, which is moreover functional, so that it is fully characterised by a single i between 1 and n). Let $F := \{j : (j, \bullet) \in \mathfrak{R}(f)\}$. Finally, the transition relation is determined by the matrix-diagram d. Call $\mathbf{D}$ be the $n \times n$ matrix with coefficients in $\mathcal{P}_f(\Sigma)$ obtained from Proposition 33. Then, let $(i, a, j) \in \alpha$ if $\mathbf{D}_{i,j}$ contains a (remember that the coefficients are finite subsets of Σ). This is well-defined because d is assumed to be ϵ-free. ◀

5 Completeness

The main idea to tackle completeness is that simulation relations themselves can be encoded as relation-diagrams into our calculus.

We have already shown how to encode relations as relation-diagrams; now we explain how to go in the other direction. From any relation-diagram $d : \blacktriangleright^n \to \blacktriangleright^m$ we can obtain a relation between $\{1, \ldots, n\}$ and $\{1, \ldots, n\}$, *i.e.*, a matrix with Boolean coefficients. As we will need to manipulate these relations in calculations below, we formalise the correspondence between relation-diagrams and relations as a *functor* from RelDiag (the sub-prop freely generated by these diagrams) to $\mathsf{Mat}_{\mathbb{B}}$, the SMC of Boolean matrices with the disjoint sum as monoidal product. The latter has natural numbers as objects and $m \times n$ matrices with Boolean coefficients as morphisms $n \to m$. Its morphisms can also be ordered by inclusion if we think of them as relations: given two Boolean $n \times m$ matrices $A = (a_{ij})$ and $B = (b_{ij})$, we write $A \leq B$ if $a_{ij} \leq b_{ij}$ for all i and j.

▶ **Definition 28.** *Let $\mathfrak{R}(\cdot)$ be the mapping given by:*

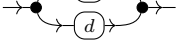
$$\mathfrak{R}\left(\to\bullet\right) = \begin{pmatrix}1\\1\end{pmatrix} \qquad \mathfrak{R}(\to\bullet) = 1 \qquad \mathfrak{R}\left(\bullet\to\right) = \begin{pmatrix}1 & 1\end{pmatrix} \qquad \mathfrak{R}(\bullet\to) = 1$$

By the freeness of RelDiag, we obtain the following result immediately.

▶ **Proposition 29.** *$\mathfrak{R}(\cdot)$ extends to a symmetric monoidal functor* RelDiag $\to \mathsf{Mat}_{\mathbb{B}}$.

We will use extensively the fact that MDA is complete for relation-diagrams.

▶ **Theorem 30** (Completeness for relation-diagrams). *If c, d are two relation-diagrams, then $c \leq_{MDA} d$ iff $\mathfrak{R}(d) \leq \mathfrak{R}(c)$.*

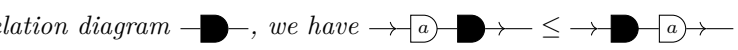
Proof. This is a standard completeness result for the symmetric monoidal theory of an idempotent (co)commutative bimonoid [7, Theorem 7.2], *i.e.* axioms (B1)-(B11). While it is usually stated for equalities, the extension to inequalities is straightforward, since inequalities can be recovered from the semi-lattice structure of the binary operation defined by  and  , that is we can show that $c \leq d$ iff  $= c$. See [20, Propositions 5.2-5.3] for the detailed proof. ◀

We will need the fact that we can (co)copy and (co)delete any relation-diagram.

▶ **Lemma 31** (Distributivity for relation-diagrams). *Any relation-diagram* $d : \blacktriangleright^m \to \blacktriangleright^n$ *satisfies*

Proof. Since the corresponding equalities hold in $\mathsf{Mat}_{\mathbb{B}}$, we can deduce the two syntactic equalities from Theorem 30 (completeness for relations). ◀

We will need a particularly simple form of simulation below for a single letter.

▶ **Lemma 32.** *For any relation diagram*  , *we have* 

Proof. By structural induction. The two inductive cases for composition and monoidal product are straightforward. The base cases are the axioms (E1-E4). ◀

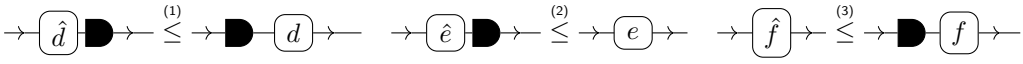
In the previous section, we have used matrix-diagrams (Definition 19) to encode the transition relations of NFA (Proposition 20). Clearly, we can also go the other way, associating a unique transition relation δ to each matrix-diagram.

▶ **Proposition 33.** *For any matrix-diagram* $d : \blacktriangleright^n \to \blacktriangleright^n$, *there exists a* $n \times n$ *matrix* $\mathbf{D}$ *with coefficients in* $\mathcal{P}_f(\Sigma)$ *such that* $[\![d]\!] = \{(\mathbf{X}, \mathbf{Y}) : \mathbf{DY} \leq \mathbf{X}\}$.

Proof. This is obvious from the way matrix-diagrams are defined. We can obtain the (i, j)-th coefficient of $\mathbf{D}$ by plugging $\bullet\!\!\rightarrow$ and $\rightarrow\!\!\bullet$ into all other left and right ports of d. ◀

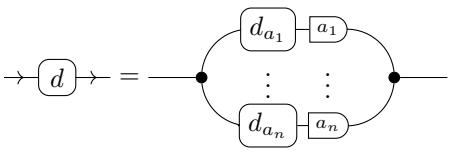
We can now show that the relation-diagram encoding a given simulation does satisfy the diagrammatic analogues of the three defining properties of simulation from Definition 2.

▶ **Lemma 34** (Simulation for representations). *Given two NFA* A *and* $\hat{A}$, *let* (e, d, f) *and* $(\hat{e}, \hat{d}, \hat{f})$ *be the representations associated to their respective encoding as automata-diagrams. If* $A \leq \hat{A}$, *there is a relation-diagram*  *such that:*

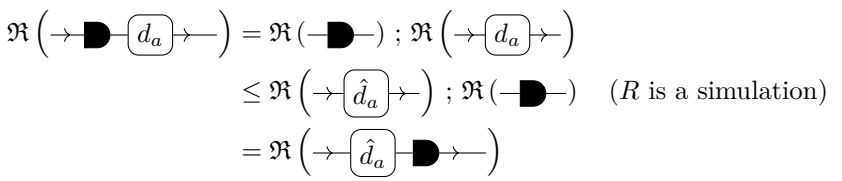


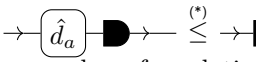
Proof. Assume that R is a simulation relation witnessing $A \leq \hat{A}$ and let  be the relation-diagram encoding R^{-1}, using Proposition 18 with the ordering of the states of A and $\hat{A}$ already fixed by the choice of the representations in the statement of the lemma. Let us prove  satisfies the required inequalities.

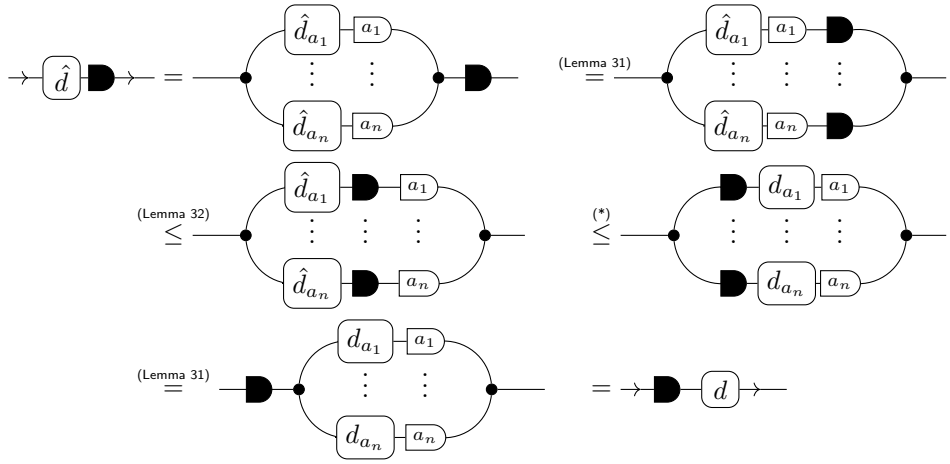
(1) For $a \in \Sigma$, let d_a be relation-diagram encoding the relation $\xrightarrow{a}_A$. Then, if $\Sigma = \{a_1, \ldots, a_n\}$, it is easy to check that



by applying the (E3-E4) axioms to merge letters, as well as the (co)unit axiom for the black (co)monoid (recall that, as a relation-diagram  can be described as a block of  , composed with a block of ,). Naturally, this also holds for $\hat{d}$.
Then, we show that behaves like a simulation for all d_a:



By completeness for relations (Theorem 30), we get the reverse syntactic inequality:  $\overset{(*)}{\leq}$. Finally, putting it all together and using copy and merge laws for relations, we have:



(2-3) Those two inequalities are just straightforward applications of completeness for relations and the definition of simulation. ◀

The first inequality of the previous lemma is insufficient, because automata-diagrams factor as $e; d^*; f$, not $e; d; f$, for a given representation. The rest of the proof is thus dedicated to lifting Lemma 34(1) to d^* instead of just d, using a proof similar to that of [20, Section 5] (of which we only sketch the main ideas). Crucially, this is where the white generators  , , and their associated axioms come into play.

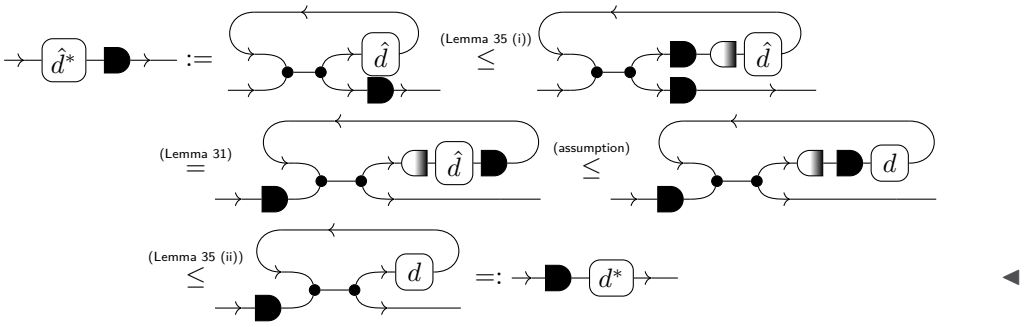
First, we build a right adjoint to a given simulation relation(-diagram) using , (see [20, Section 5.2] for the details). Then, the F axioms of MDA are enough to show the necessary adjunction: $\dashv$, this is [20, Lemma 5.14].

▶ **Lemma 35.** *For any relation-diagram* *, there exists a diagram* *, such that the following inequalities hold: (i)* $\overset{m}{\longrightarrow} \leq$ *and (ii)* $\leq \overset{n}{\longrightarrow}$.

We can now lift the simulation relation to d^* as we wanted.

▶ **Lemma 36.** *If* $\leq$ *then* $\leq$.

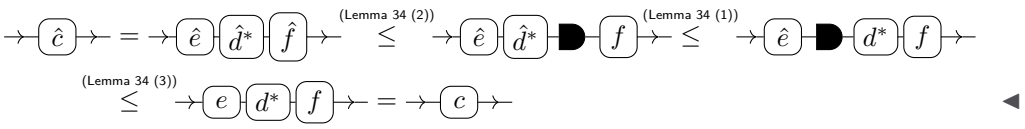
Proof. We have



◀

We are finally ready to finish our proof of the completeness of MDA.

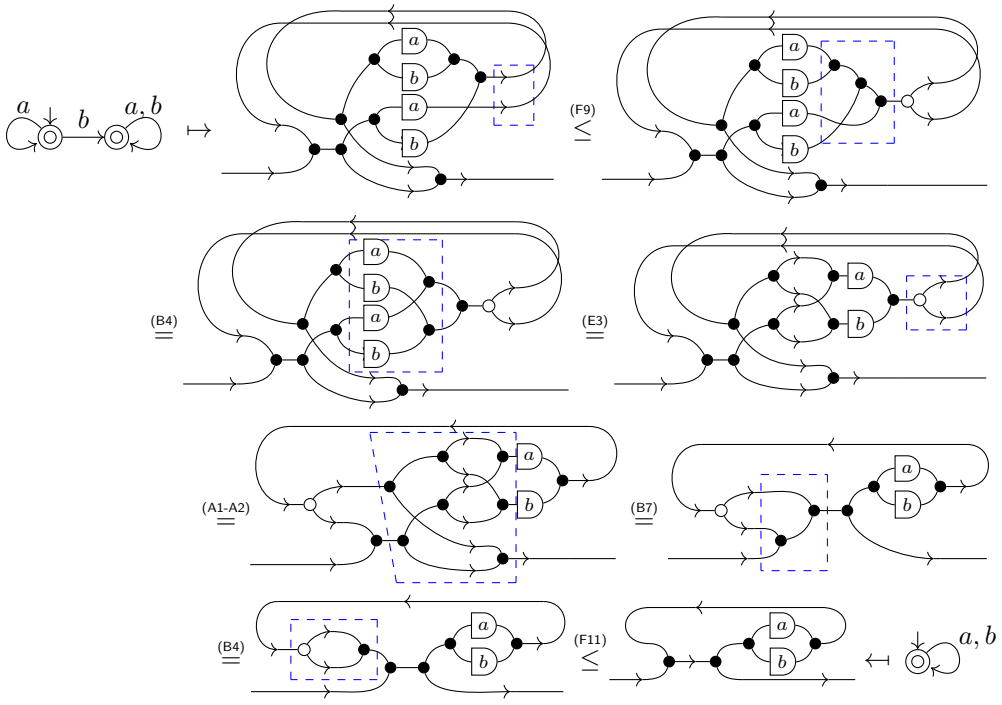
Proof of Theorem 15 (completeness). Given two automata-diagrams $c, \hat{c} : \blacktriangleright \to \blacktriangleright$, we can first extract their respective representations (e, d, f) and $(\hat{e}, \hat{d}, \hat{f})$, using Proposition 26. From Corollary 27, we can recover two a NFA A and $\hat{A}$ such that $[\![c]\!] = \{(X, Y) \mid A.Y \leq X\}$ and $[\![\hat{c}]\!] = \{(X, Y) \mid \hat{A}.Y \leq X\}$. If we also assume, as in the statement of the theorem, that $[\![\hat{c}]\!] \subseteq [\![c]\!]$, then, since $\hat{A}.1 = \hat{A} \leq \hat{A}$, we have $(\hat{A}, 1) \in [\![\hat{c}]\!]$. Thus also $(\hat{A}, 1) \in [\![c]\!]$, *i.e.*, $A.1 = A \leq \hat{A}$. In other words, $\hat{A}$ simulates A. Given such a simulation, we can encode it as a relation diagram —●— and conclude the proof using Lemma 34 as follows:



◀

▶ Remark 37 (Completeness for arbitrary automata-diagrams). The results in this paper are stated for $\blacktriangleright \to \blacktriangleright$ automata-diagrams, which correspond precisely to NFA (*cf.* Section 4). It is natural to wonder whether they extend to automata-diagrams of arbitrary type. The short answer is yes. First of all, we can always bend the wires of any given automaton-diagram $d : v \to w$ to obtain one of type $\blacktriangleright^n \to \blacktriangleright^m$. Semantically, this only amounts to changing whether a given variable appears on the left or on the right of the relation $[\![d]\!]$ [20, Proposition 5.4]. Second, it is possible to show [20, Theorem 5.5] that any automaton-diagram $\blacktriangleright^n \to \blacktriangleright^m$ distributes over  and ●—. As a result, any automaton-diagram $\blacktriangleright^n \to \blacktriangleright^m$ is fully characterised by n diagrams of type $\blacktriangleright \to \blacktriangleright^m$. Now, contrary to [20], automata-diagrams $\blacktriangleright^n \to \blacktriangleright^m$ in this paper do *not* distribute over —●  and —● in general (they do so only laxly). This means that we cannot reduce the completeness for automata-diagrams $\blacktriangleright^n \to \blacktriangleright^m$ to that of automata-diagrams $\blacktriangleright \to \blacktriangleright$. However, it is possible to define a notion of NFA with multiple sets of final states and to adapt the definition of simulation correspondingly: if $(q, s) \in R$ for some simulation relation R between two such NFA A and B, each with m sets of final states $\{F_i\}_{1 \leq i \leq m}$ and $\{G_i\}_{1 \leq i \leq m}$, then we have $F_i(q) \leq G_i(s)$ for all $1 \leq i \leq m$. Then, all results of this paper generalise to automata-diagrams $\blacktriangleright \to \blacktriangleright^m$ (with multiple right ports, but only one left port) and we can prove that MDA is complete for all automata-diagrams. While this is a stronger result, we have preferred to state our main result for the class of diagrams that correspond more closely to plain NFA, as the more general notion is not standard and introduces distracting complications.

▶ **Example 38.** We now come back to the example from the introduction and show how to prove the similarity of the two NFA.



The other inequality can be proven entirely analogously, replacing  with  , axiom (F9) with (F1), and axiom (F11) with (F3).

6 Conclusion

In this paper, we have successfully provided a finite axiomatisation of NFA modulo similarity, using a string diagrammatic syntax.

Related work. In doing so, we have built on an earlier diagrammatic axiomatisation of NFA *up to language equivalence* [20]. We have shown here that the same syntax is able to accommodate a different semantics and can be axiomatised with a slight change of (in)equational theory: left-distributivity of —[a]— over  and is now lax. This change reflects the well-known fact that simulation implies language-inclusion. Another interesting corollary is that, *for deterministic automata(-diagrams)*, lax left-distributivity is sufficient to prove language-inclusion (*i.e.*, if we can show that $c \leq d$ using the axioms of [20], we can show it using only those of MDA).

Our axiomatisation is also closely related to an existing axiomatisation of regular CCS expressions up to similarity [9]. That work builds on Milner's axiomatisation of *bisimilarity*, adding the axiom $E \leq E+F$, an axiom which is derivable in our calculus. The main difference with our work is that this axiomatisation contains implicational axioms for fixpoints. In contrast, our axiomatisation is *finite*, as was the case in our earlier work on language equivalence [20]. We were able to achieve this by using a more expressive diagrammatic calculus, in which we can encode not only the simulation relations themselves, but the proof that they satisfy the desired properties.

Future work. First, we would like to characterise the expressiveness and give a complete axiomatisation of the full syntax (including the white generators) for the simulation semantics.

Second, we want to axiomatise the same syntax up to *bisimilarity*. Recall that a relation R is a bisimulation between two NFA when both R and R^{-1} are simulations. The present completeness result implies it is possible to check "by hand" that a relation-diagram encoding R witnesses the bisimulation between two automata-diagrams c and d: we have to prove that $c \leq d$ and $d \leq c$, as in the proof of Theorem 15, using R in one direction and R^{-1} in the other. However, this requires us to keep track of which simulation relation we use in each direction, since the relation $\leq$ itself omits this crucial piece of information.

Excitingly, this paves the way for a 2-categorical approach to the theory of bisimilarity, a perspective which would allow us to track the way in which two diagrams are related explicitly: for c, d two automata-diagrams and r a relation-diagram, $r : c \Rightarrow d$ is a 2-morphism whenever r is a simulation. We aim to show that this defines a (symmetric monoidal) 2-category and find a presentation for it, using 2-morphisms to replace the axioms of MDA and further equalities between them. Since bisimulations would be 2-isomorphisms in this setting, such a presentation would allow us to construct them as 2-morphisms and prove that they are invertible and satisfy the required properties using the additional equalities.

Finally, we would like to translate the axioms of MDA into transformations of the state-transition graph of the corresponding automata, building on the extensive work on formulating string diagram rewriting as rewriting of hypergraphs [3, 4].

References

1 John Baez, Brandon Coya, and Franciscus Rebro. Props in network theory. *Theory and Applications of Categories*, 33(25):727–783, 2018.

2 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

3 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Paweł Sobociński, and Fabio Zanasi. Rewriting modulo symmetric monoidal structure. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2016.

4 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Paweł Sobociński, and Fabio Zanasi. Rewriting with Frobenius. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 165–174, 2018.

5 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Deconstructing Lawvere with distributive laws. *Journal of logical and algebraic methods in programming*, 95:128–146, 2018. doi:10.1016/J.JLAMP.2017.12.002.

6 Aurelio Carboni and RFC Walters. Cartesian bicategories I. *Journal of pure and applied algebra*, 49(1-2):11–32, 1987.

7 Brandon Coya and Brendan Fong. Corelations are the prop for extraspecial commutative frobenius monoids. *Theory and Applications of Categories*, 32(11):380–395, 2017.

8 Brendan Fong and David I Spivak. *An invitation to applied category theory: seven sketches in compositionality*. Cambridge University Press, 2019.

9 Ulrik Frendrup and Jesper Nyholm Jensen. *A complete axiomatization of simulation for regular CCS expressions*. BRICS, Department of Computer Science, Univ., 2001.

10 CAR Hoare, Bernhard Möller, Georg Struth, and Ian Wehrman. Concurrent Kleene algebra. In *Proceedings of the 20th International Conference on Concurrency Theory (CONCUR)*, pages 399–414. Springer, 2009.

11 Martin Hyland and Andrea Schalk. Glueing and orthogonality for models of linear logic. *Theoretical Computer Science*, 294(1-2):183–231, 2003. doi:10.1016/S0304-3975(01)00241-9.

12 Gregory M Kelly and Miguel L Laplaza. Coherence for compact closed categories. *Journal of Pure and Applied Algebra*, 19:193–213, 1980.

13 Stephen C Kleene. Representation of events in nerve nets and finite automata. Technical report, RAND PROJECT AIR FORCE SANTA MONICA CA, 1951.

14 Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, 1994. doi:10.1006/INCO.1994.1037.

15 Dexter Kozen. Kleene algebra with tests. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 19(3):427–443, 1997. doi:10.1145/256167.256195.

16 Stephen Lack. Composing PROPs. *Theory and Application of Categories*, 13(9):147–163, 2004.

17 F William Lawvere. Functorial semantics of algebraic theories. *Proceedings of the National Academy of Sciences of the United States of America*, 50(5):869, 1963.

18 Robin Milner. A complete inference system for a class of regular behaviours. *Journal of Computer and System Sciences*, 28(3):439–466, 1984. doi:10.1016/0022-0000(84)90023-0.

19 Andrew M Moshier. Coherence for categories of posets with applications. *Topology, Algebra and Categories in Logic (TACL)*, page 214, 2015.

20 Robin Piedeleu and Fabio Zanasi. A finite axiomatisation of finite-state automata using string diagrams. *Logical Methods in Computer Science*, 19, 2023. doi:10.46298/LMCS-19(1:13)2023.

21 Robin Piedeleu and Fabio Zanasi. An introduction to string diagrams for computer scientists. *arXiv preprint arXiv:2305.08768*, 2023. doi:10.48550/arXiv.2305.08768.

22 Valentin N Redko. On defining relations for the algebra of regular events. *Ukrainskii Matematicheskii Zhurnal*, 16:120–126, 1964.

23 Steffen Smolka, Nate Foster, Justin Hsu, Tobias Kappé, Dexter Kozen, and Alexandra Silva. Guarded Kleene algebra with tests: verification of uninterpreted programs in nearly linear time. *Proceedings of the 47th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL)*, 4:1–28, 2020. doi:10.1145/3371129.

A Algebraic operations on NFA

We define here precisely the operation we use in the paper. Fix two NFA $A = (Q, q_0, F, \alpha)$ and $B = (S, s_0, G, \beta)$.

▶ **Definition 39** (Sum). *We define the sum of A and B, denoted $A + B$ as follows: $A + B = (T, t_0, H, \gamma)$ where:*

- $T = Q \sqcup S \sqcup \{t_0\}$, *with t_0 neither appearing in Q nor S.*
- $H(t) = F(t)$ *if $t \in Q$, $H(t) = G(t)$ if $t \in S$ and $H(t_0) = F(q_0) \vee G(t_0)$*
- $(t, a, t') \in \gamma$ *whenever $(t, a, t') \in \alpha$, or $(t, a, t') \in \beta$, or $t = t_0$ and $(q_0, a, t) \in \alpha$ or $(s_0, a, t) \in \beta$.*

Intuitively, summing A and B only consists in merging their initial states into one which mimics the behaviour of both.

▶ **Definition 40** (Synchronous Product). *The product $A \times B = (T, t_0, H, \gamma)$ of A and B is defined as:*

- $T = Q \times S$
- $t_0 = (q_0, s_0)$
- $H(q, s) = F(q) \wedge G(s)$
- $(q, s) \xrightarrow{a}_{A \times B} (q', s')$ *whenever $q \xrightarrow{a}_A q'$ and $s \xrightarrow{a}_B s'$*

Intuitively, every path in $A \times B$ is also a path in both A and B.

▶ **Definition 41** (Composition). *The composition $A.B = (T, t_0, H, \gamma)$ of A and B is defined as:*

- $T = Q \sqcup \{i \mid F(q_i) = 1\} \times (S \setminus \{s_0\})$.
- $t_0 = q_0$
- $H(q) = F(q) \wedge G(s_0)$ *for $q \in Q$, and $H(i, s) = G(s)$ for $s \in S$*

- *There are three cases for transitions in $A.B$:*
  - *For $q, q' \in Q$, $q \xrightarrow{a}_{A.B} q'$ whenever $q \xrightarrow{a}_{A} q'$,*
  - *For $s, s' \in S$, $(i, s) \xrightarrow{a}_{A.B} (i, s')$ whenever $s \xrightarrow{a}_{B} s'$*
  - *For $q_i \in Q$ such that $F(q_i) = 1$ and $s \in S$, $q_i \xrightarrow{a}_{A.B} (i, s)$ whenever $s_0 \xrightarrow{a}_{B} s$*

Intuitively, to every final state in A we "attach" an entire copy of B.

▶ **Definition 42** (Prefix). *If $a \in \Sigma$ then A prefixed by a, written $a.A$, is the composition of $(\{q_0, q_1\}, q_0, \{q_0 \mapsto 0, q_1 \mapsto 1\}, \{(q_0, a, q_1)\})$ and A.*

We will also write $S.A$ for $S \in \mathcal{P}_f(\Sigma)$ some finite subsets of Σ, as a shorthand for $\left(\sum_{a \in S} a\right).A = \sum_{a \in S} a.A$.

B Background on SMCs and props

B.1 Props, String Diagrams, and Symmetric Monoidal Theories

After having defined our syntax, the free prop $\mathsf{P}_{\mathcal{S}}$ over signature $\mathcal{S}$, we give its interpretation into Sem, a SMC that constitutes our target semantics. To guarantee a compositional interpretation, we require $[\![\cdot]\!] : \mathsf{P}_{\mathcal{S}} \to \mathsf{Sem}$, the mapping of terms to their intended semantics, to be a symmetric monoidal functor.

Once we have specified $[\![\cdot]\!] : \mathsf{P}_{\mathcal{S}} \to \mathsf{Sem}$, it is natural to look for equations to reason about semantic equality directly on the diagrams themselves. Given a set of equations E, i.e., a set containing pairs of morphisms of the same type, we write $=_E$ for the smallest congruence w.r.t. the two composition operations ; and $\oplus$. We say that $=_E$ is *sound* if $c =_E d$ implies $[\![c]\!] = [\![d]\!]$. It is moreover *complete* when the converse implication also holds. We call a pair $(\mathcal{S}, E)$ a *symmetric monoidal theory* (SMT) and we can form the prop $\mathsf{P}_{\mathcal{S},E}$ obtained by quotienting each homset of $\mathsf{P}_{\mathcal{S}}$ by $=_E$. There is then a prop morphism $q : \mathsf{P}_{\mathcal{S}} \to \mathsf{P}_{\mathcal{S},E}$ witnessing this quotient.

The reader familiar with categorical logic, may find it helpful to know that the concrete description above can be described in more abstract categorical terms, in line with Lawvere's account of algebraic theories [17]: signatures can be organised into a category and the free prop $\mathsf{P}_{\mathcal{S}}$ given as a monad structure over this category. Furthermore, the category of props and prop morphisms is equivalent to the category of algebras for this monad. Then, by standard abstract nonsense, the prop $\mathsf{P}_{\mathcal{S},E}$ and the quotient morphism q arise as a coequaliser of free props. A detailed account of this presentation can be found in [1, Appendix A.2].

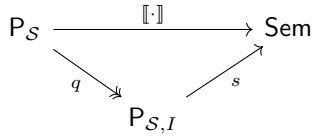
B.2 (Pre-)Ordered Props and Symmetric Monoidal Inequality Theories

Our semantic prop Sem often carries additional structure that we wish to lift to the syntax: monotone relations *qua* relations can be ordered by inclusion. The corresponding mathematical structure is that of an *ordered (or order-enriched) prop*, a prop whose homsets are also posets, with composition and monoidal product monotone maps.

In the same way that props can be presented by SMTs, an ordered prop can be presented by *symmetric monoidal inequality theory* (SMIT). Formally, the data of a SMIT is the same as that of a SMT: a signature $\mathcal{S}$ and a set I of pairs $c, d : X \to Y$ of $\mathsf{P}_{\mathcal{S}}$-arrows of the same type, that we now read as *inequalities* $c \leq d$.

As for plain props, we can construct a pre-ordered prop from a SMIT by building the free prop $\mathsf{P}_{\mathcal{S}}$ and passing to a quotient $\mathsf{P}_{\mathcal{S},I}$: we first build the pre-order on each homset by closing I under $\oplus$ and taking the reflexive and transitive closure of the resulting relation. Finally, we obtain $\mathsf{P}_{\mathcal{S},I}$ by quotienting the resulting prop by imposing anti-symmetry.

SMITs subsume SMTs, since every SMT can be presented as a SMIT, by splitting each equation into two inequalities. As a result, in the main text, we only consider SMITs, referring to them simply as *theories*, and their defining inequalities as *axioms*. When referring to a sound and complete theory, we will also use the term *axiomatisation*, as is standard in the literature. The situation for a sound and complete theory is summarised as a commutative diagram:



Soundness simply means that $[\![\cdot]\!]$ factors as $s \circ q$ through $\mathsf{P}_{\mathcal{S},E}$ and completeness means that s is a faithful prop morphism.

Strong Induction Is an Up-To Technique

Filippo Bonchi
Universitá di Pisa, Italy

Elena Di Lavore
Universitá di Pisa, Italy

Anna Ricci
Universitá di Pisa, Italy

Abstract

Up-to techniques are enhancements of the coinduction proof principle which, in lattice theoretic terms, is the dual of induction. What is the dual of coinduction up-to? By means of duality, we illustrate a theory of induction up-to and we observe that an elementary proof technique, commonly known as strong induction, is an instance of induction up-to. We also show that, when generalising our theory from lattices to categories, one obtains an enhancement of the induction definition principle known in the literature as comonadic recursion.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Logic and verification

Keywords and phrases Induction, Coinduction, Up-to Techniques, Induction up-to, Lattices, Algebras

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.28

Funding This study was carried out within the National Centre on HPC, Big Data and Quantum Computing - SPOKE 10 (Quantum Computing) and received funding from the European Union Next-GenerationEU - National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT N. 1.4 – CUP N. I53C22000690001. This research was partly funded by the Advanced Research + Invention Agency (ARIA) Safeguarded AI Programme.
Filippo Bonchi: Supported by the Ministero dell'Università e della Ricerca of Italy grant PRIN 2022 PNRR No. P2022HXNSC - RAP (Resource Awareness in Programming).

Acknowledgements The authors would like to thank Jurriaan Rot for the inspiring discussions.

1 Introduction

Induction is a fundamental tool frequently used by mathematicians, logicians, and computer scientists without much thought. It includes both *definition* and *proof* principles. The definition principle allows for the specification of data types, such as natural numbers, lists, or trees, and to define functions from them; the proof principle enables proving properties on such inductively defined structures.

The coinduction proof principle, which is formally the dual of induction, is less familiar. It first emerged in the 1970s [33] in three independent fields: set theory [14], modal logic [40], and concurrency theory [27]. Since then, it has been recognized as a fundamental principle in computer science and has been applied in various contexts [24, 1, 11, 16, 12, 31, 25, 17, 22].

Up-to techniques are enhancements of the coinduction proof principle, originally introduced by Milner in [23] to simplify coinductive arguments. Coinduction up-to has proven useful, if not essential, in numerous proofs about concurrent systems (see [30] for references). It has been used to establish decidability results [9], improve standard automata algorithms [6], and prove the completeness of domains in abstract interpretation [3].

Table 1 The lattices-to-categories correspondence. On the left, the proof and definition principles with their enhacement; On the right, the corresponding inductive invariants and algebras.

induction proof principle	induction definition principle	$f(x) \sqsubseteq x$	$a \colon FX \to X$
induction up-to	comonadic recursion	$fd(x) \sqsubseteq x$	$a \colon FDX \to X$
strong induction	course-of-value iteration	$ff^{\downarrow}(x) \sqsubseteq x$	$a \colon FF^{\downarrow}X \to X$

The theory of up-to techniques was initially developed by Sangiorgi [32] and later generalized to the abstract setting of complete lattices by Pous [28, 29]. In particular, Pous introduced the notion of *f-compatible* techniques for some monotone map f. Intuitively, these are are sound up-to techniques with advantageous composition properties.

A curiosity that may have occurred to many is the following:

Since coinduction is the dual of induction, what is the dual of coinduction up-to?

In this paper, we introduce a theory of induction up-to by simply dualizing the work of Pous [28]. Our main finding is that the well-known principle of *strong induction* over the set natural numbers $\mathbb{N}$, a principle familiar even to undergraduate students, is an example of an inductive up-to technique.

More precisely, we dualize the notion of f-compatible techniques from [28] to that of *f-cocompatible* (Definition 5) up-to techniques, which, as expected, are sound (Theorem 7) and enjoy good composition properties (Proposition 8). We show that, under mild conditions, any proof by coinduction up-to can equivalently be carried out by means of induction up-to, and vice versa (Proposition 11).

For any monotone map f, its down-closure, denoted by $f^{\downarrow}$, is always f-cocompatible (Corollary 9). We name induction up-to $f^{\downarrow}$ *strong induction* since, when f is the monotone map with the least fixed point $\mathbb{N}$, induction up-to $f^{\downarrow}$ coincides with the usual strong induction over $\mathbb{N}$ (Section 6.1). Unsurprisingly, the same approach can be applied to obtain strong induction on words (Section 6.2) and other inductive data types.

Overall, this shows that induction up-to, and in particular strong induction, provide enhancements for the induction proof principle. At this point, another curiosity arises:

What are enhancements of the induction definition principle?

Intuitively, one can think of *recursion schemes* as enhancements of the induction definition principle: they ensure that the specification of a recursive function is well-defined.

To formalize this intuition, we exploit the fact that Pous' theory [28] has a beautiful categorical meaning [5]: when generalizing this theory from lattices to categories, one obtains Bartel's λ-coinduction [2], an enhancement of the coinduction definition principle that generalizes several specification techniques common in computer science [36, 20], notably the abstract GSOS by Turi and Plotkin [37, 21].

We illustrate that, following the same pattern, the theory of induction up-to generalize to a certain recursion scheme known as *comonadic recursion* by Capretta, Uustalu, Vene and Pardo [39, 8] (Proposition 18). In particular, strong induction generalises to a scheme known as *course-of-value iteration* [38, 7] (Proposition 20). These correspondences are summarised in Table 1.

Related Work

An elegant theory of inductive enhancements has been recently introduced by Sangiorgi in [35]. This theory substantially differs from ours in its goals: Sangiorgi aims to enhance the proof methods for those behavioural equivalences and preorders [41, 42], such as trace, failure, and ready, that are defined inductively. These relations can usually be stratified, and the proposed inductive enhancements are functions on relations preserving such stratification. Like in the theory of coinductive enhancements [32, 28], the starting notion is the one of (semi-)progression and enhancements are up-closures, which is a crucial difference from our inductive invariants.

Another approach, similar in spirit to [35], is based on the techniques of unique solutions of equations [13, 34]. However, to the best of our understanding, its applicability seems to be strongly tailored to equivalence relations.

Synopsis

We begin our exposition in Section 2 with a simple example of proof by strong induction for the Fibonacci sequence. We recall the lattice-theoretic understanding of induction and coinduction in Section 3 and the theory of coinductive up-to techniques in Section 4; its dual, the theory of inductive up-to techniques, is illustrated in Section 5. In particular, Section 5.1 links coinduction up-to and induction up-to by means of an involution operator; Example 12 illustrates how, following this link, the coinductive technique of up-to equivalence becomes *up-to apartness*. Section 6.2 introduces strong induction as a certain up-to technique. The fact that this generalises strong induction on $\mathbb{N}$ is not obvious and its proof is detailed in Section 6.1. Strong induction on words is discussed in Section 6.2. In Section 7, we turn from the proof principle to the definition principle: Section 7.1 quickly recalls the induction definition principle by means of initial algebras; Section 7.2 recalls comonadic corecursion and Section 7.3 course-of-value iteration, a special case of comonadic corecursion. Finally, Section 7.4, revisits the Fibonacci, by recalling that its definition is by means of course of value iteration. The appendix contains the missing proofs and some additional material.

Until Section 7, the reader only requires some familiarity with lattice theory. Then, we expect the reader to be familiar with category theory.

2 Motivating Example: the Fibonacci sequence

Induction is a proof principle that applies to inductively defined structures. For instance, for proving that a predicate $P(n)$ hold for all natural numbers $n \in \mathbb{N}$, one has to find a predicate $Q(n)$ that implies $P(n)$, that is true for 0 and that, when true for n, then it is true for $n+1$.

$$\frac{Q(0) \quad \forall n \in \mathbb{N}. Q(n) \Rightarrow Q(n+1) \qquad Q \Rightarrow P}{\forall n \in \mathbb{N}. P(n)} \tag{1}$$

Sometimes, induction might be too weak to prove certain properties. As an example, consider the Fibonacci sequence defined as

$$\mathsf{fib}(0) \stackrel{\text{def}}{=} 1 \qquad \mathsf{fib}(1) \stackrel{\text{def}}{=} 1 \qquad \mathsf{fib}(n+2) \stackrel{\text{def}}{=} \mathsf{fib}(n+1) + \mathsf{fib}(n),$$

and suppose that one would like to prove that $\mathsf{fib}(n) \geq n$ for all $n \in \mathbb{N}$. One could start a proof by induction by checking the base case, $\mathsf{fib}(0) = 1 \geq 0$. For the inductive case, one would need to bound $\mathsf{fib}(n+1) = \mathsf{fib}(n) + \mathsf{fib}(n-1)$. At this point, one would be stuck because there is no information on $\mathsf{fib}(n-1)$ from the inductive hypothesis.

Strong induction comes to our rescue by allowing a stronger inductive hypothesis. We still need to find a predicate $Q(n)$ that implies $P(n)$ and that holds at 0, but when showing that it is true for $n+1$, we may assume that it holds for all $k \leq n$.

$$\frac{Q(0) \qquad \forall n(\forall n' \in [0,n]\,.\,Q(n')) \Rightarrow Q(n+1) \qquad Q \Rightarrow P}{\forall n \in \mathbb{N}.P(n)} \tag{2}$$

We conclude the proof of $\mathsf{fib}(n) \geq n$ for all $n \in \mathbb{N}$ by strong induction. For $n = 0$, $\mathsf{fib}(0) = 1 \geq 0$. For the inductive step, we assume that $\mathsf{fib}(k) \geq k$ for all $k \leq n+1$, and we seek to bound $\mathsf{fib}(n+1) = \mathsf{fib}(n) + \mathsf{fib}(n-1)$. If $n = 1$ then $\mathsf{fib}(2) = \mathsf{fib}(1) + \mathsf{fib}(0) = 1+1 \geq 2$. Otherwise, if $n > 1$, we use the strong inductive hypothesis:

$$\mathsf{fib}(n+2) = \mathsf{fib}(n+1) + \mathsf{fib}(n) \geq n+1+n \geq n+2\ .$$

We want to draw the reader's attention to the fact that, here, strong induction is necessary because fib is not–strictly speaking–inductively defined: the value of $\mathsf{fib}(n+1)$ does not depend only on $\mathsf{fib}(n)$. We will revisit the relationship between definitions and proofs in Section 7. Until then, we will focus only on the proof principles.

3 Preliminaries and notation

A complete lattice is a partially ordered set $(L, \sqsubseteq)$ with joins ($\sqcup$), meets ($\sqcap$), a top ($\top$) and a bottom ($\bot$) elements, least upper bounds ($\bigsqcup$) and greatest lower bounds ($\bigsqcap$). Henceforth, we use $(L, \sqsubseteq), (L_1, \sqsubseteq_1), (L_2, \sqsubseteq_2)$ to range over complete lattices and x, y, z to range over their elements. One lattice that we will often use is $\mathcal{P}(X)$, the power set of a set X, ordered by set inclusion.

Recall that a function $f\colon L_1 \to L_2$ is said to be a *monotone map* if it preserves the order: for all $x, y \in L_1$, if $x \sqsubseteq_1 y$ then $f(x) \sqsubseteq_2 f(y)$. The identity $\mathsf{id}_L\colon L \to L$ and the composition $f \circ g\colon L_1 \to L_3$ of two monotone maps $g : L_1 \to L_2$ and $f\colon L_2 \to L_3$ are monotone. Therefore, if $f\colon L \to L$ is a monotone map, then its powers f^n are also monotone, where the functions $f^n\colon L \to L$ are defined inductively as

$$f^0 \stackrel{\text{def}}{=} \mathsf{id}_L \qquad\qquad f^{n+1} \stackrel{\text{def}}{=} f \circ f^n. \tag{3}$$

We will implicitly use the fact that monotone maps form a complete lattice with their natural point-wise order: whenever $f, g\colon L_1 \to L_2$ we write $f \sqsubseteq g$ iff $f(x) \sqsubseteq g(x)$ for all $x \in L_1$.

A monotone map $f\colon L \to L$ is an *up-closure* operator if $x \sqsubseteq f(x)$ and $ff(x) \sqsubseteq f(x)$. It is a *down-closure* operator if $f(x) \sqsubseteq x$ and $f(x) \sqsubseteq ff(x)$. Particularly relevant to our exposition are the up-closures and the down-closure generated by a (co)continuous map $f\colon L \to L$, namely a monotone map preserving arbitrary least upper bounds and greatest lower bounds:

$$f^{\uparrow} \stackrel{\text{def}}{=} \bigsqcup_{i \in \mathbb{N}} f^i \qquad\qquad f^{\downarrow} \stackrel{\text{def}}{=} \bigsqcap_{i \in \mathbb{N}} f^i\ . \tag{4}$$

Given a monotone map $f\colon L \to L$, the element $x \in L$ is said to be a *post-fixed point* iff $x \sqsubseteq f(x)$; a *pre-fixed point* iff $f(x) \sqsubseteq x$; a *fixed point* iff $x = f(x)$. We write μf and νf for the *least* and *greatest fixed point*. For a monotone map f on a complete lattice L, the Knaster-Tarski fixed point theorem characterises μf as the least upper bound of all pre-fixed points of f and μf as the greatest lower bound of all its post-fixed points:

$$\mu f = \bigsqcap\{x \mid f(x) \sqsubseteq x\} \qquad \nu f = \bigsqcup\{x \mid x \sqsubseteq f(x)\}.$$

This immediately leads to the *induction* and *coinduction proof principles*, illustrated by the inference rules below, on the left and on the right, respectively [26].

$$\frac{f(y) \sqsubseteq y \quad y \sqsubseteq x}{\mu f \sqsubseteq x} \qquad \frac{y \sqsubseteq f(y) \quad x \sqsubseteq y}{x \sqsubseteq \nu f} \tag{5}$$

The induction proof principle states that in order to prove that $\mu f \sqsubseteq x$, one should provide an *inductive invariant* –namely, a pre-fixed point of f– that is below x; dually, the coinduction proof principle states that in order to prove that $x \sqsubseteq \nu f$, one should provide a *coinductive invariant*, i.e., a post-fixed point of f, that is above x.

▶ Remark 1. From this lattice theoretic perspective, it is easy to see that the coinduction proof principle is simply the *dual* of induction. Indeed, whenever $(L, \sqsubseteq)$ is a lattice, then so is $(L, \sqsupseteq)$. Similarly, if $f\colon (L, \sqsubseteq) \to (L, \sqsubseteq)$ is monotone, then so is $f\colon (L, \sqsupseteq) \to (L, \sqsupseteq)$, and the greatest fixed point of f over $(L, \sqsubseteq)$ becomes the least fixed point of f over $(L, \sqsupseteq)$.

We illustrate inductive and coinductive invariants with an example from automata theory.

▶ **Example 2** (cf. [6, Remark 2])**.** We denote by A^* the set of words over an alphabet A; ϵ denotes the empty word and $u \cdot w$ the word obtained by concatenating $u \in A^*$ with $w \in A^*$. For a word $w \in A^*$, we indicate its length with $|w|$.

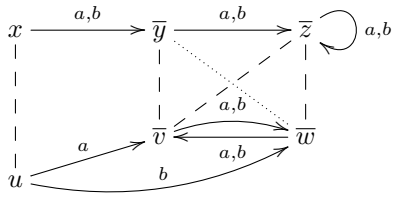
A deterministic automaton on the alphabet A is a triple (X, o, t), where X is a set of states, $o\colon X \to \{0,1\}$ is the output function, determining if a state x is accepting ($o(x) = 1$) or not ($o(x) = 0$) and $t\colon X \to X^A$ is the transition function which returns the next state, for each letter $a \in A$. Every automaton (X, o, t) induces a function $\mathsf{lan}_{\_}\colon X \to \{0,1\}^{A^*}$ defined inductively for all $x \in X$, $a \in A$ and $w \in A^*$ as $\mathsf{lan}_x(\varepsilon) = o(x)$ and $\mathsf{lan}_x(a \cdot w) = \mathsf{lan}_{t(x)(a)}(w)$. Two states $x, y \in X$ are said to be *language equivalent*, in symbols $x \sim y$, iff $\mathsf{lan}_x = \mathsf{lan}_y$.

Alternatively, $(\sim)$ can be defined as the greatest fixed point of some map on $\mathcal{P}(X \times X)$, the lattice of relations over X. The functions $l, q\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ are defined as

$$l(R) \stackrel{\text{def}}{=} \{(x,y) \mid \text{for all } a \in A,\ (t(x)(a), t(y)(a)) \in R\} \qquad q(R) \stackrel{\text{def}}{=} \{(x,y) \mid o(x) = o(y)\} \tag{6}$$

for all $R \subseteq X \times X$. One can easily check that both l and q are monotone and that $\nu(l \sqcap q) = (\sim)$. Thanks to this characterisation, one can prove that two states $x', y' \in X$ are language equivalent by means of the coinduction proof principle in (5): to show that $\{(x', y')\} \subseteq (\sim)$, it is enough to provide a relation R that is a post-fixed point of $l \sqcap q$ and such that $\{(x', y')\} \subseteq R$. Such coinductive invariants are often called *bisimulations*.

For an example, consider the following deterministic automaton, where final states are overlined and the transition function is represented by labelled arrows. The relation consisting of dashed and dotted lines is a bisimulation witnessing that $\{(x, u)\} \subseteq (\sim)$.



(7)

One can prove that $\{(x', y')\} \subseteq (\sim)$ by means of induction as well: for all $R \subseteq X \times X$, the functions $l^\dagger, p\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ are defined as follows.

$$l^\dagger(R) \stackrel{\text{def}}{=} \{(t(x)(a), t(y)(a)) \mid a \in A, (x,y) \in R\} \qquad p(R) \stackrel{\text{def}}{=} \{(x', y')\} \tag{8}$$

Note that p above, as well as q in (6), are constant functions: we will sometime take the freedom to identify them with the corresponding element in the lattice. Intuitively, $\mu(l^\dagger \sqcup p)$ represents the subset of all pairs of states that are reachable from the pair (x', y'). Thus, $\{(x', y')\} \subseteq (\sim)$ if and only if all those pairs of states are in q, i.e., if and only if $\mu(l^\dagger \sqcup p) \subseteq q$. The latter can be proved by exhibiting a relation R that is a pre-fixed point of $l^\dagger \sqcup p$ and such that $R \subseteq q$: the relation formed by the dashed and dotted lines in (7) satisfies this condition when taking (x', y') to be (x, u).

4 Coinduction up-to

Coinduction is a technique for proving $x \sqsubseteq \nu f$ for some map f on a lattice $(L, \sqsubseteq)$ by providing a coinductive invariant for f. In many situations, providing such an invariant is far too complicated. Motivated by this fact, Milner [23] introduced enhancements of the coinduction proof principle which are nowadays widely known as up-to techniques. In a nutshell, an *up-to technique* is an up-closure $d\colon (L, \sqsubseteq) \to (L, \sqsubseteq)$. An *$f$-coinductive invariant up-to d* is some $y \in L$ such that $y \sqsubseteq fd(y)$, namely a post-fixed point of fd. An up-to technique d is said to be *sound* w.r.t. f if the following *coinduction up-to principle* holds.

$$\frac{y \sqsubseteq f(d(y)) \quad x \sqsubseteq y}{x \sqsubseteq \nu f} \qquad \text{(Coinduction Up-To)}$$

In (5), one has to find an invariant y such that $y \sqsubseteq f(y)$. In (Coinduction Up-To), the search of such a y is simplified since it is enough that $y \sqsubseteq f(d(y))$. Since d is an up-closure, $f(y) \sqsubseteq f(d(y))$, which simplifies the task of finding coinductive invariants.

▶ **Example 3** (Up-to equivalence)**.** We continue Example 2 to illustrate a coinductive invariant up-to. We instantiate (Coinduction Up-To) by taking f to be $l \sqcap q$ and d to be the function $e\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ mapping any relation $R \subseteq X \times X$ into its equivalence closure. One can check $(\sim)$ by exhibiting a relation R such that $R \subseteq l \sqcap q(e(R))$.

Consider for instance the relation S consisting of only the dashed lines in (7). Note that $(y, w) \in e(S)$ but $(y, w) \notin S$. It is thus easy to see that $S \subseteq l \sqcap q(e(S))$, but S is not included in $l \sqcap q(S)$. In other words, S is a coinductive invariant up-to e but not a coinductive invariant. In Example 2, to prove that $x \sim u$ by means of coinduction, we need to take the relation consisting of both dashed and dotted lines in (7). With coinduction up-to e, it is thus enough to take only the dashed lines.

Of course, before using an up-to technique, one should prove it to be sound. Since this might be quite challenging, several theories [32, 28, 10, 29, 4] have been introduced for simplifying this task. In this paper we will consider the theory of Pous in [28] that focuses on the notion of compatible techniques: d is *f-compatible* iff $df \sqsubseteq fd$. The key results in [28] state that compatible techniques are sound and can be nicely composed.

5 Induction up-to

Recall that for applying the induction proof principle in (5), one has to find a $y \in L$ such that $f(y) \sqsubseteq y$. The idea of induction up-to is to simplify this task by weakening such constraint to $f(d(y)) \sqsubseteq y$ for some $d\colon L \to L$.

Note that if the map d is an up-closure, as it is the case of coinduction up-to, then this would only complicate our task by imposing additional constraints; indeed $f(y) \sqsubseteq f(d(y))$.

▶ **Example 4.** Recall from Example 2 that the relation R consisting of both dashed and dotted lines in (7) is an inductive invariant, i.e., $(l^\dagger \sqcup p)(R) \subseteq R$. Note that, instead $l^\dagger \sqcup p(e(R))$ is not included into R since, e.g., (v, w) is in $l^\dagger \sqcup p(e(R))$ but not in R.

As expected, the solution consists in considering down closures. An *(inductive) up-to technique* is a down closure $d\colon (L, \sqsubseteq) \to (L, \sqsubseteq)$. An *$f$-inductive invariant up-to d* is some $y \in L$ such that $fd(y) \sqsubseteq y$, namely a pre-fixed point of fd. An up-to technique d is said to be *sound* w.r.t. f if the following *induction up-to principle* holds.

$$\frac{f(d(y)) \sqsubseteq y \quad y \sqsubseteq x}{\mu f \sqsubseteq x} \qquad \text{(Induction Up-To)}$$

To prove the soundness of inductive up-to techniques, we consider the dual of the notion of *compatible functions* from [28].

▶ **Definition 5** (Cocompatible map). *Let $f, d\colon (X, \sqsubseteq) \to (X, \sqsubseteq)$ be two monotone maps. We say that d is* cocompatible with *f, shortly f-cocompatible, if $fd \sqsubseteq df$.*

When d is f-cocompatible, any inductive invariant up-to gives rise to an inductive invariant.

▶ **Proposition 6.** *Let d be a down closure that is cocompatible with some monotone map f. If y is a pre-fixed point for fd, then $d(y)$ is a pre-fixed point for f.*

Proof.

$$\begin{aligned} fd(y) &\sqsubseteq fdd(y) && (d \text{ down closure}) \\ &\sqsubseteq dfd(y) && (d \text{ is } f\text{-cocompatible}) \\ &\sqsubseteq d(y) && (y \text{ is a pre-fixed point of } fd) \end{aligned}$$

◀

▶ **Theorem 7.** *If d is f-cocompatible, then it is sound.*

Proof. We have to prove the conclusion of (Induction Up-To) assuming its premise. By $fd(y) \sqsubseteq y$ and Proposition 6, it holds that

$$fd(y) \sqsubseteq d(y).$$

Since $d(y) \sqsubseteq y$, as d is a downclosure, and $y \sqsubseteq x$ it holds that

$$d(y) \sqsubseteq x.$$

Thus by replacing y with $d(y)$ in (5), it holds that $\mu f \sqsubseteq x$. ◀

It is worth mentioning that the two results above also hold by dualising the theory in [28]. We have reported their proofs since they will be relevant in Section 7. The following result also follows easily from [28]. For convenience of the reader we report its proof in Appendix A.

▶ **Proposition 8** (The algebra of cocompatible maps). *Let $f, d, e\colon (X, \sqsubseteq) \to (X, \sqsubseteq)$ be monotone maps. Let $\{d_i\}_{i \in \mathbb{N}}$ be an $\mathbb{N}$-indexed family of monotone maps.*

1. *The identity id_X is f-cocompatible;*
2. *f is f-cocompatible;*
3. *If d and e are f-cocompatible, then $d \circ e$ is f-cocompatible;*
4. *If d is f-cocompatible then, for all $n \in \mathbb{N}$, d^n is f-cocompatible;*

5. *If d and e are f-cocompatible, then $d \sqcap e$ is f-cocompatible;*
6. *If, for all $i \in \mathbb{N}$, d_i is f-cocompatible, then $\bigsqcap_{i\in\mathbb{N}} d_i$ is f-cocompatible;*
7. *If d is f-cocompatible, then $d^{\downarrow}$ is f-cocompatible.*

▶ **Corollary 9.** *Let $f\colon (X,\sqsubseteq) \to (X,\sqsubseteq)$ be a continuous monotone map. Then, its down-closure $f^{\downarrow}$ is f-cocompatible.*

Proof. By point 2 and 7 in Proposition 8. ◀

▶ Remark 10. Note that we have defined up-to techniques to be down-closures, while compatible maps are defined as arbitrary monotone maps. This choice is justified by the fact that monotone maps compose nicely, while down-closures do not. This motivated Pous to introduce up-to techniques in the original theory in [28] as monotone maps rather than up-closures. Here, we preferred to stay with closures, as this simplifies the proofs of Proposition 6 and Theorem 7, which will be relevant in the categorical generalisation illustrated in Section 7.

Note also that restricting to down-closures does not limit the applicability of the theory: indeed, if d is an f-compatible monotone map, not necessarily a down-closure, then, by Proposition 8.7, $d^{\downarrow}$ is also f-compatible. Moreover, if $fd(y) \sqsubseteq x$, then

$$fd^{\downarrow}(y) \sqsubseteq fd(y) \sqsubseteq x$$

since $d^{\downarrow} \sqsubseteq d$. In other words, any proof up-to d is also a proof up-to $d^{\downarrow}$.

5.1 Relating Coinduction up-to and Induction Up-to via Involution

Coinduction and induction are equivalent whenever the lattice $(L,\sqsubseteq)$ comes with an involution operator $\neg\colon (L,\sqsubseteq) \to (L,\sqsupseteq)$, namely a function on L such that

$$\text{if } x \sqsubseteq y, \text{ then } \neg x \sqsupseteq \neg y \qquad \neg\neg x = x \tag{9}$$

for all $x, y \in L$. In this case, for any monotone map f on $(L,\sqsubseteq)$, one has that $\overline{f} \stackrel{\text{def}}{=} \neg f \neg$ is a monotone map. Moreover, assuming that f preserves $\bigsqcap$ of ω-chains, it holds that:

$$x \sqsubseteq \nu f \Leftrightarrow \neg(\nu f) \sqsubseteq \neg x \Leftrightarrow \mu\overline{f} \sqsubseteq \neg x.$$

Such correspondence lifts to up-to techniques: whenever one can prove the leftmost by coinduction up-to d, for some f-compatible technique d, one can equivalently prove the rightmost by induction up-to to $\overline{d}$. This is made formal by the following result.

▶ **Proposition 11.** *Let $f, d\colon (L,\sqsubseteq) \to (L,\sqsubseteq)$ be monotone maps and y be an element of L.*
1. *d is an up-closure iff $\overline{d}$ is a down-closure;*
2. *d is f-compatible iff $\overline{d}$ is $\overline{f}$-cocompatible;*
3. *y is an f-coinductive invariant up-to d iff $\neg y$ is an $\overline{f}$-inductive invariant up-to $\overline{d}$.*

▶ **Example 12** (Up-to apartness)**.** Following the above considerations, one can transform coinduction up-to equivalence in Example 3 into *induction up-to apartnesses*. Apartness relations are standard in constructive reals analysis and has been first axiomatised in [19]: $R \subseteq X \times X$ is a an *apartness relation* if it is

- irreflexive: $(x,x) \notin R$ for all $x \in X$;
- symmetric: if $(x,y) \in R$, then $(y,x) \in R$ for all $x, y \in X$;
- co-transitive: if $(x,y) \in R$, then $(x,z) \in R$ or $(z,y) \in R$, for all $x, y, z \in X$.

The reader can easily check that R is an apartness relation iff $\neg R$ is an equivalence relation, where $\neg$ indicates the complement of a relation. Recall from Example 3 that the up-closure $e\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ mapping any relation R into its equivalence closure. By Proposition 11.1, $\overline{e}$ is a down closure: it maps any R into the largest apartnesses relation contained in R. Since e is $(l \sqcap q)$-compatible, by Proposition 11.2, $\overline{e}$ is $(\overline{l \sqcap q})$-cocompatible. Since $\mathcal{P}(X \times X)$ is a boolean algebra, then $\overline{l \sqcap q} = \overline{l} \sqcup \overline{q}$; it is easy to check that $\overline{l}$ and $\overline{q}$ map any $R \subseteq X \times X$ into the following relations.

$$\overline{l}(R) = \{(x, y) \mid \text{exists } a \in A,\ (t(x)(a), t(y)(a)) \in R\} \qquad \overline{q}(R) = \{(x, y) \mid o(x) \neq o(y)\}.$$

With this characterisation, one can see that inductive invariants for $\overline{l} \sqcup \overline{q}$ are exactly those introduced [15, Definition 2.2]. Our work enhances this proof method with up-to apartness.

For an example of an inductive invariant up-to apartness, consider again the relation S consisting of the dashed lines in (7). Since S is a $(l \sqcap q)$-coinductive invariant up to e, then by Proposition 11.3, $\neg S$ is a $(\overline{l} \sqcup \overline{q})$-inductive invariant up-to $\overline{e}$.

6 Strong Induction is an up-to technique

This section studies the proof principle given by a particular f-cocompatible map: the down-closure of f. Indeed, by Corollary 9, $f^{\downarrow}$ is f-compatible and, by Theorem 7, the following proof principle is always sound.

$$\frac{f(f^{\downarrow}(y)) \sqsubseteq y \quad y \sqsubseteq x}{\mu f \sqsubseteq x} \qquad \text{(Strong Induction)}$$

We call such principle *strong induction*. Indeed, as we illustrate below, when instantiated to usual induction on natural numbers, the above proof principle coincides with the well-known strong induction illustrated in Section 2.

6.1 Strong Induction on natural numbers

We begin by illustrating how (5) generalises standard induction over natural numbers. Consider the lattice $\mathcal{P}(\mathbb{N})$ and the monotone map $b\colon \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$ defined as

$$b(X) \stackrel{\text{def}}{=} \{0\} \cup \{x + 1 \mid x \in X\} \tag{10}$$

for all $X \in \mathcal{P}(\mathbb{N})$. The least fixed point of b is the set of natural numbers, $\mu b = \mathbb{N}$. We take the sets X and Y to be the sets of natural numbers on which $P(n)$ and $Q(n)$ are true, respectively.

$$X = \{n \in \mathbb{N} \mid P(n)\} \qquad Y = \{n \in \mathbb{N} \mid Q(n)\}$$

With these choices, set inclusion corresponds to predicate implication: $Y \subseteq X$ iff $Q \Rightarrow P$. The least fix point of b is contained in X iff all natural numbers are contained in X, which means that $P(n)$ holds for all $n \in \mathbb{N}$.

$$\mu b \subseteq X \quad \text{iff} \quad \mathbb{N} \subseteq X \quad \text{iff} \quad \forall n \in \mathbb{N}. P(n)$$

Similarly, Y is a pre-fixed point of b iff Y contains 0 and it contains $n + 1$ for each $n \in Y$, which means that Q holds at 0 and it holds at $n + 1$ whenever it holds at n.

$$b(Y) \subseteq Y \quad \text{iff} \quad \{0\} \cup \{n + 1 \mid n \in Y\} \subseteq Y \quad \text{iff} \quad Q(0) \text{ and } \forall n \in \mathbb{N}. Q(n) \Rightarrow Q(n + 1)$$

These considerations show that (1) is a particular instance of (5), when we instantiate it to $b\colon \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$.

$$\frac{b(Y) \subseteq Y \quad Y \subseteq X}{\mu b \subseteq X} \qquad \text{iff} \qquad \frac{Q(0) \quad \forall n \in \mathbb{N}. Q(n) \Rightarrow Q(n+1) \qquad Q \Rightarrow P}{\forall n \in \mathbb{N}. P(n)}$$

We can now illustrate our main observation: when instantiating (Strong Induction) to b one obtains exactly the strong induction on natural numbers reported in (2). We start by computing the powers b^n of b.

▶ **Lemma 13.** *For all $n \in \mathbb{N}$, and all $X \in \mathcal{P}(\mathbb{N})$,*

$$b^n(X) = \{x \in \mathbb{N} \mid x < n\} \cup \{x + n \mid x \in X\}.$$

Proof. By induction. For the base case, we have the following derivation.

$$\begin{aligned} b^0(X) &= id_{\mathcal{P}(\mathbb{N})}(X) && (3)\\ &= X \\ &= \emptyset \cup X \\ &= \{x \in \mathbb{N} \mid x < 0\} \cup \{x + 0 \mid x \in X\} \end{aligned}$$

For the inductive case, we have the following derivation.

$$\begin{aligned} b^{n+1}(X) &= bb^n(X) && (3)\\ &= b(\,\{x \in \mathbb{N} \mid x < n\} \cup \{x + n \mid x \in X\}\,) && (\text{Ind. Hyp.})\\ &= \{0\} \cup \{x + 1 \mid x < n\} \cup \{x + n + 1 \mid x \in X\} && (10)\\ &= \{0\} \cup [1, n] \cup \{x + n + 1 \mid x \in X\} \\ &= \{x \in \mathbb{N} \mid x < n + 1\} \cup \{x + n + 1 \mid x \in X\} \end{aligned}$$ ◀

The core of our argument relies on the following result, stating that $b^{\downarrow}(X)$ is the largest closed interval from including 0 that is a subset of X.

▶ **Lemma 14.** *For any set $X \in \mathcal{P}(\mathbb{N})$, $b^{\downarrow}(X)$ is characterised as $b^{\downarrow}(X) = \{x \mid [0, x] \subseteq X\}$.*

Proof. By definition $b^{\downarrow} = \bigsqcap_{n=0}^{\infty} b^n$. Thus,

$$\begin{aligned} m \in b^{\downarrow}(X) &\Leftrightarrow \forall n \in \mathbb{N}.\, m \in b^n(X) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, (m \in \{x \in| \; x < n\} \vee m \in \{x + n \mid x \in X\}) && (\text{Lemma 13})\\ &\Leftrightarrow \forall n \in \mathbb{N}.\, (m < n \vee m \in \{x + n \mid x \in X\}) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, (\neg(m \geq n) \vee m \in \{x + n \mid x \in X\}) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, ((m \geq n) \Rightarrow m \in \{x + n \mid x \in X\}) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, ((n \leq m) \Rightarrow \exists x \in X.\, m = x + n) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, ((n \leq m) \Rightarrow \exists x \in X.\, x = m - n) \\ &\Leftrightarrow \forall n \in \mathbb{N}.\, ((n \leq m) \Rightarrow (m - n) \in X) \end{aligned}$$

In short,

$$m \in b^{\downarrow}(X) \Leftrightarrow \forall n \in \mathbb{N}.((n \leq m) \Rightarrow (m - n) \in X) \qquad (11)$$

We use (11) to prove the two inclusions of $b^{\downarrow}(X) = \{x \mid [0, x] \subseteq X\}$ separetely:

- $b^{\downarrow}(X) \subseteq \{x \mid [0,x] \subseteq X\}$. We assume that $m \in b^{\downarrow}(X)$ and we need to prove that $[0,m] \subseteq X$. Let us take an arbitrary $y \in [0,m]$. Since by (11), $(m-n) \in X$ for all $n \leq m$, then one can take n to be $m-y$ and have that $m-(m-y) \in X$, that is, $y \in X$.
- $b^{\downarrow}(X) \supseteq \{x \mid [0,x] \subseteq X\}$. We assume that $[0,m] \subseteq X$. Thus $\forall n \in \mathbb{N}.\,((n \leq m) \Rightarrow (m-n) \in X)$ that, by (11), means that $m \in b^{\downarrow}(X)$. ◀

▶ **Proposition 15.** *For any set $Y \in \mathcal{P}(\mathbb{N})$, the following are equivalent*

- $bb^{\downarrow}(Y) \subseteq Y$;
- $0 \in X$ *and* $(\forall n \in \mathbb{N}\,.\,[0,n] \subseteq Y \Rightarrow n+1 \in Y)$.

Proof.

$$
\begin{aligned}
b\,b^{\downarrow}(Y) \subseteq Y &\Leftrightarrow \{0\} \cup \{y+1 \mid y \in b^{\downarrow}(Y)\} \subseteq Y && (10)\\
&\Leftrightarrow 0 \in Y \text{ and } \{y+1 \mid y \in b^{\downarrow}(Y)\} \subseteq Y\\
&\Leftrightarrow 0 \in Y \text{ and } (\{y+1 \mid y \in \{n \in \mathbb{N} \mid [0,n] \subseteq Y\}\} \subseteq Y) && (\text{Lemma } 14)\\
&\Leftrightarrow 0 \in Y \text{ and } (\forall n \in \mathbb{N}\,.\,[0,n] \subseteq Y \Rightarrow n+1 \in Y)
\end{aligned}
$$

◀

The above proposition allows us to easily see that strong induction (2) is induction up-to $b^{\downarrow}$. The latter is illustrated below on the left. The former is reported on the right.

$$
\frac{bb^{\downarrow}(Y) \subseteq Y \quad Y \subseteq X}{\mu b \subseteq X} \quad \text{iff} \quad \frac{Q(0) \qquad \forall n(\forall n' \in [0,n]\,.\,Q(n')) \Rightarrow Q(n+1) \qquad Q \Rightarrow P}{\forall n \in \mathbb{N}.P(n)}
$$

The correspondence between the two rules mirrors that of induction. The conclusions of the two rules coincide in the same way that they did for induction. For the premise, observe that, by Proposition 15,

$$
b\,b^{\downarrow}(Y) \subseteq Y \qquad \text{iff} \qquad Q(0) \wedge (\,\forall n(\forall n' \in [0,n]\,.\,Q(n')) \Rightarrow Q(n+1)\,).
$$

6.2 Strong Induction on Words

As expected, one can use strong induction not only on $\mathbb{N}$ but on any inductive data type. Below, we illustrate strong induction on A^*, the set of words over an alphabet A.

As we did for natural numbers, we need to give a monotone map $c\colon \mathcal{P}(A^*) \to \mathcal{P}(A^*)$ that gives induction on words, i.e. whose least fixed point is A^*. The candidate monotone map c mimics the definition of the monotone map b for natural numbers: it maps a set X to the set containing the empty word and all the successors of words in X.

$$
c(X) \stackrel{\text{def}}{=} \{\epsilon\} \cup \{a \cdot w \mid w \in X,\ a \in A\} \tag{12}
$$

Since the least fixed point of c is A^*, the induction principle (5) instantiated to c give us the usual induction principle on words.

$$
\frac{c(Y) \subseteq Y \quad Y \subseteq X}{\mu c \subseteq X} \quad \text{iff} \quad \frac{Q(\epsilon) \qquad \forall a \in A.\forall w \in A^*.Q(w) \Rightarrow Q(a \cdot w) \qquad Q \Rightarrow P}{\forall w \in A^*.P(w)}
$$

We now turn our attention to (Strong Induction) instantiated with c:

$$
\frac{cc^{\downarrow}(Y) \subseteq Y \quad Y \subseteq X}{\mu c \subseteq X}
$$

What does this means in practice? To answer this question, the key is to have a handy characterisation of $c^{\downarrow}$. This is going to resemble that of $b^{\downarrow}$ but, instead of the ordering on natural numbers, we consider the suffix partial ordering of words:

$$v \sqsubseteq_{A^*} w \quad \text{iff} \quad \exists u \in A^*.\, w = u \cdot v \ .$$

The analogue of the interval $[0, n]$ for natural numbers is, then, the set of suffixes of a word, $\mathsf{Suf}(w) \stackrel{\text{def}}{=} \{u \in A^* \mid u \sqsubseteq w\}$. With this, we obtain that the down closure of c gives the biggest subset that is closed under suffixes.

$$c^{\downarrow}(X) = \{x \in A^* \mid \mathsf{Suf}(x) \subseteq X\}$$

With this result, we can explicit the strong induction principle on words.

$$\frac{Q(\epsilon) \qquad \forall a \in A. \forall w \in A^*. (\forall y \in \mathsf{Suf}(w)\,.\, Q(y)) \Rightarrow Q(a \cdot w) \qquad Q \Rightarrow P}{\forall w \in A^*. P(w)}$$

Compare this principle with strong induction on natural numbers: consider the singleton set $A = \{*\}$. Then, words on A are determined by their length, so A^* is in bijection with the natural numbers $\mathbb{N}$. Through this bijection, $\mathsf{Suf}(w)$ coincides with the interval $[0, |w|]$. This further justifies the name of strong induction.

7 From Lattice to Categories

Induction is both a proof principle and a *definition principle*. The latter can be obtained as generalisation of the former by moving from lattices to categories. As (inductive) up-to techniques are enhancements of the induction proof principle, by means of a similar generalisation, one can obtain enhancements of the induction definition principle. In this section, we illustrate that induction up-to generalises to a recursion scheme known as *comonadic recursion* [8] and strong induction generalises to *course-of-value iteration* [38, 7].

7.1 Initial agebras and the induction definition principle

Hereafter, we write $\times$ and $+$ for products and coproducts in some category $\mathbf{C}$, $\langle a, b\rangle \colon X \to Y \times Z$ for the pairing of $a \colon X \to Y$ and $b \colon X \to Z$ and $[c, d] \colon Y + Z \to X$ for the copairing of $c \colon Y \to X$ and $d \colon Z \to X$. The singleton set $\{\epsilon\}$ is denoted by 1.

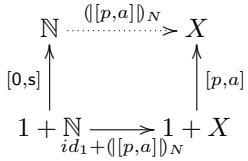
Given a functor $F \colon \mathbf{C} \to \mathbf{C}$ on some category $\mathbf{C}$, an *F-algebra* is a pair (X, a) where X is an object of $\mathbf{C}$ and $a \colon FX \to X$ is an arrow. Given two F-algebras (X, a) and (Y, b), an algebra morphism $h \colon (X, a) \to (Y, b)$ is an arrow $h \colon X \to Y$ of $\mathbf{C}$ making the diagram below commute. An F-algebra $(\mu F, \mathsf{i})$ is said to be *initial* if for any F-algebra (X, a), there exists a unique algebra morphism $(\!|a|\!)_F \colon (\mu F, \mathsf{i}) \to (X, a)$.

$$\begin{array}{ccc} X & \xrightarrow{h} & Y \\ {\scriptstyle a}\uparrow & & \uparrow{\scriptstyle b} \\ FX & \xrightarrow[Fh]{} & FY \end{array}$$

Initial algebras give the *induction definition principle*: in order to specify a morphism from μF to some object X it is enough to give an F-algebra on X.

▶ Remark 16. When the category $\mathbf{C}$ is a complete lattice, a functor F on $\mathbf{C}$ is simply a monotone map; an F-algebra is a pre-fixed point for F and an initial F-algebra is a least fixed-point. In this perspective, the induction definition principle collapses to the induction proof principle: specifying an arrow $\mu F \to X$ means exactly proving that $\mu F \sqsubseteq X$.

Consider **Set** –the category of sets and functions– and $N\colon \mathbf{Set} \to \mathbf{Set}$ the functor mapping a set X into $1 + X$ and a function a into $id_1 + a$. An initial algebra for N is provided by $(\mathbb{N}, [0, \mathsf{s}])$ where $0\colon 1 \to \mathbb{N}$ assigns to ϵ the number 0 and $\mathsf{s}\colon \mathbb{N} \to \mathbb{N}$ assigns to any $n \in \mathbb{N}$, its successor $n + 1$. Now, given an F-algebra $[p, a]\colon 1 + X \to X$, one obtains, by initiality, a function $(\!|[p, a]|\!)_N\colon \mathbb{N} \to X$, as illustrated below on the left.



$$\begin{aligned}(\!|[p,a]|\!)_N(0) &= p(\epsilon)\\ (\!|[p,a]|\!)_N(n+1) &= a((\!|[p,a]|\!)_N(n))\end{aligned}$$

The fact that the diagram on the left commutes is expressed by the conditions on the right. The first condition provides the base case of an inductive definition; the second one provides the inductive case. Note that, in order to define $(\!|[p, a]|\!)_N(n + 1)$, one uses the value $(\!|[p, a]|\!)_N(n)$. Sometimes, as in the Fibonacci sequence in Section 2, functions are specified by using not just their value at n but also their values at some smaller numbers. This can be done by enhancing the induction definition principle by means of recursion schemes.

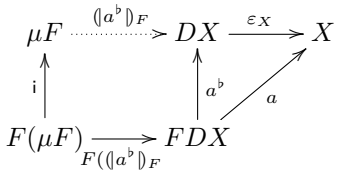
7.2 Comonadic Recursion

A *comonad* on a category $\mathbf{C}$ is a functor $D\colon \mathbf{C} \to \mathbf{C}$ together with two natural transformations, the counit $\varepsilon\colon D \Rightarrow Id$ and the comultiplication $\delta\colon D \Rightarrow DD$, such that $\varepsilon_{DX} \circ \delta_X = id_{DX} = D\varepsilon_X \circ \delta_X$ and $D\delta_X \circ \delta_X = \delta_{DX} \circ \delta_X$ for all objects X. A *distributive law* of a functor $F\colon \mathbf{C} \to \mathbf{C}$ over the comonad D is a natural transformation $\zeta\colon FD \Rightarrow DF$ such that $\varepsilon_{FX} \circ \zeta_X = F\varepsilon_X$ and $\delta_{FX} \circ \zeta_X = D\zeta_X \circ \zeta_{DX} \circ F\delta_X$.

Comonadic recursion exploits a comonad D and a distributive law $\zeta\colon FD \Rightarrow DF$ to enhance the induction definition principle. In order to define a morphism from μF to X, rather than specifying an F-algebra, one can specify an FD-algebra $a\colon FDX \to X$. Indeed, such a gives rise to

$$a^\flat \overset{\text{def}}{=} FDX \xrightarrow{F\delta_X} FDDX \xrightarrow{\zeta_{DX}} DFDX \xrightarrow{Da} DX$$

which is an F-algebra and thus, by initiality of μF, one obtains $(\!|a^\flat|\!)_F\colon \mu F \to DX$ that can be composed with the counit $\varepsilon_X\colon DX \to X$ to obtain the desired morphism from μF to X.



▶ Remark 17. Following Remark 16, when $\mathbf{C}$ is a complete lattice, a comonad D is simply a down-closure; a distributive law $\zeta\colon FD \Rightarrow DF$ witnesses that $FD \sqsubseteq DF$, namely that D is cocompatible with F. The algebra $a\colon FDX \to X$ is just an inductive invariant up-to D and $a^\flat\colon FDX \to DX$ is the corresponding inductive invariant provided by Proposition 6: observe that the three arrows in the definition of $a^\flat$ are exactly the three steps in the proof of Proposition 6. The morphism of $\varepsilon_X \circ (\!|a^\flat|\!)_F\colon \mu F \to X$ gives us the proof of Theorem 7. All this justifies the following statement.

▶ **Proposition 18.** *When* $\mathbf{C}$ *is a complete lattice, comonadic recursion is induction up-to.*

7.3 Course-of-Value Iteration

Course-of-value iteration is a recursion scheme that is obtained from comonadic recursion by taking D to be the *cofree comonad* generated by F. Below, we shortly recall this.

Coalgebras are the dual of algebras: a *coalgebra* for a functor $F\colon \mathbf{C} \to \mathbf{C}$ is a pair (X, a) with $a\colon X \to FX$. Given two F-coalgebras (X, a) and (Y, b), a coalgebra morphism $h\colon (X, a) \to (Y, b)$ is an arrow $h\colon X \to Y$ of $\mathbf{C}$ such that $Fh \circ a = b \circ h$. An F-coalgebra $(\nu F, \mathsf{a})$ is said to be *final* if for any F-coalgebra (X, a), there exists a unique coalgebra morphism $[\![a]\!]_F\colon (X, a) \to (\nu F, \mathsf{a})$.

For an object A of $\mathbf{C}$, we denote with $F_A\colon \mathbf{C} \to \mathbf{C}$ the functor mapping an object X into $FX \times A$ and arrow a into $Ff \times id_A$. Whenever F_A has a final coalgebra $\langle \mathsf{t}_A, \mathsf{o}_A \rangle\colon \nu F_A \to F(\nu F_A) \times A$ for all objects A, one can define a comonad $(F^{\downarrow}, \varepsilon^{\downarrow}, \delta^{\downarrow})$ on $\mathbf{C}$. The functor $F^{\downarrow}\colon \mathbf{C} \to \mathbf{C}$ maps an object X into the final coalgebra νF_X and an arrow $a\colon X \to Y$ into the unique final coalgebra morphism $[\![\langle \mathsf{t}_X, a \circ \mathsf{o}_X \rangle]\!]_{F_Y} : \nu F_X \to \nu F_Y$. For all objects X, the counit $\varepsilon^{\downarrow}\colon F^{\downarrow} \Rightarrow Id$ is defined as $\mathsf{o}_X\colon \nu F_X \to X$; the comultiplication $\delta^{\downarrow}\colon F^{\downarrow} \Rightarrow F^{\downarrow}F^{\downarrow}$ as $[\![\langle \mathsf{t}_X, id_{\nu F_X} \rangle]\!]_{F_{\nu F_X}} : \nu F_X \to \nu F_{\nu F_X}$.

Crucially, there exists a distributive law of F over $F^{\downarrow}$, denoted by $\zeta^{\downarrow}\colon FF^{\downarrow} \Rightarrow F^{\downarrow}F$, that is defined for all objects X as $[\![\langle F\pi_1, F\pi_2 \rangle \circ F\langle \mathsf{t}_X, \mathsf{o}_X \rangle]\!]_{FFX} : F(\nu F_X) \to \nu F_{FX}$.

▶ Remark 19. Following Remarks 16 and 17, when $\mathbf{C}$ is a complete lattice, an F-coalgebra is a post-fixed point for F and a final F-algebra is a greatest fixed-point. The cofree comonad $F^{\downarrow}$ simplifies to the down-closure generated by the monotone map F, as defined in (4). The condition on the existence of a final F_A-coalgebra for all objects A trivially holds when $\mathbf{C}$ is a lattice. The existence of the distributive law $\zeta^{\downarrow}\colon FF^{\downarrow} \Rightarrow F^{\downarrow}F$ simplifies to Corollary 9. All this justifies the following statement.

▶ **Proposition 20.** *When* $\mathbf{C}$ *is a complete lattice, course of value iteration is strong induction.*

7.4 Back to Fibonacci

We conclude by illustrating course of value iteration in the case when F is the functor N introduced in Section 7.1. First, it is convenient to recall some auxiliary ingredients.

For a set A, we write A^{∞}_{ne} for the set of all finite and infinite sequences over A that are non-empty. For a sequence $\sigma \in A^{\infty}_{ne}$, we write $\sigma(0) \cdot \sigma(1) \cdot \ldots$ whenever σ is infinite and $\sigma(0) \cdot \sigma(1) \cdot \ldots \sigma(n) \cdot \epsilon$ whenever σ has length $n+1$. Appending ϵ at the end of the word is a convenient notation to avoid confusing an element $a \in A$ with the sequence $a \cdot \epsilon \in A^{\infty}_{ne}$. For all $\sigma \in A^{\infty}_{ne}$, we define $\mathsf{hd}\colon A^{\infty}_{ne} \to A$ and $\mathsf{tl}\colon A^{\infty}_{ne} \to 1 + A^{\infty}_{ne}$ as follows.

$$\mathsf{hd}(\sigma) \stackrel{\text{def}}{=} \sigma(0) \qquad \mathsf{tl}(\sigma) \stackrel{\text{def}}{=} \begin{cases} \sigma(1) \cdot \sigma(2) \cdot \cdots \cdot \sigma(n) \cdot \epsilon & \text{if } \sigma \text{ has length } n+1 \\ \sigma(1) \cdot \sigma(2) \cdot \cdots & \text{otherwise} \end{cases}$$

The pairing $\langle \mathsf{tl}, \mathsf{hd} \rangle\colon A^{\infty}_{ne} \to (1 + A^{\infty}_{ne}) \times A$ forms a coalgebra for the functor $N_A\colon \mathbf{Set} \to \mathbf{Set}$. Actually, $(A^{\infty}_{ne}, \langle \mathsf{tl}, \mathsf{hd} \rangle)$ is a final coalgebra for N_A: for all N_A-coalgebra $\langle t, o \rangle\colon X \to (1+X) \times A$ there exists a unique morphism making the following diagram on the left commute.

$$\begin{array}{ccc} X & \xrightarrow{\quad [\![\langle t,o \rangle]\!]_{N_A} \quad} & A^{\infty}_{ne} \\ {\scriptstyle \langle t,o \rangle}\downarrow & & \downarrow{\scriptstyle \langle \mathsf{tl},\mathsf{hd} \rangle} \\ (1+X) \times A & \xrightarrow[(id_1 + [\![\langle t,o \rangle]\!]_{N_A}) \times id_A]{} & (1 + A^{\infty}_{ne}) \times A \end{array} \qquad \begin{array}{rcl} \mathsf{hd}(\,[\![\langle t,o \rangle]\!]_{N_A}(\sigma)\,) & = & o(\sigma) \\ \mathsf{tl}(\,[\![\langle t,o \rangle]\!]_{N_A}(\sigma)\,) & = & [\![\langle t,o \rangle]\!]_{N_A}(\,t(\sigma)\,) \end{array}$$

Commutation of the diagram is expressed by the conditions on the right: these provide a coinductive definition for $[\![\langle t, o \rangle]\!]_{N_A}$.

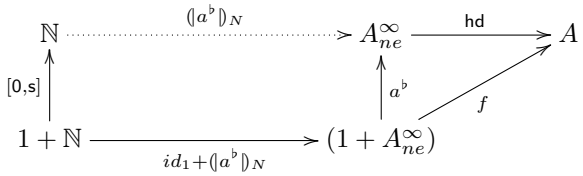
Since for all sets A there exists a final N_A-coalgebra, then there exists the cofree comonad $N^{\downarrow}$ and a distributive law $\zeta^{\downarrow}\colon NN^{\downarrow} \Rightarrow N^{\downarrow}N$ (more details in Appendix B). Most importantly, given a function $a\colon 1 + A^{\infty}_{ne} \to A$ (i.e, an $NN^{\downarrow}$-algebra), one can extend it to a function $a^{\flat}\colon 1 + A^{\infty}_{ne} \to A^{\infty}_{ne}$ (i.e, an N-algebra) coinductively defined for all $x \in 1 + A^{\infty}_{ne}$ as

$$\mathsf{hd}(\, a^{\flat}(x)\,) \stackrel{\text{def}}{=} a(x) \qquad \mathsf{tl}(\, a^{\flat}(x)\,) \stackrel{\text{def}}{=} \begin{cases} \epsilon & \text{if } x = \epsilon \\ a^{\flat}(\mathsf{tl}(x)) & \text{otherwise.} \end{cases} \tag{13}$$

By initiality of $(\mathbb{N}, [0, \mathsf{s}])$, one has a morphism $(\!|a^{\flat}|\!)_N$ inductively defined

$$\begin{array}{rcl} (\!|a^{\flat}|\!)_N(0) & = & a^{\flat}(\epsilon) \\ (\!|a^{\flat}|\!)_N(\mathsf{s}(n)) & = & a^{\flat}(\,(\!|a^{\flat}|\!)_N(n)\,) \end{array} \tag{14}$$

that can be composed with the counit hd to obtain the desired morphism $\mathbb{N} \to A$.



The inductive case in (14) can be conveniently rephrased (see Lemma 22 in Appendix B) as

$$(\!|a^{\flat}|\!)_N(\mathsf{s}(n)) = a(\,(\!|a^{\flat}|\!)_N(n)\,) \cdot (\!|a^{\flat}|\!)_N(n).$$

In summary, $(\!|a^{\flat}|\!)_N(0) = a(\epsilon) \cdot \epsilon$ and

$$(\!|a^{\flat}|\!)_N(\mathsf{s}(n)) = a((\!|a^{\flat}|\!)_N(n)) \cdot a((\!|a^{\flat}|\!)_N(n-1)) \cdot \ldots a((\!|a^{\flat}|\!)_N(0)) \cdot a(\epsilon) \cdot \epsilon.$$

Intuitively,$(\!|a^{\flat}|\!)_N(n+1)$ may depend on $(\!|a^{\flat}|\!)_N(n)$, $(\!|a^{\flat}|\!)_N(n-1)$ ...

For a concrete example take A to be $\mathbb{N}$ and a to be $fib\colon 1 + \mathbb{N}^{\infty}_{ne} \to \mathbb{N}$ defined as

$$fib(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } x = \epsilon \text{ or } x \text{ has length } 1 \\ x(0) + x(1) & \text{otherwise} \end{cases}$$

for all $x \in 1 + \mathbb{N}^{\infty}_{ne}$. The reader can easily check that $(\!|fib^{\flat}|\!)_N(0) = 1 \cdot \epsilon$, $(\!|fib^{\flat}|\!)_N(1) = 1 \cdot 1 \cdot \epsilon$, $(\!|fib^{\flat}|\!)_N(2) = 2 \cdot 1 \cdot 1 \cdot \epsilon$ and so on. By composing $(\!|fib^{\flat}|\!)_N$ with $\mathsf{hd}\colon A^{\infty}_{ne} \to A$ is clear that one obtains the Fibonacci sequence recalled in Section 2.

8 Conclusions and future work

We have introduced induction up-to by dualising the framework of coinduction up-to from [28]. More precisely, we defined the notion of cocompatible functions (Definition 5) and proved that such functions provide sound up-to techniques (Theorem 7) that can be conveniently composed in various ways (Proposition 8). In particular, for any monotone function f, its downclosure $f^{\downarrow}$ is always f-cocompatible (Corollary 9). We refer to induction up-to $f^{\downarrow}$ as strong induction. Our main insight is that the well-known principle of strong induction over the natural numbers is induction up-to $b^{\downarrow}$ (Section 6.1), where b is the map having $\mathbb{N}$ as its least fixed point.

We then demonstrated that, by applying comonadic recursion [39] to lattices, we obtain exactly our theory of induction up-to (Proposition 18), while strong induction corresponds to the lattice-theoretic version of course-of-value iteration [38, 7] (Proposition 20).

We believe that these results shed light on the relationship between the schemes used to define programs and the techniques employed to prove their properties. It is no coincidence that, in Section 2, we required strong induction to prove properties of the Fibonacci sequence, which is defined by course-of-value iteration.

By dualising the results in [5], which enhance the fibrational framework of Hermida and Jacobs [18], we can distill compatible inductive up-to techniques from each comonad D. Verifying all the details is a substantial task that we leave for future work.

References

1 Roberto M Amadio and Luca Cardelli. Subtyping recursive types. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 15(4):575–631, 1993. doi:10.1145/155183.155231.

2 Falk Bartels. Generalised coinduction. *Mathematical Structures in Computer Science*, 13(2):321–348, 2003. doi:10.1017/S0960129502003900.

3 Filippo Bonchi, Pierre Ganty, Roberto Giacobazzi, and Dusko Pavlovic. Sound up-to techniques and complete abstract domains. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 175–184, 2018. doi:10.1145/3209108.3209169.

4 Filippo Bonchi, Barbara König, and Daniela Petrisan. Up-to techniques for behavioural metrics via fibrations. *Math. Struct. Comput. Sci.*, 33(4-5):182–221, 2023. doi:10.1017/s0960129523000166.

5 Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. A general account of coinduction up-to. *Acta Informatica*, 54(2):127–190, 2017. doi:10.1007/S00236-016-0271-4.

6 Filippo Bonchi and Damien Pous. Checking NFA equivalence with bisimulations up to congruence. In Roberto Giacobazzi and Radhia Cousot, editors, *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013*, pages 457–468. ACM, 2013. doi:10.1145/2429069.2429124.

7 Daniela Cancila, Furio Honsell, and Marina Lenisa. Generalized coiteration schemata. *Electronic Notes in Theoretical Computer Science*, 82(1):76–93, 2003. doi:10.1016/S1571-0661(04)80633-9.

8 Venanzio Capretta, Tarmo Uustalu, and Varmo Vene. Recursive coalgebras from comonads. *Information and Computation*, 204(4):437–468, 2006. doi:10.1016/j.ic.2005.08.005.

9 Didier Caucal. Graphes canoniques de graphes algébriques. *RAIRO Theor. Informatics Appl.*, 24:339–352, 1990. doi:10.1051/ita/1990240403391.

10 Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Valeria Vignudelli. Up-to techniques for generalized bisimulation metrics. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, volume 59 of *LIPIcs*, pages 35:1–35:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.35.

11 Thierry Coquand. Infinite objects in type theory. In *International Workshop on Types for Proofs and Programs*, pages 62–78. Springer, 1993. doi:10.1007/3-540-58085-9_72.

12 Erik P. de Vink and Jan J.M.M. Rutten. Bisimulation for probabilistic transition systems: a coalgebraic approach. *Theoretical Computer Science*, 221(1):271–293, 1999. doi:10.1016/S0304-3975(99)00035-3.

13 Adrien Durier, Daniel Hirschkoff, and Davide Sangiorgi. Divergence and unique solution of equations. *Logical Methods in Computer Science*, 15, 2019. doi:10.23638/LMCS-15(3:12)2019.

14 Marco Forti and Furio Honsell. Set theory with free construction principles. *Annali della Scuola Normale Superiore di Pisa-Classe di Scienze*, 10(3):493–522, 1983.

15 Herman Geuvers and Bart Jacobs. Relating apartness and bisimulation. *Logical Methods in Computer Science*, 17, 2021. doi:10.46298/lmcs-17(3:15)2021.

16 Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3, 2007. doi:10.2168/LMCS-3(4:11)2007.

17 Ichiro Hasuo, Shunsuke Shimizu, and Corina Cîrstea. Lattice-theoretic progress measures and coalgebraic model checking. In *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 718–732, 2016. doi:10.1145/2837614.2837673.

18 Claudio Hermida and Bart Jacobs. Structural induction and coinduction in a fibrational setting. *Information and computation*, 145(2):107–152, 1998. doi:10.1006/inco.1998.2725.

19 Arend Heyting. Zur intuitionistischen axiomatik der projektiven geometrie. *Mathematische Annalen*, 98(1):491–538, 1928.

20 Bart Jacobs, Alexandra Silva, and Ana Sokolova. Trace semantics via determinization. In *International Workshop on Coalgebraic Methods in Computer Science*, pages 109–129. Springer, 2012. doi:10.1007/978-3-642-32784-1_7.

21 Bartek Klin. Bialgebras for structural operational semantics: An introduction. *Theor. Comput. Sci.*, 412(38):5043–5069, 2011. doi:10.1016/j.tcs.2011.03.023.

22 Dexter Kozen and Alexandra Silva. Practical coinduction. *Mathematical Structures in Computer Science*, 27(7):1132–1152, 2017. doi:10.1017/S0960129515000493.

23 Robin Milner. *Communication and concurrency*, volume 84 of *PHI Series in computer science*. Prentice Hall, 1989.

24 Robin Milner and Mads Tofte. Co-induction in relational semantics. *Theoretical computer science*, 87(1):209–220, 1991. doi:10.1016/0304-3975(91)90033-X.

25 Keiko Nakata and Tarmo Uustalu. Resumptions, weak bisimilarity and big-step semantics for while with interactive I/O: An exercise in mixed induction-coinduction. In *In Proceedings of SOS 2010*, pages 57–75, 2010. doi:10.4204/EPTCS.32.5.

26 David Park. Fixpoint induction and proofs of program properties. *Machine intelligence*, 5, 1969.

27 David Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science: 5th GI-Conference Karlsruhe, March 23–25, 1981*, pages 167–183. Springer, 2005.

28 Damien Pous. Complete lattices and up-to techniques. In Zhong Shao, editor, *Programming Languages and Systems, 5th Asian Symposium, APLAS 2007, Singapore, November 29-December 1, 2007, Proceedings*, volume 4807 of *Lecture Notes in Computer Science*, pages 351–366. Springer, 2007. doi:10.1007/978-3-540-76637-7_24.

29 Damien Pous. Coinduction all the way up. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 307–316, 2016. doi:10.1145/2933575.2934564.

30 Damien Pous and Davide Sangiorgi. Enhancements of the bisimulation proof method. In Davide Sangiorgi and Jan J. M. M. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge tracts in theoretical computer science*, pages 233–289. Cambridge University Press, UK, 2012.

31 Jan J.M.M. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata, and power series. *Theoretical Computer Science*, 308(1):1–53, 2003. doi:10.1016/S0304-3975(02)00895-2.

32 Davide Sangiorgi. On the bisimulation proof method. *Mathematical Structures in Computer Science , Volume 8 , Issue 5*, 1998. doi:10.1017/S0960129598002527.

33 Davide Sangiorgi. On the origins of bisimulation and coinduction. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 31(4):1–41, 2009. doi:10.1145/1516507.1516510.

34 Davide Sangiorgi. Equations, contractions, and unique solutions. *ACM Transactions on Computational Logic (TOCL)*, 18(1):1–30, 2017. doi:10.1145/2971339.

35 Davide Sangiorgi. From enhanced coinduction towards enhanced induction. *Proceedings of the ACM on Programming Languages*, 6(POPL):1–29, 2022. doi:10.1145/3498679.

36 Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing the powerset construction, coalgebraically. 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 272–283. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPIcs.FSTTCS.2010.272.

37 Daniele Turi and Gordon Plotkin. Towards a mathematical operational semantics. In *Proceedings of Twelfth Annual IEEE Symposium on Logic in Computer Science*, pages 280–291. IEEE, 1997. doi:10.1109/LICS.1997.614955.

38 Tarmo Uustalu and Varmo Vene. Primitive (co) recursion and course-of-value (co) iteration, categorically. *Informatica*, 10(1):5–26, 1999. doi:10.3233/INF-1999-10102.

39 Tarmo Uustalu, Varmo Vene, and Alberto Pardo. Recursion schemes from comonads. *Nordic Journal of Computing*, 8(3):366–390, 2001. URL: http://www.cs.helsinki.fi/njc/References/uustaluvp2001:366.html.

40 Johan Van Benthem. *Modal logic and classical logic*. 1983.

41 Rob J van Glabbeek. The linear time - branching time spectrum II: The semantics of sequential systems with silent moves extended abstract. In *International Conference on Concurrency Theory*, pages 66–81. Springer, 1993.

42 Rob J Van Glabbeek. The linear time - branching time spectrum I. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, pages 3–99. North-Holland / Elsevier, 2001. doi:10.1016/b978-044482830-9/50019-9.

A Appendix to Section 5

Proof of Proposition 8. We prove in sequence each point.

1. $f \circ id_X = f = id_X \circ f$;
2. $f \circ f = f \circ f$;
3. By the following derivation

$$\begin{aligned} f \circ d \circ e &\sqsubseteq d \circ f \circ e && (d \text{ is } f - \textit{compatible}) \\ &\sqsubseteq d \circ e \circ f && (e \text{ is } f - \textit{compatible} \text{ and } d \text{ is monotone}) \end{aligned}$$

4. By induction.
   Base case: $n = 0$. By (3) and point 1.
   Inductive case: $n = n + 1$. By hypothesis d is f-compatible; by induction hypothesis d^n is f-compatible; by point 3, $d \circ d^n$ is f-compatible. By definition, see (3), $d^{n+1} = d \circ d^n$. Thus d^{n+1} is f-compatible.
5. Proving $f(d \sqcap e) \sqsubseteq (d \sqcap e)f$ means proving that: for all $x \in X$,

   $$f(d \sqcap e)(x) \sqsubseteq (d \sqcap e)f(x).$$

   Since, for all $x \in X$,

   $$d \sqcap e(x) = d(x) \sqcap e(x),$$

   it holds that:

   $$\begin{aligned} f(d \sqcap e)(x) &= f(d(x) \sqcap e(x)) \\ (d \sqcap e)f(x) &= df(x) \sqcap ef(x) \end{aligned}$$

   In summary, to prove $f(d \sqcap e) \sqsubseteq (d \sqcap e)f$, we need to prove that, for all $x \in X$, $f(d(x) \sqcap e(x)) \sqsubseteq df(x) \sqcap ef(x)$. We can proceed separately:

- First, $f(d(x) \sqcap e(x)) \sqsubseteq df(x)$:

$$\begin{aligned} f(d(x) \sqcap e(x)) &\sqsubseteq fd(x) && (d(x) \sqcap e(x) \sqsubseteq d(x)) \\ &\sqsubseteq df(x) && (d \text{ is } f-\textit{compatible}) \end{aligned}$$

- Similarly for $f(d(x) \sqcap e(x)) \sqsubseteq df(x)$.

Since $f(d(x) \sqcap e(x))$ is below both $df(x)$ and $ef(x)$, we have that

$$f(d(x) \sqcap e(x)) \sqsubseteq df(x) \sqcap ef(x).$$

6. The proof is analogous to the one of Point 5. We need to prove that, for all $x \in X$,

$$f \bigsqcap_{i \in \mathbb{N}} d_i(x) \sqsubseteq \bigsqcap_{i \in \mathbb{N}} d_i f(x)$$

Thus, it is enough to prove that $\forall i \in \mathbb{N}$, $f \bigsqcap_j d_j(x) \sqsubseteq d_i f(x)$. We proceed as follows:

$$\begin{aligned} f \bigsqcap_{j \in \mathbb{N}} d_j(x) &\sqsubseteq fd_i(x) && (\textstyle\bigsqcap_{j \in \mathbb{N}} d_j \sqsubseteq d_i) \\ &\sqsubseteq d_i f(x) && (d_i \text{ is } f-\textit{compatible}) \end{aligned}$$

7. By (4) and Lemmas 4 and 6. ◀

Proof of 11. We prove the two points separately.

1. Assume that d is an up-closure; Thus $\neg x \sqsubseteq d(\neg x)$ and $d(d(\neg x)) \sqsubseteq d(\neg x)$ for all $x \in L$; By (9), $\neg\neg x \sqsupseteq \neg d(\neg x)$ and $\neg d(d(\neg x)) \sqsupseteq \neg d(\neg x)$, i.e., $x \sqsupseteq \overline{d}(x)$ and $\overline{d}\,\overline{d}(x) \sqsupseteq \overline{d}(x)$. The reverse implication has the same proof.

2. Observe that for all monotone maps i, j on L, it holds that $i \sqsubseteq j$ iff $i\neg \sqsubseteq j\neg$ iff $\neg j \sqsubseteq \neg i$. Thus

$$\begin{aligned} df \sqsubseteq fd &\Leftrightarrow df\neg \sqsubseteq fd\neg \\ &\Leftrightarrow \neg fd\neg \sqsubseteq \neg df\neg \\ &\Leftrightarrow \neg f\neg\neg d\neg \sqsubseteq \neg d\neg\neg f\neg \\ &\Leftrightarrow \overline{f}\,\overline{d} \sqsubseteq \overline{d}\,\overline{f} \end{aligned}$$

3. By the following derivation

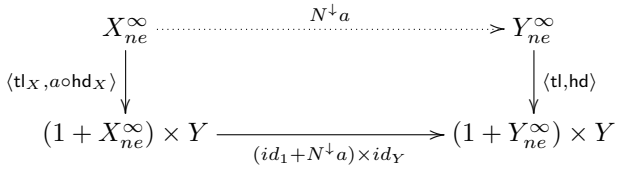
$$\begin{aligned} y \sqsubseteq fd(y) &\Leftrightarrow \neg fd(y) \sqsubseteq \neg y \\ &\Leftrightarrow \neg f\neg\, \neg d\neg(\neg y) \sqsubseteq \neg y \\ &\Leftrightarrow \overline{f}\,\overline{d}(\neg y) \sqsubseteq \neg y \end{aligned}$$

◀

B Details on Section 7.4

In Section 7.3, we give the recipe to compute the cofree comonad for an arbitrary functor F and in Section 7.4 we used the cofree comonad for the functor $N\colon \mathbf{Set} \to \mathbf{Set}$ in order to illustrate course-of-value Iteration for the natural numbers. For reader convenience, in this appendix we illustrate in details the cofree comonad $(N^{\downarrow}, \epsilon^{\downarrow}, \delta^{\downarrow})$ and the distributive law $\zeta\colon NN^{\downarrow} \Rightarrow N^{\downarrow}N$ by simply unfolding the definitions of Section 7.3.

First we illustrate the endofunctor $N^{\downarrow}\colon \mathbf{Set} \to \mathbf{Set}$. It maps any set X into the set X^{∞}_{ne} since, as discussed in the main text, $(X^{\infty}_{ne}, \langle \mathsf{tl}, \mathsf{hd} \rangle)$ is a final N_X coalgebra. For a function $a\colon X \to Y$, $N^{\downarrow}a\colon X^{\infty}_{ne} \to Y^{\infty}_{ne}$ is the unique map from the N_Y-coalgebra $(X^{\infty}_{ne}, \langle \mathsf{tl}_X, a \circ \mathsf{hd}_X \rangle)$ to the final N_Y-coalgebra $(Y^{\infty}_{ne}, \langle \mathsf{tl}, \mathsf{hd} \rangle)$ as illustrated below.



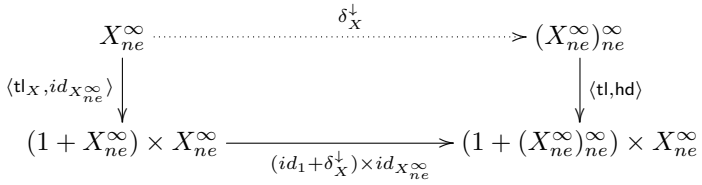
Thus, the function $N^\downarrow a \colon X^\infty_{ne} \to Y^\infty_{ne}$ can be defined conductively as follows.

$$\begin{aligned} \mathsf{hd}(\, N^\downarrow a(\sigma)\,) &= a(\mathsf{hd}_X(\sigma)) \\ \mathsf{tl}(\, N^\downarrow a(\sigma)\,) &= \begin{cases} \epsilon & \text{if } \mathsf{tl}_X(\sigma) = \epsilon; \\ N^\downarrow a(\mathsf{tl}_X(\sigma)) & \text{otherwise.} \end{cases} \end{aligned}$$

More explicitly, $N^\downarrow a$ maps $\sigma \in X^\infty_{ne}$ into

$$a(\sigma(0)) \cdot a(\sigma(1)) \cdot \ldots$$

We can now illustrate the natural transformations. The counit $\epsilon \colon N^\downarrow \Rightarrow Id$ is given, for all sets X, by $\mathsf{hd}_X \colon X^\infty_{ne} \to X$. The comultiplication $\delta^\downarrow \colon N^\downarrow \Rightarrow N^\downarrow N^\downarrow$ is given by the unique morphism from the $N_{X^\infty_{ne}}$-coalgebra $(X^\infty_{ne}, \langle \mathsf{tl}_X, id_{X^\infty_{ne}} \rangle)$ to the final $N_{X^\infty_{ne}}$-coalgebra $((X^\infty_{ne})^\infty_{ne}, \langle \mathsf{tl}, \mathsf{hd} \rangle)$, as illustrated below.



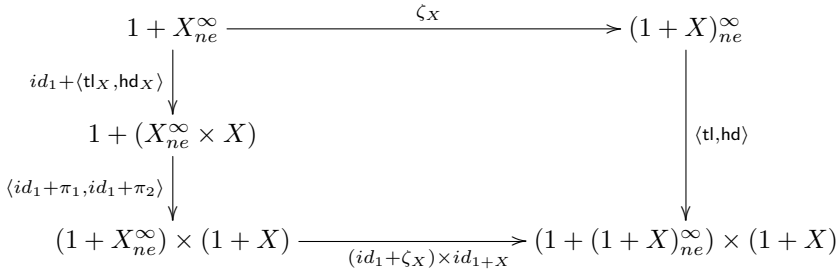
Thus, the function $\delta^\downarrow_X \colon X^\infty_{ne} \to (X^\infty_{ne})^\infty_{ne}$ can be coinductively defined as follows.

$$\begin{aligned} \mathsf{hd}(\, \delta^\downarrow_X(\sigma)\,) &= \sigma \\ \mathsf{tl}(\, \delta^\downarrow_X(\sigma)\,) &= \begin{cases} \epsilon & \text{if } \mathsf{tl}_X(\sigma) = \epsilon; \\ \delta^\downarrow_X(\mathsf{tl}_X(\sigma)) & \text{otherwise.} \end{cases} \end{aligned}$$

Intuitively, $\sigma \in X^\infty_{ne}$ is mapped into

$$\begin{array}{llll} \sigma(0) & \sigma(1) & \sigma(2) & \ldots \\ \sigma(1) & \sigma(2) & \ldots & \\ \sigma(2) & \ldots & & \\ \vdots & & & \end{array}$$

So far, we have described the comonad $(N^\downarrow, \epsilon^\downarrow, \delta^\downarrow)$. We can now move to the distributive law. For all sets X, the distributive law $\zeta \colon NN^\downarrow \Rightarrow N^\downarrow N$ is given by the unique morphism from the N_{1+X}-coalgebra illustrated below on the left to the final N_{1+X}-coalgebra $((1 + X)^\infty_{ne}, \langle \mathsf{tl}, \mathsf{hd} \rangle)$.



Thus, the function $\zeta_X \colon 1 + X^\infty_{ne} \to (1+X)^\infty_{ne}$ can be coinductively defined as follows.

$$\begin{aligned}
\mathsf{hd}(\,\zeta_X(\epsilon)\,) &= \epsilon \\
\mathsf{tl}(\,\zeta_X(\epsilon)\,) &= \epsilon \\
\mathsf{hd}(\,\zeta_X(\sigma)\,) &= \mathsf{hd}_X(\sigma) \\
\mathsf{tl}(\,\zeta_X(\sigma)\,) &= \begin{cases} \epsilon & \text{if } \mathsf{tl}_X(\sigma) = \epsilon; \\ \zeta_X(\mathsf{tl}_X(\sigma)) & \text{otherwise.} \end{cases}
\end{aligned}$$

Intuitively $[\![h]\!]_{1+X}$ maps ϵ into the sequence $\epsilon \cdot \epsilon$ and any $\sigma \in X^\infty_{ne}$ into the same sequence $\sigma \in (1+X)^\infty_{ne}$.

We conclude this appendix with some computation that allows for the handier characterisation of $(\!|a^\flat|\!)_N$ illustrated in the main text.

▶ **Lemma 21.** *For all $n \in \mathbb{N}$, $a^\flat(\mathsf{tl}(\,(\!|a^\flat|\!)_N(n)\,)) = (\!|a^\flat|\!)_N(n)$.*

Proof. By induction on $\mathbb{N}$.

For 0,

$$\begin{aligned}
a^\flat(\mathsf{tl}(\,(\!|a^\flat|\!)_N(0)\,)) &= a^\flat(\mathsf{tl}(\,a^\flat(\epsilon)\,)) && (14)\\
&= a^\flat(\epsilon) && (13)\\
&= (\!|a^\flat|\!)_N(0) && (14)
\end{aligned}$$

For $n+1$,

$$\begin{aligned}
a^\flat(\mathsf{tl}(\,(\!|a^\flat|\!)_N(n+1)\,)) &= a^\flat(\mathsf{tl}(\,a^\flat(\!|a^\flat|\!)_N(n)\,)) && (14)\\
&= a^\flat(a^\flat(\,\mathsf{tl}((\!|a^\flat|\!)_N(n)\,))) && (13)\\
&= a^\flat((\!|a^\flat|\!)_N(n)) && \text{(Induction Hypothesis)}\\
&= (\!|a^\flat|\!)_N(n+1) && (14)
\end{aligned}$$

◀

▶ **Lemma 22.** *For all $n \in \mathbb{N}$, $(\!|a^\flat|\!)_N(\mathsf{s}(n)) = a(\,(\!|a^\flat|\!)_N(n)\,) \cdot (\!|a^\flat|\!)_N(n)$.*

Proof.

$$\begin{aligned}
(\!|a^\flat|\!)_N(\mathsf{s}(n)) &= a^\flat(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,) && (14)\\
&= \mathsf{hd}(a^\flat(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,)) \cdot \mathsf{tl}(a^\flat(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,)) \\
&= a(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,) \cdot a^\flat(\mathsf{tl}(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,)) && (13)\\
&= a(\,(\!|a^\flat|\!)_N(\mathsf{s}(n))\,) \cdot (\!|a^\flat|\!)_N(n) && \text{(Lemma 21)}
\end{aligned}$$

◀

Correspondences Between Codensity and Coupling-Based Liftings, a Practical Approach

Samuel Humeau
ENS de Lyon, CNRS, LIP, UMR 5668, 69342, Lyon cedex 07, France

Daniela Petrisan
CNRS, IRIF, Université Paris Diderot, Paris, France

Jurriaan Rot
Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands

Abstract

The Kantorovich distance is a widely used metric between probability distributions. The Kantorovich-Rubinstein duality states that it can be defined in two equivalent ways: as a supremum, based on non-expansive functions into $[0, 1]$, and as an infimum, based on probabilistic couplings.

Orthogonally, there are categorical generalisations of both presentations proposed in the literature, in the form of *codensity liftings* and what we refer to as *coupling-based liftings*. Both lift endofunctors on the category **Set** of sets and functions to that of pseudometric spaces, and both are parameterised by modalities from coalgebraic modal logic.

A generalisation of the Kantorovich-Rubinstein duality has been more nebulous – it is known not to work in some cases. In this paper we propose a compositional approach for obtaining such generalised dualities for a class of functors, which is closed under coproducts and products. Our approach is based on an explicit construction of modalities and also applies to and extends known cases such as that of the powerset functor.

2012 ACM Subject Classification Theory of computation → Categorical semantics

Keywords and phrases Kantorovich distance, behavioural metrics, Kantorovich-Rubinstein duality, functor liftings

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.29

Related Version *Full Version*: `https://hal.science/hal-04789352` [13]

Funding This work was supported by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX- 0007) operated by the French National Research Agency (ANR), and supported by the NWO grant OCENW.M20.053.

Acknowledgements The authors would like to thank Pedro Nora for suggestions and discussions.

1 Introduction

The *Kantorovich* (or *Wasserstein*, or *Monge-Kantorovich*) distance [17] is a standard and widely used metric between probability distributions, studied amongst others in transportation theory [27]. In concurrency theory, the Kantorovich distance forms the basis of so-called *behavioural metrics*, which are quantitative generalisations of bisimilarity. They allow a more fine-grained and robust comparison of system behaviours than classical Boolean-valued behavioural equivalences [8, 9, 26].

In its discrete version the Kantorovich distance takes as argument a (pseudo-)metric on a set X, and lifts it to a (pseudo-)metric on the set of (finitely supported) distributions $\mathcal{D}(X)$. The celebrated *Kantorovich-Rubinstein duality* [18] states that this distance can be computed in two ways, yielding the same result: as an infimum indexed by probabilistic couplings, and as a supremum indexed by non-expansive functions into the $[0, 1]$ interval with the Euclidean

distance. This fundamental result is useful for analysis and computation of these distances (e.g., [1, 7, 15, 25, 27]). A detailed proof in a broader context than just finitely supported distributions can be found in [27].

Orthogonally to this duality, in the last years there have been several proposals to generalise the Kantorovich distance from distributions to general endofunctors on the category **Set** of sets and functions. The problem then becomes to lift such functors to the category of (pseudo-)metric spaces. This is particularly useful in the context of a coalgebraic presentation of systems, where the type of the system at hand is parametric in the given **Set** endofunctor. In particular, this allows a uniform presentation of various types of probabilistic systems, but also, for instance, metrics on deterministic automata [6].

There are categorical generalisations of both presentations of the Kantorovich distance: the *coupling-based* approach and the one based on non-expansive maps [2, 5, 6, 11]. The latter has recently been established as an instance of the so-called *codensity liftings*[1] [24]. Both approaches are parametric in (sets of) *modalities* or *evaluation maps*, that allow a degree of freedom in the choice of liftings.

We aim to relate the two approaches, by studying generalisations of the Kantorovich-Rubinstein duality to a wide class of functors beyond $\mathcal{D}$. This problem was first proposed and studied in [2], where it is shown to hold in some concrete cases but also to fail in other basic instances, even for very elementary functors such as the diagonal functor Δ mapping a set X to $X \times X$. In the latter article the authors restrict the study to liftings parametric in exactly one modality they call an *evaluation map*, which is assumed to be the same for both codensity and coupling-based liftings. We depart from this by allowing modalities to differ on both sides. This approach can already be found in [10]. There, it is shown that every coupling-based lifting can be presented as a codensity lifting; but the proof is non-constructive and yields a large collection of modalities.

In the current paper, we approach the problem of generalised Kantorovich-Rubinstein dualities from a concrete perspective, with an emphasis on compositionality aspects. Given a modality as parameter for a coupling-based lifting, we aim to explicitly translate it to modalities for a codensity lifting, in such a way that the two correspond. More explicitly, we show that the class of such correspondences between coupling-based and codensity liftings is closed under coproducts and products (and conversely, that every correspondence for a coproduct of functors can be recovered from correspondences on its constituents). We also investigate correspondences for the identity functor, where there is flexibility in the choice of modalities; and for the powerset functor, extending earlier correspondence results [2, 12].

These correspondence results then allow us to define a concrete *grammar* of functors for which we obtain a correspondence between coupling-based and codensity liftings. In fact, we obtain several grammars based on different assumptions on the underlying poset of truth values (assumed to be a quantale). Base cases include the constant functors, distribution functor, powerset functor and the identity functor; recursive constructions the product and coproduct. These results allow us to obtain or recover induced codensity and coupling-based presentations for a range of examples of behavioural metrics, including metrics on streams, labelled Markov chains, deterministic automata, and non-deterministic automata.

[1] *Kantorovich metric* is used interchangeably in the literature to refer to both presentations. We therefore avoid this terminology by consistently referring to the two presentations as *coupling-based* and *codensity* liftings respectively.

In the last part of the paper, we investigate the limitations of our approach through the concrete example of *conditional transition systems* [4, 3]. In contrast to the earlier examples, here, our grammar does give us a metric (and a correspondence result), but it is not the one considered earlier in the literature. We prove that, in fact, the metric from the literature can be expressed with a codensity lifting but not with a coupling-based lifting.

2 Codensity and coupling-based liftings: correspondences by example

In this section we motivate and describe the problem of correspondences between codensity and coupling-based liftings at the general level of functors, and our approach in this paper, by means of two examples: the Kantorovich-Rubinstein duality for distributions, and a similar correspondence for the shortest-distinguishing-word-distance on deterministic automata [6].

Distributions. We start by recalling the classical Kantorovich distance, in the discrete case. In this paper we focus on pseudometrics (defined like metrics except that different elements can have distance 0) as is common in the use of these types of distances in concurrency theory. For a pseudometric $d\colon X \times X \to [0,1]$, the Kantorovich distance is a pseudometric on the set $\mathcal{D}(X)$ of distributions on X, defined, for $\mu, \nu \in \mathcal{D}(X)$ by:

$$\mathcal{D}^{\downarrow}(d)(\mu,\nu) = \inf_{\sigma \in \Omega(\mu,\nu)} \sum_{x,y \in X} d(x,y) \cdot \sigma(x,y) \tag{1}$$

where $\Omega(\mu,\nu)$ is the set of couplings between μ and ν, i.e., probability distributions on $X \times X$ whose marginals are μ and ν respectively. It can equivalently be computed as:

$$\mathcal{D}^{\uparrow}(d)(\mu,\nu) = \sup_{f\colon X \to [0,1] \text{ n.e.}} \left| \sum_{x \in X} f(x) \cdot \mu(x) - \sum_{x \in X} f(x) \cdot \nu(x) \right| \tag{2}$$

where the supremum ranges over *non-expansive functions* f into $[0,1]$ equipped with the Euclidean distance, i.e., such that $|f(x) - f(y)| \leq d(x,y)$ for all $x, y \in X$. The equality $\mathcal{D}^{\uparrow}(d) = \mathcal{D}^{\downarrow}(d)$ is an instance (see [27, Particular case 5.16]) of a general result known as the Kantorovich-Rubinstein duality (see [27, Theorem 5.10]).

A different example. The Kantorovich distance can be seen as a *lifting* of the distribution functor on the category **Set** of sets and functions to the category **PMet** of pseudometric spaces and non-expansive functions between them. We proceed with a quite different example of a similar phenomenon: a lifting of a functor from **Set** to $\mathbf{PMet}_{\leq 1}$ the category of pseudometric spaces bounded by 1 presented in two ways, as an infimum over a variant of couplings and a supremum over non-expansive functions. This example describes a distance between *deterministic finite automata (DFA)*. Empty infima and suprema are defined w.r.t. the interval $[0,1]$ where the pseudometrics take their values: $\sup \emptyset = 0$ and $\inf \emptyset = 1$.[2]

We view DFA over an alphabet A as coalgebras for the functor $F\colon \mathbf{Set} \to \mathbf{Set}$, $F(X) = 2 \times X^A$ where $2 = \{0,1\}$. Coalgebras for this functor are of the form $\langle l, \delta \rangle\colon X \to 2 \times X^A$, where X is the set of states, the output function $l\colon X \to 2$ describes which states are accepting ($l(x) = 1$), and $\delta\colon X \to X^A$ is the transition function. As usual, a word $\omega \in A^*$ is *accepted* in a state x when, after reading ω starting on x we end up with an accepting state.

[2] The *knowledge* package is used throughout the paper. Most of the introduced vocabulary and notations are clickable and the associated links brings the reader to their definitions.

Given $c \in [0,1)$, the *shortest-distinguishing-word-distance* $d_{\mathrm{sdw}}(x,y)$ between states x, y is 0 if they recognise the same language and $c^{|\omega|}$ for ω a shortest word that belongs exactly to one of the two languages recognised by x and y. As shown in [6], this distance can be computed recursively as a fixpoint of the map Φ on pseudometrics $X \times X \to [0,1]$ given by

$$\Phi(d)(x,y) = \begin{cases} 1 \text{ if } l(x) \neq l(y) \\ c \cdot \max_{a \in A}\{d(\delta(x)(a), \delta(y)(a))\} \end{cases}$$

The shortest-distinguishing-word-distance can be obtained via a *lifting* $\overline{F}\colon \mathbf{PMet} \to \mathbf{PMet}$ of the **Set** functor F, mapping a pseudometric space (X,d) to a pseudometric space $(FX, \overline{F}d)$, that we will recall below. The operator Φ above factors through the pseudometric $\overline{F}d$. Explicitly, it decomposes as $\Phi = \overline{F}d \circ \langle l, \delta \rangle$.

In fact, just as in the Kantorovich-Rubinstein duality, the lifting $\overline{F}$ can be obtained in two different ways. Given a pseudometric $d\colon X \times X \to [0,1]$ and $(l_1, \delta_1), (l_2, \delta_2) \in 2 \times X^A$, it arises as a coupling-based lifting $\overline{F}^{\downarrow}d$ by:

$$\overline{F}^{\downarrow}(d)((l_1,\delta_1),(l_2,\delta_2)) = \inf_{(l,\delta) \in F(X \times X),\ F\pi_i(l,\delta) = (l_i, \delta_i)} \left(\sup_{a \in A} c \cdot d(\delta(a)) \right) \tag{3}$$

Here elements of $2 \times (X \times X)^A$ are viewed as a variant of couplings.

Secondly we obtain $\overline{F}$ as a so-called codensity lifting by:

$$\overline{F}^{\uparrow}(d)((l_1,\delta_1),(l_2,\delta_2)) = \sup_{f\colon X \to [0,1] \text{ n.e.}} \{|c \cdot f(\delta_1(a)) - c \cdot f(\delta_2(a))| \mid a \in A\} \cup \{|l_1 - l_2|\} \tag{4}$$

where, as above, the supremum again ranges over non-expansive functions f. Just as for the Kantorovich distance, we again obtain an equality of functors on **PMet**: $\overline{F}^{\uparrow} = \overline{F}^{\downarrow}$.

Modalities as parameters. The abstract coupling-based and codensity liftings take as input a single *modality* (coupling-based), or a *set of modalities* (codensity), known from the semantics of coalgebraic modal logic. For the Kantorovich lifting, this modality is the expected value function $\mathbb{E}\colon \mathcal{D}([0,1]) \to [0,1]$, which appears implicitly in both presentations.

In the case of deterministic automata, while a single modality suffices for the coupling-based lifting, for the codensity lifting we actually use a *set* of modalities (one for each letter, plus a modality for acceptance of states). This observation is important for the investigation in this paper. Instead of aiming for a one-to-one matching of modalities, which we refer to as a *duality*, we allow a coupling-based lifting specified by a single modality to be matched by a codensity lifting specified by a set of modalities; we refer to this as a *correspondence*. This allows us to cover examples such as DFA and way more, and circumvent the problem already observed in [2]: even for the product functor with the modality $\max\colon [0,1] \times [0,1] \to [0,1]$ there is no duality. However, there is a correspondence, that is, multiple modalities are needed for the codensity lifting to match the coupling-based one.

The idea of allowing multiple modalities for codensity liftings is not new and can be found already at the origins of codensity liftings [19, 24, 21]. Here we show how it can be used to generalise Kantorovich-Rubinstein dualities in a very concrete manner; the aim is to define classes of functors and modalities for which we can explicitly describe correspondences between associated coupling-based and codensity liftings.

Outline. We work at the general level of pseudometrics valued in a quantale; we recall the definitions in Section 3. We then recall the abstract notions of coupling-based and codensity liftings of functors along modalities, and formulate the problem of correspondences (Section 4).

In Section 5 we prove our main results on correspondences: we show that the class of functors for which we have correspondences is closed under products and coproducts, and includes constant and identity functors, yielding a family of correspondences for simple polynomial functors including DFA. In Section 6 we revisit the duality results for the finite powerset and finite probability distribution functors. As a consequence we are able to study in Section 7 a class of functors for which we have correspondences, generated by a grammar. The case of conditional transition systems, for which we prove that there are no correspondences possible, is treated in Section 8.

3 Preliminaries: quantales and pseudometrics

We use quantales to model a general notion of truth object, including both Booleans and real number intervals. In this section we recall the necessary preliminaries on quantales, and the associated general notion of pseudometrics valued in a quantale; instances include the standard notion of pseudometrics on real numbers as well as equivalence relations.

▶ **Definition 1.** *A* quantale $\mathcal{V}$ *is a complete lattice with an associative "and" operation* $\otimes\colon \mathcal{V}\times\mathcal{V}\to\mathcal{V}$ *which is distributive over arbitrary joins. We only consider quantales that are* commutative*, i.e., the operation* $\otimes$ *is commutative,* unital*, meaning* $\otimes$ *admits a unit element, and* affine*, which means that the top element* $\top$ *is the unit of* $\otimes$*:* $x\otimes\top = x = \top\otimes x$ *for all* $x\in\mathcal{V}$*.*

Given a quantale $\mathcal{V}$, there is an operation $[-,-]\colon \mathcal{V}\times\mathcal{V}\to\mathcal{V}$ that is characterised by $x\otimes y\leq z \Leftrightarrow x\leq [y,z]$. It is simply defined by $[y,z] = \bigvee\{x\in\mathcal{V} \mid x\otimes y\leq z\}$.

▶ **Example 2.**

- Any complete Boolean algebra is a quantale with $\otimes = \wedge$; in particular, the usual Boolean algebra $2 = \{\bot,\top\}$ with $\bot\leq\top$. In this case, $[x,y] = \top$ iff $x\leq y$.
- Any interval $[0,M]$ with $M\in(0,\infty]$, given with *reversed order* and $x\otimes y = \min(x+y,M)$ the *truncated sum* is a quantale. Here the top element is given by $\top = 0$, and the bottom element by $\bot = M$. For this quantale, we have $[x,y] = \max\{y-x,0\}$. We write $\overline{\mathbb{R}}$ for the case that $M=\infty$, i.e., the non-negative real numbers extended with a top element.

We use quantales as an abstract notion of truth object; accordingly, we define a $\mathcal{V}$*-predicate* on a set X to be a map $p\colon X\to\mathcal{V}$. Of particular interest are $\mathcal{V}$-pseudometrics. These are predicates on $X\times X$ that are reflexive, symmetric and transitive in a suitable sense that can be expressed at the general level of quantales.

▶ **Definition 3.** *A* $\mathcal{V}$-pseudometric *on a set* X *is a map* $d\colon X\times X\to\mathcal{V}$ *which is:*

- reflexive*:* $d(x,x) = \top$ *for all* $x\in X$*,*
- symmetric*:* $d(x,y) = d(y,x)$ *for all* $x,y\in X$*,*
- transitive*:* $\bigvee_{z\in X} d(x,z)\otimes d(z,y)\leq d(x,y)$ *for all* $x,y\in X$*.*

▶ **Example 4.**

- 2-pseudometrics are equivalence relations.
- $\overline{\mathbb{R}}$-pseudometrics are the "usual" pseudometrics, that is, maps $d\colon X\times X\to\overline{\mathbb{R}}$ such that $d(x,x) = 0$, $d(x,y) = d(y,x)$ and $d(x,y)\leq d(x,z)+d(z,y)$ for all $x,y,z\in\overline{\mathbb{R}}$. To see why we obtain the triangle inequality from transitivity, recall that the order on the quantale is reversed w.r.t. the usual order on real numbers, and that $\otimes = +$.

The order on a quantale $\mathcal{V}$ is extended pointwise to functions:

$$\forall f, g \colon X \to \mathcal{V},\ f \leq g \Leftrightarrow (\forall x \in X,\ f(x) \leq g(x))$$

Given two $\mathcal{V}$-pseudometrics d_X and d_Y on sets X and Y respectively, a map $f \colon X \to Y$ is a *($\mathcal{V}$-pseudometric) morphism* from d_X to d_Y if $d_X \leq d_Y \circ (f \times f)$. We write **$\mathcal{V}$-PMet** for the category whose objects are $\mathcal{V}$-pseudometrics and arrows morphisms between them.

▶ Remark 5. Because of the reversal of the order on $\overline{\mathbb{R}}$, morphisms of $\overline{\mathbb{R}}$-pseudometrics are non-expansive maps. To avoid confusion with the quantale definition, we refrain from using the word *non-expansive* altogether, and replace it by *($\mathcal{V}$-pseudometric) morphism* instead.

The following canonical $\mathcal{V}$-pseudometric is essential for the notion of codensity lifting.

▶ **Definition 6.** *The* Euclidean pseudometric $\mathrm{d}_e \colon \mathcal{V} \times \mathcal{V} \to \mathcal{V}$ *is defined as follows:*

$$\mathrm{d}_e(x, y) = [x, y] \wedge [y, x]$$

▶ **Example 7.**

- For the quantale $\mathsf{2}$, the Euclidean pseudometric $\mathrm{d}_e \colon \mathsf{2} \times \mathsf{2} \to \mathsf{2}$ sends equal elements to $\top$ and different ones to $\bot$.
- For the quantale $\overline{\mathbb{R}}$, the Euclidean pseudometric $\mathrm{d}_e \colon \overline{\mathbb{R}} \times \overline{\mathbb{R}} \to \overline{\mathbb{R}}$ instantiates to the usual Euclidean distance, i.e., $\mathrm{d}_e(x, y) = |y - x|$.

4 Liftings, dualities, and correspondences

In this section we recall the definitions of coupling-based and codensity liftings from the literature (Section 4.2) and define the problem of their correspondence. This is followed by a few technical tools that we use in the proofs of correspondence (Section 4.3). We start with the notion of (well-behaved) modality (Section 4.1), used in these liftings.

4.1 Well-behaved modalities

Given a **Set** endofunctor F, a *modality* for F is a function $\tau \colon F\mathcal{V} \to \mathcal{V}$. Modalities are standard in the semantics of coalgebraic modal logic [23]. Note that here we assume $\mathcal{V}$ to be a quantale, to have a suitable notion of pseudometrics.

In order for coupling-based liftings to be well-defined some particular conditions on the functor and the associated modalities are needed. The underlying functor F is assumed to preserve weak pullbacks. In this context, we say τ is *well-behaved* [2] when:

- it is *monotone*, meaning that for all $\mathcal{V}$-predicates $p \leq q$ we have $\tau \circ F(p) \leq \tau \circ F(q)$,
- for all $\mathcal{V}$-predicates p and q, $\tau \circ F(p \otimes q) \geq (\tau \circ Fp) \otimes (\tau \circ Fq)$,
- with $i \colon \{\top\} \hookrightarrow \mathcal{V}$ the inclusion map, $Fi(F\{\top\}) = \tau^{-1}(\top)$.

▶ **Example 8.**

- On the identity functor, with the quantale $\overline{\mathbb{R}}$, a modality is just a map $\tau \colon \mathcal{V} \to \mathcal{V}$. It is well-behaved if and only if it is monotone, subadditive (i.e., $\tau(r + s) \leq \tau(r) + \tau(s)$), and satisfies $\tau(0) = 0$.
- For the *finite powerset functor* $\mathcal{P}$ mapping a set X to the set of its finite subsets $\mathcal{P}(X)$, with the same quantale $\overline{\mathbb{R}}$, the maximum function is well-behaved.
- For the *finite distribution functor* $\mathcal{D}$ mapping a set X to the set of its finitely supported probability distributions $\mathcal{D}(X)$, with the quantale $[0, 1]$, the map $\mathbb{E} \colon \mathcal{D}([0, 1]) \to [0, 1]$ giving the expected value of a probability distribution is well-behaved.

In the rest of this paper we consider constant, identity, finite powerset, finite probability distribution functors, and combine them using products and coproducts. All these functors preserve weak pullbacks (see [14, Propositions 4.2.6 and 4.2.10] for example), ensuring we can consider well-behaved modalities for them.

Well-behaved modalities can be constructed from existing ones. The case of composition is a generalisation to quantales of a particular case of [2, Theorem 7.2].

▶ **Proposition 9.** *Given three arbitrary well-behaved modalities $\tau, \tau' \colon F\mathcal{V} \to \mathcal{V}$ and $\tau_{\mathrm{Id}} \colon \mathcal{V} \to \mathcal{V}$, the following modalities are again well-behaved: $\tau_{\mathrm{Id}} \circ \tau$, $\tau \otimes \tau'$ and $\tau \wedge \tau'$.*

4.2 Liftings and correspondences

Given a functor $P \colon \mathbf{C} \to \mathbf{D}$, a *lifting* of a $\mathbf{D}$-functor $F \colon \mathbf{D} \to \mathbf{D}$ from $\mathbf{D}$ to $\mathbf{C}$ is a functor $\overline{F} \colon \mathbf{C} \to \mathbf{C}$ such that $P \circ \overline{F} = F \circ P$. Here we only consider the case when $\mathbf{D} = \mathbf{Set}$ and $\mathbf{C} = \mathcal{V}\text{-}\mathbf{PMet}$, with P the forgetful functor sending a $\mathcal{V}$-pseudometric of type $X \times X \to \mathcal{V}$ to its underlying set X and acting as the identity on arrows. We define coupling-based liftings, codensity liftings, and the associated notion of correspondence that we study here.

Given $t_1, t_2 \in FX$, a *coupling* of t_1 and t_2 is an element $t \in F(X \times X)$ such that $F\pi_i(t) = t_i$. The set of couplings of t_1 and t_2 is denoted by $\Omega(t_1, t_2)$. In particular, if $F = \mathcal{D}$ is the distribution functor on $\mathbf{Set}$, these are precisely probabilistic couplings: joint distributions on $X \times X$ whose marginals coincide with given distributions t_1, t_2. The following lifting arises from [11]. See also [6]. It is not referred to in the literature as "coupling-based"; we use this terminology to distinguish it from the codensity lifting.

▶ **Definition 10.** *Let $F \colon \mathbf{Set} \to \mathbf{Set}$ be a functor which preserves weak pullbacks, and let τ be a well-behaved modality for F. The* coupling-based lifting *of F to $\mathcal{V}\text{-}\mathbf{PMet}$ is given by, with $d \in \mathcal{V}\text{-}\mathbf{PMet}$ and $t_1, t_2 \in FX$:*

$$F^{\downarrow}_{\tau}(d)(t_1, t_2) = \bigvee_{t \in \Omega(t_1, t_2)} \tau \circ Fd(t)$$

▶ **Example 11.** In Section 2, we studied a coupling-based lifting for DFA, which yields the shortest-distinguishing-word-distance. To see this as an instance of Definition 10, the quantale is $\mathcal{V} = [0, 1]$ as introduced in Section 3, the functor F is the one mapping a set X to $2 \times X^A$. The well-behaved modality takes as argument an element $(l, \delta) \in 2 \times \mathcal{V}^A$ and returns $\bigwedge_{a \in A} c \cdot \delta(a)$ for some $c \in [0, 1)$. The resulting lifting is precisely the one given in Equation (3).

The *codensity lifting* is defined in a very general setting for monads in [19]. Here, we use an instance, which appears in [24, 21] and, for a single modality, in [2].

▶ **Definition 12.** *Let $F \colon \mathbf{Set} \to \mathbf{Set}$, and let Γ be a family of modalities for F. The* codensity lifting *of F along Γ to $\mathcal{V}\text{-}\mathbf{PMet}$ is defined by, with $d \in \mathcal{V}\text{-}\mathbf{PMet}$ and $t_1, t_2 \in FX$:*

$$F^{\uparrow}_{\Gamma}(d)(t_1, t_2) = \bigwedge_{\tau \in \Gamma,\ f \colon d \to \mathrm{d}_e} \mathrm{d}_e(\tau \circ Ff(t_1), \tau \circ Ff(t_2))$$

Note that the maps $f \colon d \to \mathrm{d}_e$ in the definition of the codensity lifting are $\mathcal{V}$-pseudometric morphisms, so that, for instance, in $\overline{\mathbb{R}}$, they correspond to non-expansive maps (cf. Remark 5).

▶ **Example 13.** In the example of Section 2 the codensity lifting has one modality mapping $(l, \delta) \in 2 \times \mathcal{V}^A$ to l, as well as one modality τ_a for each letter $a \in A$ given by $\tau_a(l, \delta) = c \cdot \delta(a)$. The resulting codensity lifting coincides with the lifting given in Equation (4).

Note that the coupling-based liftings are only allowed to have one modality, whereas the codensity ones may have multiple modalities. Informally, one of the reasons coupling-based liftings do not require multiple modalities can be seen by looking again at the functor for DFA. Recall that it maps a set X to $2 \times X^A$ for $2 = \{0, 1\}$ and A an alphabet. Considering couplings forces comparisons using d to be done separately on each letter. Codensity liftings instead use one modality per letter to ensure a similar condition. On a higher level of abstraction, the multiple modalities of codensity liftings relate to the expressivity of some modal logics (see, e.g., [20]). On the other hand, the unique modality for coupling-based liftings has often been called an evaluation function (see [2]). Having only one way of evaluating something seems natural in a given evaluation context.

In the rest of this paper we study possible equalities between the coupling-based and codensity liftings. In general, we allow a well-behaved modality for the coupling-based liftings to be matched by a set of (different) modalities for the codensity lifting. In case the modalities are the same, we call this Kantorovich-Rubinstein duality.

▶ **Definition 14.** *Let $F\colon \mathbf{Set} \to \mathbf{Set}$ be a weak-pullback preserving functor, $\tau\colon F\mathcal{V} \to \mathcal{V}$ a well-behaved modality, and Γ a set of modalities for F. A* correspondence *is a triple of the form (F, τ, Γ) such that the associated coupling-based and codensity liftings coincide, as in:*

$$F^{\downarrow}_{\tau} = F^{\uparrow}_{\Gamma}$$

If $\Gamma = \{\tau\}$ then we refer to this as Kantorovich-Rubinstein duality*, or simply duality.*

Two correspondences (F, τ_1, Γ_1) and (F, τ_2, Γ_2) for the same functor are called *equivalent* when they yield the same liftings:

$$F^{\downarrow}_{\tau_1} = F^{\uparrow}_{\Gamma_1} = F^{\uparrow}_{\Gamma_2} = F^{\downarrow}_{\tau_2}$$

This is denoted by $(F, \tau_1, \Gamma_1) \sim (F, \tau_2, \Gamma_2)$.

When the associated liftings are not equal but one *inequality* holds nonetheless such as

$$F^{\downarrow}_{\tau_1} = F^{\uparrow}_{\Gamma_1} \leq F^{\uparrow}_{\Gamma_2} = F^{\downarrow}_{\tau_2}$$

we write $(F, \tau_1, \Gamma_1) \leq (F, \tau_2, \Gamma_2)$.

4.3 Kantorovich-Rubinstein dualities and tools to get them

The focus of this paper is on general correspondences between coupling-based and codensity liftings, but in some cases there is the stronger property of Kantorovich-Rubinstein duality, meaning correspondences like $(F, \tau, \{\tau\})$. In the remainder of this section we consider a few technical definitions and results that are useful in obtaining such duality results.

First, coupling-based liftings are always smaller than codensity liftings. This is a known result; it is a direct consequence of codensity liftings being initial in a well-chosen category as shown in [10]. It can also be found in [2, Theorem 5.27] in the particular case of the quantale $[0, M]$ as presented in Section 3.

▶ **Proposition 15.** *Given a* **Set** *endofunctor F and an associated well-behaved modality, we have $F^{\downarrow}_{\tau} \leq F^{\uparrow}_{\{\tau\}}$.*

Hence to get a Kantorovich-Rubinstein duality, we only need the other inequality. Two tools are introduced for that: optimal couplings to know the value of coupling-based liftings and optimal functions to know that of codensity liftings.

Given a functor $F\colon \mathbf{Set} \to \mathbf{Set}$, an associated modality τ, a $\mathcal{V}$-pseudometric $d\colon X \times X \to \mathcal{V}$, and $t_1, t_2 \in FX$, an *optimal coupling* is a coupling $t \in F(X \times X)$ of t_1 and t_2 such that

$$F^{\downarrow}_{\tau} d(t_1, t_2) = \tau \circ Fd(t)$$

When for all d, t_1, and t_2 either an optimal coupling exists or no couplings of t_1 and t_2 exist we say that F *has all optimal couplings*. Optimal couplings are well-known in the context of transport theory [27] and in existing proofs of duality for the finite powerset functor [12, 2]. We note that, whenever all elements in a class of functors $\mathcal{F}$ have all optimal couplings, then functors in the closure of $\mathcal{F}$ by coproducts and products do too.

An *optimal function* for a functor F, an associated modality τ, a $\mathcal{V}$-pseudometric $d\colon X \times X \to \mathcal{V}$, and $t_1, t_2 \in FX$ is a morphism in $\mathcal{V}$-**PMet** $f\colon d \to \mathrm{d}_e$ such that:

$$F^{\uparrow}_{\{\tau\}} d(t_1, t_2) = \mathrm{d}_e(\tau \circ Ff(t_1), \tau \circ Ff(t_2))$$

In the context of the general continuous Kantorovich-Rubinstein duality from optimal transport theory such optimal functions do exist (see [27, proof of Theorem 5.10]) but are not explicitly defined. As we will see they exist for all the functors we consider.

The following lemma is useful to design optimal functions. It is well-known in the context of quantale-enriched categories. Instead of defining some pseudometric morphism $f\colon d \to \mathrm{d}_e$ on X as a whole, it can be defined on $Y \subseteq X$ only, first as some morphism $g\colon d \circ i \to \mathrm{d}_e$ and then extended using the lemma. This is useful when only parts of X are of interest, say some subset of it for the powerset functor, or the support of some probability distribution for the distribution functor. The extension of g to f from Y to X is done by giving the greatest values to f on $X \backslash Y$ while ensuring it is a $\mathcal{V}$-pseudometric morphism.

▶ **Lemma 16.** *Let $d\colon X \times X \to \mathcal{V}$ be a $\mathcal{V}$-pseudometric, and $Y \overset{i}{\subseteq} X$. For all $\mathcal{V}$-pseudometric morphisms $g\colon d \circ (i \times i) \to \mathrm{d}_e$ there exists $f\colon d \to \mathrm{d}_e$ s.t. $g = f \circ i$ and f is the least such morphism.*

5 Correspondences through coproducts and products

In this section we obtain new correspondences between coupling-based and codensity liftings, for polynomial functors. More specifically, given a set of functors $\mathcal{F}$ and associated correspondences (F, τ_F, Γ_F), these are extended to correspondences for the coproduct (Section 5.1) and product (Section 5.2) of the underlying functors. In Section 5.3 we study correspondences for constant and identity functors. Finally in Section 5.4 we combine these results to describe several collections of functors for which we have correspondences. Throughout this section we fix a quantale $\mathcal{V}$.

5.1 Coproduct functors

We start by showing that correspondences are closed under coproducts. Given sets S_i we write κ_j for the jth *coprojection* $\kappa_j\colon S_j \to \coprod S_i$.

▶ **Proposition 17.** *Given correspondences $(F_i, \tau_i\colon F_i\mathcal{V} \to \mathcal{V}, \Gamma_i)$ there is a correspondence*

$$\left(\coprod F_i, [\tau_i], \bigcup \overline{\Gamma_i}\right) \qquad \text{where } \overline{\Gamma_i} = \{\overline{\tau} \mid \tau \in \Gamma_i\} \cup \{\tau_{\top,i}\};$$

the map $[\tau_i]\colon \coprod F_i\mathcal{V} \to \mathcal{V}$ is the cotupling of the individual modalities; for $\tau \in \Gamma_i$,

$$\overline{\tau}(x) = \begin{cases} \tau(y) \text{ when there exists } y \in F_i\mathcal{V} \text{ and } x = \kappa_i y \\ \top \text{ otherwise} \end{cases}$$

and finally, $\tau_{\top,i}(x) = \begin{cases} \top \text{ when there exists } y \in F_i\mathcal{V} \text{ and } x = \kappa_i y \\ \bot \text{ otherwise} \end{cases}$.

When there are no couplings for two given elements, as it happens for instance for elements in different components of a coproduct, the coupling-based lifting always gives the value $\bot$. The modalities $\tau_{\top,i}$ are there to ensure that the codensity lifting also gives the value $\bot$ in this case. The other modalities are just the extensions of the modalities from the components of the coproduct to the coproduct itself.

▶ Remark 18. Proposition 17 gives a correspondence but not a duality, in the sense of Definition 14, that is, the correspondence relates a single modality for the coupling-based lifting to a set of modalities for the codensity lifting. This is the case even if the correspondences of the component functors F_i are dualities.

As it turns out, the inverse process of going from a correspondence at the level of the coproduct to correspondences for its components is also possible:

▶ **Proposition 19.** *If there is a correspondence* $(\coprod F_i, \tau, \Gamma)$*, then there are correspondences* $(F_i, \tau \circ \kappa_i, \Gamma \circ \kappa_i)$ *for each* i*.*

Going back and forth between the components of a coproduct and the coproduct itself using the results above gives back the same correspondences:

▶ **Proposition 20.** *Whenever one of the correspondences on the left exists we also have the corresponding equivalence:*

$$\begin{aligned} (F_i, \tau_i, \Gamma_i) &\sim \left(F_i, [\tau_j] \circ \kappa_i, \bigcup \overline{\Gamma_j} \circ \kappa_i\right) \text{ and} \\ \left(\coprod F_i, \tau, \Gamma\right) &\sim \left(\coprod F_i, [\tau \circ \kappa_i], \bigcup \overline{\Gamma \circ \kappa_i}\right) \end{aligned}$$

This shows that all correspondences on the coproduct arises as in Proposition 17.

5.2 Product functors

We turn to the construction of correspondences for products from ones on their components. For this, we ask for some kind of distributivity condition on the quantale. In fact, the case of products is more involved than that of coproducts. Each of the three propositions for coproducts above has a version for products, but each gets their own specific restrictions in the form of conditions on the quantale and modalities or even on the statement itself.

There are several possibilities depending on the number of couplings and on whether we want finite or arbitrary products. Below, we say that a functor F has *finite couplings* if, for any $t_1, t_2 \in FX$, the set $\Omega(t_1, t_2) \subseteq F(X \times X)$ of couplings is finite.

▶ **Proposition 21.** *Let* (F, τ_F, Γ_F) *and* (G, τ_G, Γ_G) *be correspondences. If one of the following conditions holds:*

1. *F and G have finite couplings and $\mathcal{V}$ is distributive, meaning for all $x, y, z \in \mathcal{V}$, we have that $(x \vee z) \wedge (y \vee z) = (x \wedge y) \vee z$; or,*
2. *$\mathcal{V}$ is join-infinite distributive: for all $x \in \mathcal{V}$ and $V \subseteq \mathcal{V}$, $x \wedge \bigvee V = \bigvee\{x \wedge v \mid v \in V\}$;*

then we have a correspondence $(F \times G, (\tau_F \circ \pi_1) \wedge (\tau_G \circ \pi_2), (\Gamma_F \circ \pi_1) \cup (\Gamma_G \circ \pi_2))$*.*

Under stronger distributivity conditions, this can be extended to infinite products.

▶ **Proposition 22.** *Let $(F_i, \tau_{F_i}, \Gamma_{F_i})$ be correspondences. If one of the following holds:*

1. *F_i has finite couplings and $\mathcal{V}$ is meet-infinite distributive, meaning that for all $x \in \mathcal{V}$ and $V \subseteq \mathcal{V}$, $x \vee \bigwedge V = \bigwedge\{x \vee v \mid v \in V\}$; or,*
2. *$\mathcal{V}$ is completely distributive, meaning that for all sets $K \subseteq I \times J$ such that K projects onto I, and any subset $\{x_{ij} \mid (i,j) \in K\} \subseteq \mathcal{V}$, $\bigwedge_{i\in I}\left(\bigvee_{j\in K(i)} x_{ij}\right) = \bigvee_{f\in A}\left(\bigwedge_{i\in I} x_{if(i)}\right)$ where $A = \{f\colon I \to J \mid \forall i \in I,\ f(i) \in K(i)\}$;*

then we have a correspondence $(\prod F_i, \bigwedge(\tau_{F_i} \circ \pi_i), \bigcup(\Gamma_{F_i} \circ \pi_i))$.

▶ **Example 23.** The quantales 2 and $[0, M]$ from Example 2 are both completely distributive.

Replacing the infinite distributivity conditions on the quantale by countably infinite versions directly gives similar statements for countable products.

In order for the inverse process from products to components to be uniquely defined we first need to ensure that the functor has been decomposed as much as possible using coproducts following Propositions 17, 19, and 20. The following well-known lemma is useful for this (see [22, Proposition 1.3.12]):

▶ **Lemma 24.** *Let $F\colon \mathbf{Set} \to \mathbf{Set}$ be a functor. There is a family of functors $\{F_i\}_{i\in F\{\top\}}$ indexed by $F\{\top\}$ such that $F \simeq \coprod F_i$, and if $F_i = \coprod G_j$ then all G_j but one are the empty functor sending any set to the empty set.*

Having characterised how functors decompose along coproducts and how correspondences work with regard to such decompositions, we can now assume functors not to be writable as non-trivial coproducts meaning that if a functor can be written as a coproduct, all but one of the components must be the empty functor. Furthermore we assume for convenience that functors are not empty functors. Those two hypotheses can be formalised as follows: by the previous lemma, functors send $\{\top\}$ and by extension any singleton to a singleton. Hence, given τ well-behaved, with $i\colon \{\top\} \hookrightarrow \mathcal{V}$, $Fi(F\{\top\}) = \tau^{-1}(\top)$ must be a singleton.

▶ **Proposition 25.** *Given a correspondence $\left(\prod_{i\in I} F_i, \tau, \Gamma\right)$, we again have correspondences $\left(F_i, \tau_{|i}, \Gamma_{|i}\right)$ where $\tau_{|i}(x) = \tau(\top_1, \ldots, \top_{i-1}, x, \top_{i+1}, \ldots, \top_I)$, with $(\top_j)_j = \tau^{-1}(\top)$ and similarly for modalities in $\Gamma_{|i}$.*

In a similar fashion as for the coproduct, we would like that going back and forth between products and their components, following Proposition 21 or 22 and Proposition 25, preserves correspondences. We can only do that partially:

▶ **Proposition 26.** *Whenever the correspondences on the left exist and we have the right distributivity conditions from Proposition 21 or 22 for the correspondence on the right to exist, we have the following equivalence and inequality:*

$$\begin{aligned} (F_i, \tau_i, \Gamma_i) &\sim \left(F_i, \left(\bigwedge(\tau_j \circ \pi_j)\right)_{|i}, \left(\bigcup(\Gamma_j \circ \pi_j)\right)_{|i}\right) \\ \left(\prod F_i, \tau, \Gamma\right) &\leq \left(\prod F_i, \bigwedge(\tau_{|i} \circ \pi_i), \bigcup(\Gamma_{|i} \circ \pi_i)\right) \end{aligned}$$

The inequality comes from the following aspect of the proof. When using the correspondences on the components of a product of functors to retrieve a correspondence on the product itself, a choice is made to build modalities for the product. The natural choice of using a meet ensures the inequality while there is no obvious way to get an equivalence of correspondences.

5.3 Constant and identity functors

In this subsection we give correspondence results for constant functors, and for the identity functor. For constant functors, the essence is given by the constant-to-**1** functor.

▶ **Proposition 27.** *The triple* $(\mathbf{1}, \tau, \Gamma)$ *with the functor* $\mathbf{1}$ *sending any set to a fixed singleton is a correspondence if and only if* $\Gamma = \{\tau\}$ *and* τ *is the constant to* $\top$ *modality.*

Proof. We write $\mathbf{1} = \{*\}$.

$\Rightarrow$: as modalities $\tau\colon \{*\} \to \mathcal{V}$ are assumed well-behaved, given the inclusion $i\colon \{\top\} \hookrightarrow \mathcal{V}$ we must have $\mathbf{1}i[\mathbf{1}\{\top\}] = \tau^{-1}(\top)$ which translates to $\tau^{-1}(\top) = \{*\}$.

$\Leftarrow$: both liftings lift all pseudometrics to the one constant to $\top$. ◀

Next, we extend this result to arbitrary constant functors. There is only one correspondence result in this case.

▶ **Corollary 28.** *For* A *a set, denoting by* A *the associated* constant functor *mapping all sets to* A*, there is a correspondence* $(A, \tau_\top, \{\tau_a \mid a \in A\})$ *with* $\tau_\top$ *the constant to* $\top$ *modality and*

$$\tau_a(b) = \begin{cases} \top \text{ when } b = a \\ \bot \text{ otherwise} \end{cases}$$

Furthermore, any other correspondence (A, τ, Γ) *is equivalent to* $(A, \tau_\top, \{\tau_a \mid a \in A\})$.

Proof. This is a direct consequence of Propositions 17, 19, 20 and 27 viewing the functor constant and equal to A as a coproduct:

$$A \simeq \coprod_{a \in A} \mathbf{1}$$ ◀

We give a general duality result for the *identity functor* Id acting as the identity on both sets and functions, generalising [2, Example 5.29]. An instance of well-behaved modalities for the identity functor was discussed in Example 8.

▶ **Proposition 29.** *For any well-behaved modality* $\tau\colon \mathcal{V} \to \mathcal{V}$ *for the identity functor, there is a correspondence* $(\mathrm{Id}, \tau, \{\tau\})$.

Proof. By Proposition 15 we need only prove

$$\mathrm{Id}_\tau^\downarrow \geq \mathrm{Id}_\tau^\uparrow$$

Let X be a set, $d\colon X \times X \to \mathcal{V}$ a $\mathcal{V}$-pseudometric, and $x, y \in X$. There is exactly one coupling of x with y given by (x, y). In particular it is optimal:

$$\mathrm{Id}_\tau^\downarrow d(x, y) = \tau(d(x, y))$$

Furthermore,

$$\mathrm{Id}_\tau^\uparrow d(x, y) = \bigwedge_{f\colon d \to \mathrm{d}_e} \mathrm{d}_e(\tau \circ f(x), \tau \circ f(y))$$

Using Lemma 16, we define $f\colon d_{|\{x\}} \to \mathrm{d}_e$ by $f(x) = \top$ and extend it to a morphism $f\colon d \to \mathrm{d}_e$. This gives $f(y) = d(x, y)$ so that $\mathrm{d}_e(\tau \circ f(x), \tau \circ f(y)) = \tau(d(x, y))$. Hence

$$\mathrm{Id}_\tau^\uparrow d(x, y) \leq \tau \circ d(x, y) = \mathrm{Id}_\tau^\downarrow d(x, y)$$

ending the proof. Note that f is *a fortiori* an optimal function. ◀

5.4 Putting it together: correspondences for polynomial functors

Having seen correspondences for products, coproducts, the identity functor, and constant functors, we combine these results to obtain a "grammar" of functors with correspondences.

We start without products. Using Propositions 17 and 29 as well as Corollary 28 gives correspondences for functors in the following grammar of **Set** endofunctors:

$$F ::= A \mid \mathrm{Id}_\tau \mid \coprod F_i$$

where τ indicates choices of well-behaved modalities in the case of the identity functor. Furthermore by Propositions 20 and 29 and Corollary 28, any coupling-based lifting of a functor in this grammar is equivalent to a correspondence given by our construction.

Observing that for a set A, $\coprod_{a\in A} F \simeq A \times F$, the grammar can equivalently be defined as

$$F ::= A \mid \mathrm{Id}_\tau \mid A \times F \mid \coprod F_i$$

where τ is a well-behaved modality for the identity functor. We can also note that these functors are exactly those isomorphic to $A + B \times \mathrm{Id}$ for some sets A and B.

▶ **Example 30** (Discounting on streams)**.** Consider the stream functor $X \mapsto A \times X$ for some alphabet A. The associated final coalgebra is the set of streams A^ω. The results above tell us that to get a correspondence we need one well-behaved modality $\tau\colon \mathcal{V} \to \mathcal{V}$ per letter $a \in A$ for the codensity lifting, through the isomorphism $A \times \mathrm{Id} \simeq \coprod_{a\in A} \mathrm{Id}$: we have a way of computing distance between streams in a specific way for each letter in A. When $\mathcal{V} = ([0, M], \geq, +)$ for some real number $M \in (0, \infty]$, choosing $\tau_a(x) = c \cdot x$, the constant c is then a discount factor allowing one to give more value to the first letters of a stream. We can choose a different constant c_a for each letter a giving different values to different letters. We can also go further: if we do not want a linear discount we can use arbitrary well-behaved modalities for the identity functor. On the quantale $([0, M], \geq, +)$, those are exactly sub-additive monotone maps mapping 0 to 0.

We also get correspondences for the following grammar of *simple polynomial functors*:

$$F ::= A \mid \mathrm{Id}_\tau \mid A \times F \mid \coprod F_i \mid \prod F_i$$

These functors all have finite couplings. Following Propositions 21 and 22 we can either get finite or arbitrary products in the grammar above by assuming $\mathcal{V}$ to be distributive or meet-infinite distributive respectively.

▶ **Example 31.** We retrieve the correspondence of Section 2 as the particular case of the quantale $([0, M], \geq, +)$, the functor $2 \times \mathrm{Id}^A$, and indexed modalities τ for the identity functors always mapping $x \in [0, M]$ to $c \cdot x$ for some $c \in [0, 1)$. Replacing $[0, M]$ by the usual Boolean quantale $2 = \{\top, \bot\}$ and indexed modalities by the identity functions gives a correspondence equivalent to the usual relation lifting [14] associated to behavioural equivalence of DFA.

6 Dualities for the powerset and probability distribution functors

We give duality results for the powerset functor mapping a set X to the set of its subsets $\mathcal{P}X$, recovering a result from the literature [2, 12] reproduced here as Corollary 35, and complementing it with a duality result where $\mathcal{V}$ is a total order (Corollary 34).

▶ Remark 32. The non-empty powerset functor $\mathcal{P}_{\geq 1}$ mapping a set X to the set of its non-empty subsets might have been chosen instead of $\mathcal{P}$. Note that $\mathcal{P} = \mathbf{1} + \mathcal{P}_{\geq 1}$. Up-to equivalence, as a consequence of Propositions 17, 19 and 27 a correspondence for $\mathcal{P}$ is exactly given by one for $\mathcal{P}_{\geq 1}$ and conversely.

Given $T_1, T_2 \in \mathcal{P}X$, $t_1 \in T_1$, and a $\mathcal{V}$-pseudometric $d\colon X \times X \to \mathcal{V}$, we write M_{t_1} for the *set of maximal elements* of $\{d(t_1, t_2) \mid t_2 \in T_2\}$, and similarly for elements $t_2 \in T_2$. Expressing a coupling-based lifting as a codensity one in a correspondence implies turning a join in a meet. The following proposition gives a condition allowing one to do that in the case of the powerset functor by making a modality absorb a meet.

▶ **Proposition 33.** *Let $\tau\colon \mathcal{P}\mathcal{V} \to \mathcal{V}$ be a well-behaved modality. If for every $\mathcal{V}$-pseudometric $d\colon X \times X \to \mathcal{V}$ and sets $T_1, T_2 \in \mathcal{P}X$ the following equality holds*

$$\tau\left\{\bigvee M_{t_1} \mid t_1 \in T_1\right\} \bigwedge \tau\left\{\bigvee M_{t_2} \mid t_2 \in T_2\right\} = \tau\left(\left(\bigcup_{t_1 \in T_1} M_{t_1}\right) \cup \left(\bigcup_{t_2 \in T_2} M_{t_2}\right)\right)$$

then there is a duality $(\mathcal{P}, \tau, \{\tau\})$.

▶ **Corollary 34.** *Let $\tau\colon \mathcal{P}\mathcal{V} \to \mathcal{V}$ be a well-behaved modality. If the order on $\mathcal{V}$ is total then there is a duality $(\mathcal{P}, \tau, \{\tau\})$.*

▶ **Corollary 35.** *We assume $\mathcal{V}$ to be completely distributive. We have duality for the coupling-based and codensity liftings of the powerset functor, both along the meet modality which maps a subset of $\mathcal{V}$ to the meet of its elements.*

For the probability distribution functor we fix a constant $M \in (0, \infty]$ and use the quantale $([0, M], \geq, +)$. The following (Kantorovich-Rubinstein) duality result is well-known (see, e.g., [27, Theorem 5.10] for a general proof in the continuous case and [16, Appendix A] for the discrete finite case):

▶ **Proposition 36.** *There is a duality $(\mathcal{D}, \mathbb{E}, \{\mathbb{E}\})$ with $\mathcal{D}$ the finite probability distribution functor and $\mathbb{E}$ giving the expectation of probability distributions.*

We extend this result slightly to allow post-composition by well-behaved modalities:

▶ **Proposition 37.** *Let $\tau\colon [0, M] \to [0, M]$ be a well-behaved modality. If τ is additive, meaning that for all $x, y \in [0, M]$, $\tau(x + y) = \tau(x) + \tau(y)$, then there is a duality $(\mathcal{D}, \tau \circ \mathbb{E}, \{\tau \circ \mathbb{E}\})$.*

7 Correspondences for grammars of functors

In this section we combine the results from the previous two sections, allowing the construction of correspondences for certain classes of functors including powerset, distribution, identity and constant functors, as well as products and coproducts thereof.

Whenever $\mathcal{V}$ is totally ordered and completely distributive, using results from Sections 5 and 6 we can construct correspondences for the following grammar of functors:

$$F ::= A \mid \mathrm{Id}_\tau \mid \mathcal{P}_\tau \mid \prod F_i \mid \coprod F_i$$

where the indices τ are well-behaved modalities for either the identity or powerset functors.

Noting that $\mathcal{P}\left(\coprod_{i \in I} F_i\right) \cong \prod_{i \in I} \mathcal{P}(F_i)$, this grammar can be reformulated as follows:

$$G ::= A \mid \mathrm{Id}_\tau \mid A \times G \mid \coprod G$$

$$F ::= A \mid \mathrm{Id}_\tau \mid \mathcal{P} \circ G \mid \prod F_i \mid \coprod F_i$$

where the indices of the form τ indicate a choice of possible well-behaved modalities for the identity functor when appearing in F and for the powerset functor when appearing in G.

▶ **Example 38.** In a similar fashion as in Example 31, considering the completely distributive quantale $([0, M], \geq, +)$, the functor $2 \times \mathcal{P}^A$ which is associated to *non-deterministic automata*, and indexed modalities for the powerset functors to be all mapping finite subsets $V \subset [0, M]$ to $c \cdot \max(V)$ for a fixed $c \in [0, 1)$, we get a lifting associated to a shortest-distinguishing-word-distance for non-deterministic systems as is detailed in Section 2 for DFA.

▶ Remark 39. This grammar with finite products only defines a fragment of finitary Kripke polynomial functors as defined in [14] which correspond to replacing $\mathcal{P}(G)$ by $\mathcal{P}(F)$, and having only finite products but including arbitrary exponents.

▶ Remark 40. Using the identity modality for the identity functors and the meet modality for the powerset functors, the modalities given by this construction for the coupling-based liftings are exactly the *canonical evaluation maps* as defined in [6].

In the particular case of $\mathcal{V} = [0, M]$ we can extend the above grammar with the finite probability distribution functor and get correspondence results for the following:

$$F ::= A \mid \mathrm{Id}_\tau \mid \mathcal{P}_\tau \mid \mathcal{D}_\tau \mid \prod F_i \mid \coprod F_i$$

where the indices τ indicate choices of well-behaved modalities for either the identity, powerset, or distribution functors, furthermore of a modality of the form $\tau \circ \mathbb{E}$ for the latter as described in Proposition 37.

Observe that $\mathcal{D}\left(\coprod_{i \in I} F_i\right) \cong \mathcal{D}(I) \times \prod_{i \in I} \mathcal{D}(F_i)$ so that the grammar above can be expressed as follows:

$$G ::= A \mid \mathrm{Id}_\tau \mid A \times G \mid \coprod G$$
$$F ::= A \mid \mathrm{Id}_\tau \mid \mathcal{P} \circ G \mid \mathcal{D} \circ G \mid \prod F_i \mid \coprod F_i$$

where the indices τ indicate choices of possible well-behaved modalities for the identity functor in Id_τ and for the powerset or distribution functors when appearing in G for $\mathcal{P} \circ G$ and $\mathcal{D} \circ G$ respectively.

▶ **Example 41.** We fix $M = 1$ so that $\mathcal{V} = [0, 1]$. Consider the functor $\mathcal{D}(\mathbf{1} + \mathrm{Id})^A$ for a set of labels A and $\mathbf{1}$ a singleton as in Proposition 27. It is associated to *labelled Markov processes*. To get a correspondence we need one modality $\tau_a \colon \mathcal{D}[0, 1] \to [0, 1]$ per label $a \in A$. By Proposition 37 we can take them all to be $\tau_a(\mu) = c \cdot \mathbb{E}(\mu)$ with $c \in [0, 1)$ a fixed constant. The resulting correspondence is associated to the usual metrics for labelled Markov processes as introduced in [8]. This is a direct consequence of [26, Proposition 29] which expresses the associated lifting for one label in a "codensity-like" manner.

8 A counter-example: conditional transition systems

We now highlight an example where a correspondence can *not* be provided by our construction: some codensity lifting may not be obtained as a coupling-based lifting.

A *conditional transition system* (CTS) over an alphabet A and a partially ordered set of conditions $(\mathcal{L}, \leq)$ is a tuple $(X, A, \mathcal{L}, \delta)$ where X is a set of states and the transition map $\delta \colon X \times A \to ((\mathcal{L}, \leq) \to (\mathcal{P}(X), \supseteq))$ associates to each pair $(x, a) \in X \times A$ a monotone function $\delta_{x,a}$ from the poset of conditions to $\mathcal{P}(X)$. For the associated functor to live in **Set** we restrict to the CTSs for which $\mathcal{L}$ is trivially ordered by $\forall x, y \in \mathcal{L}, x \leq y$. Hence any map $\delta_{x,a} \colon \mathcal{L} \to \mathcal{P}(X)$ is monotone, and CTSs are exactly coalgebras for the functor $F = \mathcal{P}(-)^{A \times \mathcal{L}}$ (see [4]).

The associated bisimulations, not detailed here, are related to a lifting $\overline{F}\colon \mathcal{P}(\mathcal{L})\textbf{-PMet} \to \mathcal{P}(\mathcal{L})\textbf{-PMet}$ of F defined by

$$\begin{aligned}\overline{F}r(T_1,T_2) = \{l \in \mathcal{L} \mid \forall a \in A, &\forall (x,y) \in T_1(l,a) \times T_2(l,a),\\ &\exists (x',y') \in T_1(l,a) \times T_2(l,a),\\ &(l \in r(x,y')) \text{ and } (l \in r(x',y))\}\end{aligned}$$

on objects and $\overline{F}f = Ff$ on maps.

▶ **Proposition 42.** *For $a \in A$ consider a modality $\tau_a\colon \mathcal{P}(\mathcal{P}(\mathcal{L}))^{\mathcal{L}\times A} \to \mathcal{P}(\mathcal{L})$, defined by:*

$$\forall f\colon \mathcal{L}\times A \to \mathcal{P}(\mathcal{P}(\mathcal{L})), \tau_a(f) = \{l \in \mathcal{L} \mid l \in \cap f(l,a)\}$$

The codensity lifting defined by the family $(\tau_a)_{a\in A}$ of modalities is equal to $\overline{F}$.

▶ **Proposition 43.** *If A is non-empty and $|\mathcal{L}| \geq 2$ then there is no well-behaved modality $\tau\colon \mathcal{P}(\mathcal{P}(\mathcal{L}))^{\mathcal{L}\times A} \to \mathcal{P}(\mathcal{L})$ such that the resulting coupling-based lifting is equal to $\overline{F}$.*

Proof. Let $a \in A$ be some letter and $c_1, c_2 \in \mathcal{L}$ be two distinct conditions. Consider $d\colon X \times X \to \mathcal{P}(\mathcal{L})$ constant to $\top$, that is, $\mathcal{L}$. It is obviously a $\mathcal{P}(\mathcal{L})$-pseudometric. Finally we consider $T_1, T_2 \in \mathcal{P}(X)^{\mathcal{L}\times A}$ such that $T_1(c_1,a) = \emptyset$ and $T_1(c_2,a) = X$, and $T_2(c_1,a) = T_2(c_2,a) = X$. Directly, because $T_1(c_1,a) = \emptyset$ but $T_2(c_1,a) \neq \emptyset$ there are no couplings of T_1 and T_2, and for all well-behaved modalities, $F^{\downarrow}_{\tau}d(T_1,T_2) = \emptyset$. On the other hand,

$$\forall (x,y) \in T_1(c_2,a)\times T_2(c_2,a),\ \exists (x',y') \in T_1(c_2,a)\times T_2(c_2,a),\ (l \in d(x,y')) \text{ and } (l \in d(x',y))$$

Hence $c_2 \in \overline{F}d(T_1,T_2) \neq \emptyset$. ◀

Hence, conditional transition systems give a limitation to the correspondence results provided above. Note however that the associated functor has a correspondence induced by a grammar above. The reason it is not equivalent to the lifting $\overline{F}$ is that it considers the set $\mathcal{L}\times A$ as being a set of actions, while we want conditions in $\mathcal{L}$ to be independent of one another and compared separately. Indeed, conditions are fixed throughout the execution of a CTS whereas actions may change at each step.

9 Conclusions and future work

We have studied correspondences between coupling-based and codensity liftings, moving from the classical Kantorovich-Rubinstein duality for distributions to different types of endofunctors on **Set**. In particular, we have shown that such types of correspondences are closed under coproducts and products and used that to provide explicit correspondences for several grammars of functors, including polynomial functors with the possibility of extending them using the powerset and the probability distribution functors. This instantiates to usual liftings of functors associated to (non)deterministic finite automata, or labelled Markov processes, both with discount.

In [10] the authors have shown that on an abstract level all coupling-based liftings are in fact codensity liftings, implying that all coupling-based liftings have some associated correspondences. Section 8 shows that the converse does not hold by providing an example of lifting that arises as a codensity lifting but not as a coupling-based one. Our work proves that correspondences for coproducts of functors are characterised by ones for the components of the coproducts, but gives no such result for products of functors. For example we could consider

the coupling-based lifting of the *diagonal functor* $\Delta\colon X \mapsto X \times X$ along the well-behaved modality $\otimes\colon \mathcal{V} \times \mathcal{V} \to \mathcal{V}$. The question of whether it arises from coupling-based liftings of the identity functors, and the problem of relating it to a codensity lifting in a correspondence remain open. Note however that if coupling-based liftings require their modalities to be well-behaved, codensity liftings, as defined in [10] need no such assumption and can be defined along sets of any modalities. Hence it is possible that to see the coupling-based lifting of Δ along $\otimes$ as a codensity lifting it is needed to consider general modalities for the latter.

In Section 8 we have seen that a certain lifting for conditional transition systems can not be obtained as a coupling-based lifting. This leads to the question of whether the definition of coupling-based liftings could be extended somehow to encompass this example and obtain a correspondence with the existing codensity lifting.

References

1 Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. On-the-fly computation of bisimilarity distances. *Log. Methods Comput. Sci.*, 13(2), 2017. doi:10.23638/LMCS-13(2:13)2017.

2 Paolo Baldan, Filippo Bonchi, Henning Kerstan, and Barbara König. Coalgebraic behavioral metrics. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:20)2018.

3 Harsh Beohar, Barbara König, Sebastian Küpper, and Alexandra Silva. Conditional transition systems with upgrades. *Sci. Comput. Program.*, 186, 2020. doi:10.1016/J.SCICO.2019.102320.

4 Harsh Beohar, Barbara König, Sebastian Küpper, Alexandra Silva, and Thorsten Wißmann. A coalgebraic treatment of conditional transition systems with upgrades. *Log. Methods Comput. Sci.*, 14(1), 2018. doi:10.23638/LMCS-14(1:19)2018.

5 Filippo Bonchi, Barbara König, and Daniela Petrisan. Up-to techniques for behavioural metrics via fibrations. In Sven Schewe and Lijun Zhang, editors, *29th International Conference on Concurrency Theory, CONCUR 2018, September 4-7, 2018, Beijing, China*, volume 118 of *LIPIcs*, pages 17:1–17:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICS.CONCUR.2018.17.

6 Filippo Bonchi, Barbara König, and Daniela Petrisan. Up-to techniques for behavioural metrics via fibrations. *Math. Struct. Comput. Sci.*, 33(4-5):182–221, 2023. doi:10.1017/S0960129523000166.

7 Y. Brenier, U. Frisch, M. Hénon, G. Loeper, S. Matarrese, R. Mohayaee, and A. Sobolevskiĭ. Reconstruction of the early Universe as a convex optimization problem. *Monthly Notices of the Royal Astronomical Society*, 346(2):501–524, December 2003. doi:10.1046/j.1365-2966.2003.07106.x.

8 Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled markov processes. *Theor. Comput. Sci.*, 318(3):323–354, 2004. doi:10.1016/J.TCS.2003.09.013.

9 Alessandro Giacalone, Chi-Chang Jou, and Scott A. Smolka. Algebraic reasoning for probabilistic concurrent systems. In Manfred Broy and Cliff B. Jones, editors, *Programming concepts and methods: Proceedings of the IFIP Working Group 2.2, 2.3 Working Conference on Programming Concepts and Methods, Sea of Galilee, Israel, 2-5 April, 1990*, pages 443–458. North-Holland, 1990.

10 Sergey Goncharov, Dirk Hofmann, Pedro Nora, Lutz Schröder, and Paul Wild. Kantorovich functors and characteristic logics for behavioural distances. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures - 26th International Conference, FoSSaCS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2023, Paris, France, April 22-27, 2023, Proceedings*, volume 13992 of *Lecture Notes in Computer Science*, pages 46–67. Springer, 2023. doi:10.1007/978-3-031-30829-1_3.

11 Dirk Hofmann. Topological theories and closed objects. *Advances in Mathematics*, 215(2):789–824, 2007. doi:10.1016/j.aim.2007.04.013.

12 Dirk Hofmann and Pedro Nora. Hausdorff coalgebras. *Appl. Categorical Struct.*, 28(5):773–806, 2020. doi:10.1007/S10485-020-09597-8.

13 Samuel Humeau, Daniela Petrisan, and Jurriaan Rot. Correspondences between codensity and coupling-based liftings, a practical approach, 2024. Version of this paper with the appendix. URL: https://hal.science/hal-04789352.

14 Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016. doi:10.1017/CBO9781316823187.

15 Bart Jacobs. Drawing from an urn is isometric. In Naoki Kobayashi and James Worrell, editors, *Foundations of Software Science and Computation Structures - 27th International Conference, FoSSaCS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part I*, volume 14574 of *Lecture Notes in Computer Science*, pages 101–120. Springer, 2024. doi:10.1007/978-3-031-57228-9_6.

16 Bart Jacobs. Drawing with distance, 2024. arXiv:2405.18182, doi:10.48550/arXiv.2405.18182.

17 Leonid Kantorovich. On the translocation of masses (in Russian). *Doklady Akademii Nauk*, 37(7–8):227–229, 1942. Translated to English in Management Science, 5, 1 (1958).

18 Leonid Kantorovich and Gennady S. Rubinstein. On a space of totally additive functions (in Russian). *Vestnik Leningrad. Univ*, 13(7):52–59, 1958.

19 Shin-ya Katsumata, Tetsuya Sato, and Tarmo Uustalu. Codensity lifting of monads and its dual. *Log. Methods Comput. Sci.*, 14(4), 2018. doi:10.23638/LMCS-14(4:6)2018.

20 Yuichi Komorida, Shin-ya Katsumata, Clemens Kupke, Jurriaan Rot, and Ichiro Hasuo. Expressivity of quantitative modal logics : Categorical foundations via codensity and approximation. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–14. IEEE, 2021. doi:10.1109/LICS52264.2021.9470656.

21 Barbara König and Christina Mika-Michalski. (metric) bisimulation games and real-valued modal logics for coalgebras. In Sven Schewe and Lijun Zhang, editors, *29th International Conference on Concurrency Theory, CONCUR 2018, September 4-7, 2018, Beijing, China*, volume 118 of *LIPIcs*, pages 37:1–37:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICS.CONCUR.2018.37.

22 Robert Paré. Taut functors and the difference operator, 2024. arXiv:2407.21129.

23 Lutz Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. In Vladimiro Sassone, editor, *Foundations of Software Science and Computational Structures, 8th International Conference, FOSSACS 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005, Proceedings*, volume 3441 of *Lecture Notes in Computer Science*, pages 440–454. Springer, 2005. doi:10.1007/978-3-540-31982-5_28.

24 David Sprunger, Shin-ya Katsumata, Jérémy Dubut, and Ichiro Hasuo. Fibrational bisimulations and quantitative reasoning: Extended version. *J. Log. Comput.*, 31(6):1526–1559, 2021. doi:10.1093/LOGCOM/EXAB051.

25 Franck van Breugel. Probabilistic bisimilarity distances. *ACM SIGLOG News*, 4(4):33–51, 2017. doi:10.1145/3157831.3157837.

26 Franck van Breugel and James Worrell. A behavioural pseudometric for probabilistic transition systems. *Theor. Comput. Sci.*, 331(1):115–142, 2005. doi:10.1016/J.TCS.2004.09.035.

27 Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009. doi:10.1007/978-3-540-71050-9.

A Complete Inference System for Probabilistic Infinite Trace Equivalence

Corina Cîrstea
University of Southampton, UK

Lawrence S. Moss
Indiana University, Bloomington, IN, USA

Victoria Noquez
Saint Mary's College of California, Moraga, CA, USA

Todd Schmid
Bucknell University, Lewisburg, PA, USA

Alexandra Silva
Cornell University, Ithaca, NY, USA

Ana Sokolova
Paris Lodron University of Salzburg, Austria

Abstract

We present the first sound and complete axiomatization of *infinite* trace semantics for generative probabilistic transition systems. Our approach is categorical, and we build on recent results on proper functors over convex sets. At the core of our proof is a characterization of infinite traces as the final coalgebra of a functor over convex algebras. Somewhat surprisingly, our axiomatization of infinite trace semantics coincides with that of finite trace semantics, even though the techniques used in the completeness proof are significantly different.

2012 ACM Subject Classification Theory of computation → Logic; Theory of computation → Formal languages and automata theory

Keywords and phrases Coalgebra, infinite trace, semantics, logic, convex sets

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.30

Funding This material is based upon work supported by the National Science Foundation under Grant No. DMS-1928930, while the authors were in residence at the Mathematical Sciences Research Institute in Berkeley, California, during the Summer Research in Mathematics program of 2024.
Corina Cîrstea: partly supported by the Leverhulme Trust Research Project Grant RPG-2020-232.
Lawrence S. Moss: Lawrence S. Moss was supported by grant #586136 from the Simons Foundation.
Alexandra Silva: ERC grant Autoprobe (no. 101002697). This work was done in part while the author was visiting the Simons Institute for the Theory of Computing.

Acknowledgements We thank Wojtek Rozowski for insightful discussions on related topics and the anonymous reviewers for helpful suggestions that improved the material presented in the paper. We thank the National Science Foundation and the Simons Laufer Mathematical Sciences Institute for their support of our work.

1 Introduction

Probabilistic transition systems have been studied in the semantics and verification literature for decades. There are many variants, from the simplest Rabin model [16] to systems that encompass multiple layers of randomized and non-deterministic choice. A good overview of existing systems and an expressiveness hierarchy was provided in [26, 3].

One important class of probabilistic systems are so-called *generative* probabilistic transition systems (GPTS). These are much like ordinary (nondeterministic) labelled transition systems, but each state is assigned a (sub-)probability distribution over outgoing transitions instead of a set of outgoing transitions. Every state in a GPTS *generates* a probability distribution of traces. The traces generated can be *finite* or *infinite* depending whether the GPTS models *explicit termination*.

In this paper, we will consider GPTS *without* explicit termination, also widely known in the literature as Labelled Markov Chains (LMCs), and therefore we are only interested in including infinite traces in the semantics. That is, each state of an LMC we consider generates a probability distribution on infinite traces (a.k.a. streams). The main goal of this paper is to provide an axiomatic characterization of when two states in these LMCs generate the same probability distribution on streams. We provide a syntax and an inference system to reason about distributions on streams generated by a state of an LMC, and prove that the axiomatization is both sound and complete.

Axiomatizing trace distribution semantics is difficult in general, and this is made more challenging by the presence of infinite traces. One of the seminal works on axiomatizing probabilistic behaviours is due to Stark and Smolka [29], but they studied probabilistic bisimilarity (in the sense of [11]), which is a finer equivalence than trace distributions. A decade later [25], Silva and Sokolova showed that adding one extra axiom to Stark and Smolka's axiomatization of probabilistic bisimilarity was enough to obtain a sound and complete axiomatization of *finite* trace distribution equivalence. At the core of Silva and Sokolova's completeness result was the observation that finite trace distribution equivalence coincides with bisimilarity after determinization in the category of *convex algebras*, algebraic structures that model the closure of convex sets under convex combinations. Stark and Smolka's result is the probabilistic analogue of an earlier paper of Milner [15], whereas Silva and Sokolova's is the probabilistic analogue of an earlier paper of Rabinovich [17], where it is shown that a sound and complete axiomatization of trace semantics of labelled transition systems can be obtained from an axiomatization of bisimilarity. All these works, non-deterministic and probabilistic, restrict themselves to *finite traces*.

To achieve our goal, we use a categorical perspective on the semantics of LMCs. This is in the spirit of [25], but there are crucial technical hurdles to overcome: First, we need to find an endofunctor on a category that models LMCs as coalgebras and allows the derivation of the stream distribution semantics in a canonical way. More specifically, we need to give a coalgebraic characterization of the map that assigns to every state of an LMC the distribution on streams that the state generates. To this end, we carefully craft the endofunctor G on the category CA of convex algebras and convex algebra homomorphisms in Section 5. Second, we show that our endofunctor satisfies a number of desirable properties that enable a sound and complete axiomatization, including the preservation of pullbacks and *properness* [14]. Finally, we need to find a suitable syntax for specifying finite LMCs where stream semantics is of interest. Each of these steps pushes the boundaries of existing work on semantics and decidability of trace equivalence for automata, and they require new technical results that form the core contributions of our paper. We briefly describe our contributions below and give an outline of the paper.

- In Section 2, we recall basic definitions on labelled Markov chains and their semantics.
- In Section 3, we recall the syntax of Stark and Smolka's process algebra [29] and Silva and Sokolova's axioms for finite trace equivalence [23], which will form the basis of our inference system and allow us to state our intended soundness and completeness results.
- In Section 4, we explain our high-level strategy for proving completeness, which follows the *coalgebraic completeness method* described in [22] that originates in [8, 24, 13].

- In Section 5, we define the endofunctor G, which forms the basis of all of our developments. The functor G is defined on the category CA of convex algebras and convex algebra homomorphisms (see Definition 4.1), and makes use of an important *mass-splitting property* that resembles a side condition present in [6]. Crucially, we characterize stream distribution semantics as a final G-coalgebra semantics, via a *determinization* construction that turns LMCs into G-coalgebras. This construction is interesting in its own right, given its simplicity compared to existing finality-based approaches to infinite trace semantics [9, 5, 6].
- In Section 6, we define a G-coalgebra structure on the set of process terms modulo axioms, which endows the terms with an operational semantics. We show that this term coalgebra is universal among the free and finitely generated G-coalgebras by providing unique solutions to finite systems of equations arising from a coalgebra structure.
- In Section 7, we conclude our proof of completeness by establishing that G satisfies a property called *properness*, introduced by Milius in [14]. The proof that G is proper uses a topological characterization of congruences of finitely generated convex algebras due to Sokolova and Woracek [27].
- We conclude with a discussion of related and future work, and the implications of the completeness theorem in Section 8.

Our completeness result is remarkable for two reasons: First and foremost, our axiomatization is precisely the same as Silva and Sokolova's for finite trace semantics. In other words, both the (finite) trace distribution semantics and the stream distribution semantics give rise to the same valid equations between term expressions. Second, the completeness result uses a novel proof of properness [14, 28] that appears to hinge on the topology of bisimulations between coalgebras over convex algebras. The latter is a significant point of departure from the properness proof method of Sokolova and Woracek [28].

2 Labelled Markov Chains and Stream Semantics

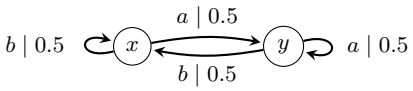
In this section, we briefly recall basic definitions of labelled Markov chains, stream semantics, and the framework of universal coalgebra.

Labelled Markov chains. Given a set X, define $\mathcal{D}(X)$ to be the set of finitely supported probability distributions on X. That is, $\theta \in \mathcal{D}(X)$ if and only if $\theta\colon X \to [0,1]$, $\theta(x) > 0$ for finitely many $x \in X$, and $\theta(X) = \sum_{x\in X} \theta(x) = 1$. Since the support is finite, each $\theta \in \mathcal{D}(X)$ can be written in the form $\sum_{i=1}^{n} r_i \cdot x_i$ such that $r_i \in (0,1]$ and $x_i \in X$ for each $i \leq n$. We write $1 \cdot x$ for the *Dirac delta* at $x \in X$.

For a fixed finite set A of formal symbols called *actions*, a *labelled Markov chain* (or *LMC*) is a pair (X, β) consisting of a set X of *states* and a *transition function* $\beta\colon X \to \mathcal{D}(A \times X)$. An LMC is said to be *finite* if it has finitely many states.

One graphical depiction of a finite LMC is the directed graph with a node for each state and a decorated edge $x \xrightarrow{a|r}_{\beta} y$ between nodes x and y whenever $\beta(x)(a,y) = r$ with $r > 0$. We typically drop the β notation whenever the transition function is clear from context.

▶ **Example 2.1.** The LMC $(X, \beta\colon X \to \mathcal{D}(A \times X))$ with $A = \{a,b\}$, $X = \{x,y\}$, and $\beta(x)(a,y) = \beta(x)(b,x) = \beta(y)(b,x) = \beta(y)(a,y) = 0.5$ is depicted in (2.1).



(2.1)

Stream semantics. A *word* over a finite alphabet A is a finite sequence $a_1 \cdots a_n$ (written as a juxtaposition) of elements of A. We write ε for the *empty* word. A *stream* is an infinite sequence $(a_1, a_2, \dots)$ of elements from A. We write A^* for the set of words and A^ω for the set of streams. The set A^ω carries a topology, with basis given by the *cylinder sets*,

$$B_w = \{(a_1, \dots, a_n, \dots) \mid a_1 \cdots a_n = w\}$$

where $w \in A^*$ is a word. In the notation above, $B_\varepsilon = A^\omega$, as every stream begins with ε.

Recall that a *Borel set* is an element of the σ-algebra generated by the open sets of a topological space, a *Borel measure* is a measure defined on the Borel sets, and a *Borel probability distribution* is a Borel measure with total probability 1 [19].

▶ **Definition 2.2.** *A* stream distribution *is a Borel probability distribution on the space A^ω. The set of all stream distributions on A^ω is written* $\mathsf{Prob}(A^\omega)$.

Each state of an LMC corresponds to a unique stream distribution that records the probability of that state eventually emitting streams in a given Borel set. The following proposition is a special case of [9, Proposition 3.12].

▶ **Proposition 2.3.** *Let (X, β) be an LMC. There is a unique map $[\![-]\!]_\beta : X \to \mathsf{Prob}(A^\omega)$ such that for any $x \in X$ and any $w \in A^*$ and $a \in A$,*

$$[\![x]\!]_\beta (B_{aw}) = \sum_{y \in X} \beta(x)(a, y)\, [\![y]\!]_\beta (B_w)$$

The map $[\![-]\!]_\beta$ above is the *stream semantics* of (X, β). Given states $x, y \in X$, we say x and y are *stream equivalent* if $[\![x]\!]_\beta = [\![y]\!]_\beta$.

LMCs as coalgebras. Universal coalgebra is by now a standard framework for studying state-based systems like LMCs [20]. The theory is sufficiently general for capturing systems where the states come with additional structure. Systems with structured state spaces are central to the main result of this paper, so we state the definitions below for more general categories than the category Set of sets and functions.

▶ **Definition 2.4.** *Given an endofunctor on a category $F \colon \mathsf{C} \to \mathsf{C}$, an* F-coalgebra *is a pair (X, c) consisting of an object X of C and an arrow $c \colon X \to F(X)$. A* coalgebra homomorphism *$h \colon (X, c^X) \to (Y, c^Y)$ is an arrow $h \colon X \to Y$ such that $c^Y \circ h = F(h) \circ c^X$. We write $\mathsf{Coalg}_{\mathsf{C}}(F)$ for the category of F-coalgebras and their homomorphisms.*

The set-mapping $X \mapsto \mathcal{D}(X)$ is a functor, with action on functions given by

$$\mathcal{D}(f)(\theta) = \sum_{i=1}^{n} r_i \cdot f(x_i)$$

where $f \colon X \to Y$ and $\theta = \sum_{i=1}^n r_i \cdot x_i$. The set-mapping $X \mapsto A \times X$ is also a functor, with the action on functions being $f \mapsto \mathrm{id}_A \times f$. By composition, $\mathcal{D}(A \times -)$ is an endofunctor on Set. The point is that LMCs are precisely $\mathcal{D}(A \times -)$-coalgebras. Unravelling the definitions, a coalgebra homomorphism between LMCs $h \colon (X, \beta) \to (Y, \vartheta)$ is a function $h \colon X \to Y$ such that for any $x \in X$, if $\beta(x) = \sum_{i=1}^n r_i \cdot (a_i, x_i)$, then

$$\vartheta(h(x)) = \sum_{i=1}^{n} r_i \cdot (a_i, h(x_i))$$

Coalgebra homomorphisms are precisely the maps that preserve the branching-time behaviour of probabilistic systems.

A category C is *concrete* if there is a faithful functor $U\colon \mathsf{C} \to \mathsf{Set}$. An object X in a concrete category C is essentially a set $U(X)$ with additional structure, and arrows $X \to Y$ are functions that preserve that structure. We write $x \in X$ for $x \in U(X)$. The category Set is of course concrete, as witnessed by the identity functor.

▶ **Definition 2.5.** *Let (X, c^X) and (Y, c^Y) be F-coalgebras where $F\colon \mathsf{C} \to \mathsf{C}$ and C is concrete, $x \in X$ and $y \in Y$. We say x and y are* behaviourally equivalent *and write $x \sim y$ if there is a cospan $(X, c^X) \xrightarrow{h} (Z, c^Z) \xleftarrow{k} (Y, c^Y)$ in $\mathsf{Coalg}_{\mathsf{C}}(F)$ such that $h(x) = k(y)$.*

For LMCs, behavioural equivalence (which coincides with probabilistic bisimilarity) implies stream equivalence [21, Theorem 6.7].

▶ **Proposition 2.6.** *Let (X, β) and (Y, ϑ) be LMCs, $x \in X$, $y \in Y$. If $x \sim y$, then $[\![x]\!]_\beta = [\![y]\!]_\vartheta$.*

The converse fails: for LMCs, behavioural equivalence is strictly finer than stream equivalence (see, e.g., [21, Figure 8]). It follows that there is no LMC structure $(\mathsf{Prob}(A^\omega), c)$ such that $[\![-]\!]_\beta : (X, \beta) \to (\mathsf{Prob}(A^\omega), c)$ is always a coalgebra homomorphism.

3 Axiomatizing Stream Semantics

In this section, we recall Stark and Smolka's specification language for probabilistic transition systems [29] and the axioms for trace equivalence proposed by Silva and Sokolova [25].

A Specification Language for LMCs

Fix an infinite set V of *variables*. Consider the set of *terms* generated by the grammar below,

$$e, f ::= v \mid ae \mid e \oplus_r f \mid \mu v\ e$$

where $v \in V$, $a \in A$, and $r \in [0, 1]$. A variable v is *bound* in a term e if it appears within the scope of $\mu v\ (-)$, and *guarded* if it appears within the scope of some $a(-)$. The set PTerm of *productive process terms* is the set of terms e such that every variable v appearing in e is both guarded and bound. Given variables $v_1, \ldots, v_n$, we write $\mathsf{PTerm}(v_1, \ldots, v_n)$ for the set of guarded terms whose free variables are contained in $\{v_1, \ldots, v_n\}$.

Intuitively, the operation $a(-)$ is *prefixing by a*, and ae denotes the process that makes an a-labelled transition with probability 1 into e. The operations $\oplus_r$ are called *convex sums*, and $e \oplus_r f$ denotes the process whose outgoing transitions are the same as e and f, but with probabilities scaled by $r \in [0, 1]$ and $1 - r$ respectively. The operation $\mu v\ (-)$ is *recursion in v*, and $\mu v\ g$ behaves exactly as $g[\mu v\ g/v]$ does, where $g[\mu v\ g/v]$ denotes the productive process term obtained by substituting every free occurrence of v in g with $\mu v\ g$. Recursion is the source of loops in the LMCs specified by productive process terms. The intuition behind each operation on productive process terms is formalized as follows.

▶ **Definition 3.1.** *For any $e, f \in \mathsf{PTerm}$, $a \in A$, $v \in V$, $g \in \mathsf{PTerm}(v)$, and $r \in [0, 1]$, define*

$$\tau(ae) = 1 \cdot (a, e) \quad \tau(e \oplus_r f) = r\ \tau(e) + (1 - r)\ \tau(f) \quad \tau(\mu v\ g) = \tau(g[\mu v\ g/v])$$

Then (PTerm, τ) is the syntactic LMC.

Each probabilistic process term e shares its stream semantics with a state in a finite LMC. In particular, let $\langle e \rangle$ be the set of probabilistic process terms f such that $e \xrightarrow{a_1|r_1} \cdots \xrightarrow{a_n|r_n} f$. Then $\langle e \rangle$ is finite and τ restricts to a transition structure $\tau_{\langle e \rangle} \colon \langle e \rangle \to \mathcal{D}(A \times \langle e \rangle)$ [21]. We also have $[\![e]\!]_\tau = [\![e]\!]_{\tau_{\langle e \rangle}}$, since $[\![e]\!]_\tau$ only depends on states reachable from e.

The converse is also true. The following theorem, analogous to Kleene's theorem for regular expressions [10], is a direct consequence of results presented in [29].

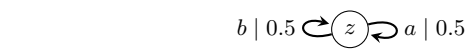
▶ **Theorem 3.2.** *Let (X, β) be a finite LMC and let $x \in X$. There exists an $e \in \mathsf{PTerm}$ such that e and x are behaviourally equivalent.*

As an immediate consequence of Theorem 3.2 and Proposition 2.6, we have that PTerm is a fully expressive specification language for states of finite LMCs.

▶ **Corollary 3.3.** *Let (X, β) be a finite LMC and let $x \in X$. There exists an $e \in \mathsf{PTerm}$ such that $[\![e]\!]_\tau = [\![x]\!]_\beta$.*

From now on, we drop τ and simply write $[\![e]\!]$ instead of $[\![e]\!]_\tau$, for $e \in \mathsf{PTerm}$.

▶ **Example 3.4.** The state x in the LMC (2.1) has the same stream semantics as the term $\mu v\ (bv \oplus_{0.5} a(\mu u\ (au \oplus_{0.5} bv))))$. However, it appears that there is a redundancy in the LMC (2.1). Both x and y emit a and b with the same probability, and each transitions to the other with the same probability. Thus, the stream semantics of both states x and y is the unique Borel probability distribution ρ satisfying $\rho(B_{a_1 \cdots a_n}) = 0.5^n$ for any $a_1 \cdots a_n \in \{a, b\}^*$, making x and y stream equivalent to the state z below. This one-state LMC corresponds to the process term $\mu v\ (av \oplus_{0.5} bv)$.



It follows that $[\![\mu v\ (bv \oplus_{0.5} a(\mu u\ (au \oplus_{0.5} bv))))]\!] = [\![\mu v\ (av \oplus_{0.5} bv)]\!]$.

Axioms for stream equivalence

As we have seen from Example 3.4, even very different looking productive process terms can be stream equivalent. To facilitate reasoning about equivalence, we give a set of inference rules for deducing algebraically that two productive process terms are stream equivalent.

▶ **Definition 3.5** (Provable equivalence)**.** *Probabilistic process terms $e, f \in \mathsf{PTerm}$ are said to be* provably equivalent*, written $e \equiv f$, if $e = f$ can be proven from axioms in Fig. 1. We write $[e]$ for the $\equiv$-equivalence class of e.*

The main goal of the paper is to prove that the axioms in Fig. 1 are sound and complete to reason about stream semantics of LMCs:

$$e \equiv f \iff [\![e]\!] = [\![f]\!] \qquad (\Longleftarrow) : \text{Completeness} \qquad (\Longrightarrow) : \text{Soundness}$$

Soundness was established in [21, Theorem 6.9]. The main result in this paper is completeness, which verifies [21, Conjecture 1].

▶ **Theorem 3.6** (Completeness)**.** *Let $e, f \in \mathsf{PTerm}$. If $[\![e]\!] = [\![f]\!]$, then $e \equiv f$.*

$$e = e \oplus_r e \qquad \mu v\ g = \mu u\ g[u/v] \qquad \frac{e_1 = e_2}{ae_1 = ae_2}$$

$$e_1 \oplus_r e_2 = e_2 \oplus_{1-r} e_1 \qquad \mu v\ g = g[\mu v\ g/v] \qquad \frac{e_1 = f_1 \quad e_2 = f_2}{e_1 \oplus_r e_2 = f_1 \oplus_r f_2}$$

$$(e_1 \oplus_r e_2) \oplus_s e_3 = e_1 \oplus_{rs} (e_2 \oplus_{\frac{s(1-r)}{1-rs}} e_3) \qquad \frac{f = g[f/v]}{f = \mu v\ g} \qquad \frac{e_1 = f_1 \ \ldots \ e_n = f_n}{k[\vec{e}/\vec{v}] = k[\vec{f}/\vec{v}]}$$

$$a(e_1 \oplus_r e_2) = ae_1 \oplus_r ae_2$$

Figure 1 Axioms for probabilistic term equivalence. Above, $e, e_i, f, f_i \in \mathsf{PTerm}$, $\vec{e} = (e_1, \ldots, e_n)$, $\vec{f} = (f_1, \ldots, f_n)$, $g \in \mathsf{PTerm}(v)$, and $k \in \mathsf{PTerm}(v_1, \ldots, v_n)$. We assume that u is not bound in g in the first axiom of the second column. The term $k[\vec{e}/\vec{v}]$ is obtained by simultaneously replacing v_i with e_i for each $i \leq n$. Note that the equivalence relation axioms are implicit. The difference with the axiomatization for bisimilarity is the distributivity axiom (lower-left).

4 Blueprint for Proving Completeness

The main goal of the rest of the paper is to prove Theorem 3.6, completeness of our inference system. We begin with a high-level sketch of the proof to ease the flow into the upcoming technical sections. At the core of our argument will be the fact that the semantics of terms, as given by $[\![-]\!]$, can be *factorized*:

$$\mathsf{PTerm} \xrightarrow{[-]} \mathsf{PTerm}/{\equiv} \xrightarrow{\partial^\dagger} \mathsf{Prob}(A^\omega) \qquad [\![-]\!] : \mathsf{PTerm} \to \mathsf{Prob}(A^\omega) \tag{4.1}$$

The existence of this factorization is a consequence of soundness, which implies that $[\![-]\!]$ factors through the quotient $\mathsf{PTerm}/{\equiv}$ for a particular function $\partial^\dagger \colon \mathsf{PTerm}/{\equiv} \to \mathsf{Prob}(A^\omega)$. Once we have such factorization, we can reason as follows:

$$[\![e]\!] = [\![f]\!] \implies \partial^\dagger([e]) = \partial^\dagger([f]) \overset{\star}{\implies} [e] = [f] \implies e \equiv f$$

Now completeness follows if we can justify the $\star$ step, which amounts to injectivity of $\partial^\dagger$. In other words, Theorem 3.6 follows if $\partial^\dagger$ is injective. And that is precisely what we are going to prove. Before we outline the completeness proof, we need a few notions from convex algebra.

▶ **Definition 4.1.** *A* convex algebra *is an algebraic structure consisting of a set X and a family of binary operations $\oplus_p \colon X \times X \to X$ (written infix) satisfying*

$$x \oplus_1 y = x \quad x \oplus_r x = x \quad x \oplus_r y = y \oplus_{1-r} x \quad (x \oplus_r y) \oplus_s z = x \oplus_{rs} \left(y \oplus_{\frac{s(1-r)}{1-rs}} z\right)$$

An affine map*, or* convex algebra homomorphism*, between convex algebras $(X, \oplus_p^X)$ and $(Y, \oplus_p^Y)$ is a function $h \colon X \to Y$ that satisfies $h(x \oplus_p^X y) = h(x) \oplus_p^Y h(y)$ for each $p \in [0, 1]$. The category of convex algebras and affine maps is denoted* CA*.*

A convex algebra $(X, \oplus_p^X)$ is free *and generated by a set $B \subseteq X$ if every map $f \colon B \to Y$ from B to the carrier of a convex algebra $(Y, \oplus_p^Y)$ extends to a unique affine map $f^\# \colon (X, \oplus_p^X) \to (Y, \oplus_p^Y)$. The set B is then the set of generators of the free algebra $(X, \oplus_p^X)$. If B is a finite set, then the free algebra generated by B is* free finitely generated*,* ffg*, for short. A convex algebra is* finitely generated*,* fg*, for short, if it is a homomorphic image of a free finitely generated one.*

Note that we will often write X instead of $(X, \oplus_p)$ if the convex algebra structure is clear from the context.

Back to the intended completeness result as outlined above, we break the proof of injectivity of $\partial^\dagger$ into 3 steps, each of independent interest.

Step 1

We identify the category of convex algebras as the right base category to define the stream semantics of LMCs. More precisely, we define a functor G on CA and show that the convex algebra of Borel probability distributions $\mathsf{Prob}(A^\omega)$ carries a final G-coalgebra structure $(\mathsf{Prob}(A^\omega), \zeta)$. By turning any LMC (X, β) into a G-coalgebra $(\mathcal{D}(X), \partial_\beta)$ via a *determinization* construction (see Definition 5.11), we obtain the *determinized stream semantics* (X, β), $(\!|-|\!)_\beta = \partial^\dagger_\beta \circ \eta \colon X \to \mathcal{D}(X) \to \mathsf{Prob}(A^\omega)$ via the final coalgebra homomorphism $\partial^\dagger_\beta \colon (\mathcal{D}(X), \partial_\beta) \to (\mathsf{Prob}(A^\omega), \zeta)$. We then relate this determinized stream semantics to the original stream semantics $[\![-]\!]$ defined in Proposition 2.3 using the syntactic LMC (PTerm, τ) as shown in the diagram (4.1).

Step 2

We provide a G-coalgebra structure $(\mathsf{PTerm}/\!\equiv, \partial)$ on the equivalence classes of terms modulo provable equivalence and show that every ffg G-coalgebra (X, β) (i.e., X is ffg) has a unique coalgebra homomorphism into $(\mathsf{PTerm}/\!\equiv, \partial)$. This is related to solving certain systems of equations in $\mathsf{PTerm}/\!\equiv$. We also show that $(\mathsf{PTerm}/\!\equiv, \partial)$ is *locally fg*, in the following sense:

▶ **Definition 4.2.** *A G-coalgebra (X, γ) is* locally fg *if for any $x \in X$, there is a subcoalgebra (U, γ_U) of (X, γ) such that $x \in U$ and U is fg. A locally fg G-coalgebra (X, γ) is* final *if every locally fg G-coalgebra admits a unique coalgebra homomorphism into (X, γ).*

The significance of $(\mathsf{PTerm}/\!\equiv, \partial)$ being locally fg is related to the lemma below.

▶ **Lemma 4.3.** *Every homomorphic image of a locally fg G-coalgebra is also locally fg.*

Consider the surjective-injective factorization of the coalgebra homomorphism $\partial^\dagger$ below.

$$(\mathsf{PTerm}/\!\equiv, \partial) \xrightarrow{\ q\ } (J, \rho) \xrightarrow{\ \iota\ } (\mathsf{Prob}(A^\omega), \zeta), \qquad \partial^\dagger \colon (\mathsf{PTerm}/\!\equiv, \partial) \to (\mathsf{Prob}(A^\omega), \zeta)$$

To show that $\partial^\dagger$ is injective, it suffices to show that the map q has a left inverse, a coalgebra homomorphism $k \colon (J, \rho) \to (\mathsf{PTerm}/\!\equiv, \partial)$ such that $k \circ q = \mathrm{id}$, as then

$$\partial^\dagger([e]) = \partial^\dagger([f]) \Leftrightarrow \iota \circ q([e]) = \iota \circ q([f]) \Rightarrow q([e]) = q([f]) \Rightarrow k \circ q([e]) = k \circ q([f]) \Leftrightarrow [e] = [f].$$

One way to do this is to show that $(\mathsf{PTerm}/\!\equiv, \partial)$ is the *final* locally fg G-coalgebra. In such a case, by Lemma 4.3, (J, ρ) is also locally fg, and therefore admits the desired (necessarily unique) coalgebra homomorphism k. Indeed, by finality, since $k \circ q$ and id are both homomorphisms from $(\mathsf{PTerm}/\!\equiv, \partial)$ to itself, they must be the same, i.e., $k \circ q = \mathrm{id}$.

Step 3

Lastly, we will establish sufficient conditions guaranteeing that $(\mathsf{PTerm}/\!\equiv, \partial)$ is the final locally fg G-coalgebra. Our end goal will be to apply the following theorem, which can be obtained from a combination of [14, Corollary 5.9] and [27, Corollary 5.5].

▶ **Theorem 4.4.** *Suppose that F is a finitary proper endofunctor on* CA *that preserves surjective affine maps. Then an F-coalgebra (Y, ω) is a final locally fg coalgebra if and only if (i) (Y, ω) is locally fg and (ii) for every ffg F-coalgebra $(\mathcal{D}(X), \partial_\beta)$, there is a unique coalgebra homomorphism $(\mathcal{D}(X), \partial_\beta) \to (Y, \omega)$.*

Theorem 4.4 uses the notion of a *proper* functor, which we will define in Definition 7.6 below.

After having completed Step 2, we will have already seen that $(\mathsf{PTerm}/{\equiv}, \partial)$ is locally fg, and furthermore that every ffg G-coalgebra admits a unique coalgebra homomorphism into $(\mathsf{PTerm}/{\equiv}, \partial)$. Thus, completing Step 3 hinges on showing that the functor G is finitary, that it preserves surjective affine maps, and that G is proper. Step 3 is the most technical of the three steps.

To summarize, here are our obligations stated in the three steps above:

1. We must define $G : \mathsf{CA} \to \mathsf{CA}$, endow $\mathsf{Prob}(A^\omega)$ with a G-coalgebra structure ζ, turning $\mathsf{Prob}(A^\omega, \zeta)$ into a final G-coalgebra.
2. Given an LMC (X, β), we must explain how it is determinized to yield a G-coalgebra $(\mathcal{D}(X), \partial_\beta)$, and how its stream semantics $[\![-]\!]$ is obtained from the final coalgebra homomorphism as $[\![-]\!] = \partial^\dagger_\beta \circ \eta$. In other words, we must relate the stream semantics to the determinzed stream semantics $(\!|-|\!)_\beta$.
3. We must define a coalgebra structure $\partial : \mathsf{PTerm}/{\equiv} \to G(\mathsf{PTerm}/{\equiv})$ and show that $(\mathsf{PTerm}/{\equiv}, \partial)$ is locally fg and that free fg G-coalgebras admit unique coalgebra homomorphisms into $(\mathsf{PTerm}/{\equiv}, \partial)$.
4. We must show that G is finitary, preserves surjective algebra homomorphisms, and is proper.

5 Step 1: Convex (Co)Algebras and the Functor G

We begin executing each of the steps in Section 4. We first need some basic definitions on the category CA of convex algebras.

Convex algebras. Recall that a *convex algebra* is an algebraic structure consisting of a set X and a collection of *convex sum operations* $\oplus_r \colon X \times X \to X$ indexed by $r \in [0, 1]$ satisfying the equations in Definition 4.1, and recall that we write CA for the category of convex algebras.

▶ **Example 5.1.** Prime examples of convex algebras are *convex* subsets of $\mathbb{R}^n$, i.e., subsets $C \subseteq \mathbb{R}^n$ such that $\vec{p}, \vec{q} \in C$ implies that $\vec{p} \oplus_r \vec{q} = r\vec{p} + (1-r)\vec{q} \in C$ for all $r \in [0, 1]$. Moreover, for any subset $U \subseteq \mathbb{R}^n$, there is a smallest convex algebra containing U, namely the *convex hull* $\mathrm{conv}(U) = \{r\vec{p} + (1-r)\vec{q} \mid \vec{p}, \vec{q} \in U \text{ and } r \in [0, 1]\}$.

We may use the following syntax as a generalized convex sum in an arbitrary convex algebra: given $r_1, \ldots, r_n \in (0, 1)$ and $x_1, \ldots, x_n$, define

$$\bigoplus_{i=1}^{n} r_i \cdot x_i = x_n \oplus_{r_n} \left(\bigoplus_{i=1}^{n-1} \frac{r_i}{1 - r_n} \cdot x_i \right) \tag{5.1}$$

It is important to note that, technically, the base case is $n = 2$. We can also use this notation if $r_i = 0$ for $i \neq j$ and $r_j = 1$, but in that case we define $\bigoplus_{i=1}^{n} r_i \cdot x_i = x_j$. Up to the convex algebra axioms, any two ways of reordering the summands of (5.1) produces equivalent terms. This justifies the slight abuse of notation $\bigoplus_{x \in S} r_x \cdot x$, where S is a set and $r_{(-)} \colon S \to [0, 1]$ is a function such that $\sum_{x \in S} r_x = 1$ and only finitely many of the r_x are non-zero.

Free convex algebras. $(\mathcal{D}(X), \oplus_p)$ is the *free* convex algebra generated by the set X. Hence, for any convex algebra $(Y, \oplus_p^Y)$, and any function $f\colon X \to Y$, there is a unique *linear extension* $f^\#\colon (\mathcal{D}(X), \oplus_p) \to (Y, \oplus_p^Y)$ of f such that $f^\#(1 \cdot x) = f(x)$. The universal property of free convex algebras gives rise to the adjunction $\mathcal{F} \dashv \mathcal{U}$, where $\mathcal{F}(X) = (\mathcal{D}(X), \oplus_p)$ is the *free* functor that maps a set to the free convex algebra generated by it and a function $f\colon X \to Y$ to $\mathcal{D}(f)\colon \mathcal{D}(X) \to \mathcal{D}(Y)$, and $\mathcal{U}$ is the *forgetful* functor from CA to Set that forgets the algebraic structure and is identity on homomorphisms.

The free functor $\mathcal{F}$ is a left adjoint to the forgetful functor, and clearly $\mathcal{D} = \mathcal{U} \circ \mathcal{F}$. It follows that $(\mathcal{D}, \eta, \mu)$ is a monad on Set with $\eta_X(x) = 1 \cdot x$ and $\mu_X = (\mathrm{id}_{\mathcal{D}(X)})^\#$, and furthermore, CA is isomorphic to the category of Eilenberg-Moore algebras for $\mathcal{D}$ [31]. In particular, the free convex algebra generated by a set X is the Eilenberg-Moore algebra $(\mathcal{D}(X), \mu_X)$. We often omit writing the forgetful functor when no confusion arises, and (in accordance with our convention to drop the algebra structure when no confusion arises) also often just write $\mathcal{D}(X)$ for the free algebra $\mathcal{F}(X)$.

Adding a fresh element $\bot$ to a convex algebra. In order to define the endofunctor G, we need the following construction on convex algebras. Given a convex algebra X, define $X_\bot = \{\bot\} \cup \{r \cdot x \mid r \in (0,1], x \in X\}$. The set $X_\bot$ obtains a convex algebra structure with respect to the convex sum operation defined

$$\bot \oplus_q \bot = \bot \qquad r \cdot x \oplus_q \bot = (qr) \cdot x \qquad \bot \oplus_q s \cdot y = ((1-q)s) \cdot y$$
$$r \cdot x \oplus_q s \cdot y = (qr + (1-q)s) \cdot (x \oplus_{\frac{qr}{qr+(1-q)s}} y)$$

▶ **Lemma 5.2.** *Let X be a convex algebra. As defined above, $(X_\bot, \oplus)$ is a convex algebra. Moreover, given $r \cdot x$ and $s \cdot y$ in $X_\bot$, $r \cdot x = s \cdot y$ if and only if $r = s$ and $x = y$.*

▶ Remark 5.3. We introduce some notation going forwards. We often use the notation $0 \cdot x$ for $\bot$, even implicitly, despite that $0 \cdot x = 0 \cdot y$ for all $x, y \in X$.

The construction $(-)_\bot\colon \mathsf{CA} \to \mathsf{CA}$ is a functor whose action on convex algebra homomorphisms is given by $h_\bot(r \cdot x) = r \cdot h(x)$ for any convex algebra homomorphism $h\colon (X, \oplus_p) \to (Y, \oplus_p)$ and any $x \in X$. The homomorphism $h_\bot$ additionally satisfies $h_\bot(\bot) = \bot$. Freely adjoining $\bot$ is analogous to going from probability distributions to sub-probability distributions (maps $\theta\colon X \to [0,1]$ such that $\sum_{x \in X} \theta(x) \leq 1$). The following lemma makes this precise.

▶ **Lemma 5.4.** *Let $\mathcal{D}_\bot$ be the finitely supported sub-probability distribution functor, and let $\mathsf{Prob}_\bot(A^\omega)$ be the set of Borel sub-probability measures on A^ω. Then as convex algebras, $\mathcal{D}(X)_\bot \cong \mathcal{D}_\bot(X)$ and $\mathsf{Prob}(A^\omega)_\bot \cong \mathsf{Prob}_\bot(A^\omega)$.*

The functor $G : \mathsf{CA} \to \mathsf{CA}$

We are now ready to introduce the functor on CA needed to move from Set to CA. There are different ways to define such a functor, e.g. Silva and Sokolova [25] use another functor for the axiomatization of finite trace semantics. The choice of the "right" functor so that our intended results go through, i.e., the choice of this particular functor G, is one of the main contributions of this paper.

Given a convex algebra X and a convex algebra homomorphism $h\colon X \to Y$, let

$$G(X) = \left\{ f : A \to X_\bot \mid \sum_{a \in A} r_a^f = 1 \right\} \qquad G(h)(f)(a) = r_a^f \cdot h(x_a^f) \tag{5.2}$$

where $f(a) = r_a^f \cdot x_a^f$ for each $f \in G(X)$ and $a \in A$. Equivalently, $G(h)(f) = h_\perp \circ f$. Note that in the definition of $G(X)$ above, the sum is the usual sum of real numbers, and that we define $r_a^f = 0$ and leave x_a^f undefined when $f(a) = \perp$.

▶ **Proposition 5.5.** *As it is defined in* (5.2)*, G is an endofunctor on* CA.

We use the following terminology to refer to the defining property of G: If $f : A \to X_\perp$ has the property that $\sum_a r_a^f = 1$, as mentioned in (5.2), we say that f satisfies the *mass-splitting property*, or that f is *mass splitting*.[1]

In particular, a function $f\colon A \to \mathcal{D}_\perp(X)$ is mass splitting, i.e., $f \in G(\mathcal{D}(X))$, if and only if the total mass $\sum_{a\in A}\sum_{x\in X} f(a)(x)$ is equal to 1. Given such a function, one can reverse-engineer a unique probability distribution $\theta \in \mathcal{D}(A \times X)$ such that f computes the marginal $f(a) = \theta(\{a\} \times X)$ for each $a \in A$. Thus, a G-coalgebra of the form $(\mathcal{D}(X), \gamma)$ represents the same data as an LMC (X, β) by reverse-engineering $\beta(x)$ from $\gamma(1 \cdot x)$ for each $x \in X$. We think of G-coalgebras as the *deterministic* counterpart of LMCs. Their exact relationship will be made precise at the end of this section.

▶ Remark 5.6. Note that as a set, $X_\perp \cong 1 + (0, 1] \times X$, and so the description of $G(X)$ above can also be taken as a definition of a functor $H\colon \mathsf{Set} \to \mathsf{Set}$. Indeed, G is a lifting of H to CA. However, the convex algebra structure on $X_\perp$ is not the convex algebra structure on $1 + (0, 1] \times X$ obtained from (co)products in CA. The convex algebra structure is instead hand-tailored to match the structure of sub-probability distributions.

In a given G-coalgebra (X, γ), we write $\mathsf{mass}_\gamma(a, x)$ for $r_a^{\gamma(x)}$, and whenever $r_a^{\gamma(x)} > 0$, we write $\mathsf{next}_\gamma(a, x)$ for $x_a^{\gamma(x)}$. Then whenever $\gamma(x)(a) = \perp$, $\mathsf{mass}_\gamma(a, x) = 0$ while $\mathsf{next}_\gamma(a, x)$ is undefined; and when $\mathsf{mass}_\gamma(a, x) > 0$,

$$\gamma(x)(a) = \mathsf{mass}_\gamma(a, x) \cdot \mathsf{next}_\gamma(a, x). \tag{5.3}$$

where the $\cdot$ symbol here is from $X_\perp$. Note that we often drop γ and write simply mass and next. In this notation, the mass-splitting property says that for all $x \in X$, we have $\sum_{a\in A} \mathsf{mass}(a, x) = 1$.

Given G-coalgebras (X, γ) and (Y, ω), unravelling the definitions of mass and next reveals that a function $h : X \to Y$ is a coalgebra homomorphism if and only if

$$\mathsf{mass}(a, x) \cdot h(\mathsf{next}(a, x)) = \mathsf{mass}(a, h(x)) \cdot \mathsf{next}(a, h(x)) \tag{5.4}$$

for any $a \in A$ and $x \in X$. In other words, for all $x \in X$ and $a \in A$, $\mathsf{mass}(a, x) = \mathsf{mass}(a, h(x))$, and if this is greater than 0, then $h(\mathsf{next}(a, x)) = \mathsf{next}(a, h(x))$ as well.

A final G-coalgebra. We are now in the position to show that $\mathsf{Prob}(A^\omega)$ is the carrier of a final G-coalgebra. First, observe that, like $\mathcal{D}(X)$, $\mathsf{Prob}(A^\omega)$ is a convex algebra with the canonical convex sums, $\rho \oplus_r \theta = r\rho + (1 - r)\theta$. In the proof of Theorem 5.13, we use the $\mathcal{D}$-algebra in the more general, Eilenberg-Moore, form $(\mathsf{Prob}(A^\omega), \Sigma)$, where

$$\Sigma(\sum_{i=1}^{n} r_i \cdot \rho_i)(B) = \sum_{i=1}^{n} r_i \rho_i(B) \tag{5.5}$$

[1] The mass-splitting property was inspired by a condition in Goy and Rot's paper [6, Proposition 4.5].

▶ **Definition 5.7.** *The G-coalgebra structure* $(\mathsf{Prob}(A^\omega), \zeta)$ *is given by, for* $\rho \in \mathsf{Prob}(A^\omega)$,

$$\zeta(\rho)(a) = \begin{cases} \bot & \text{if } \rho(B_a) = 0 \\ \rho(B_a) \cdot \big(B \mapsto \rho(aB)/\rho(B_a)\big) & \text{if } \rho(B_a) > 0 \end{cases} \tag{5.6}$$

where for Borel B, $aB = \{(a, a_1, \dots) \mid (a_1, \dots) \in B\}$ *is the Borel set obtained by prefixing.*

It is easy to check that ζ is a convex algebra homomorphism and that $\zeta(\rho)$ satisfies the mass-splitting property for each $\rho \in \mathsf{Prob}(A^\omega)$.

▶ Remark 5.8. It is important to note that $\mathsf{next}_\zeta(a, -) \colon \mathsf{Prob}(A^\omega) \to \mathsf{Prob}(A^\omega)$ is *not* (in general) a convex algebra homomorphism.

▶ **Theorem 5.9.** *The G-coalgebra* $(\mathsf{Prob}(A^\omega), \zeta)$ *is final. That is, for any G-coalgebra* (X, γ)*, there is a unique coalgebra homomorphism* $\gamma^\dagger \colon (X, \gamma) \to (\mathsf{Prob}(A^\omega), \zeta)$.

Here is a hint of a hint. We define $\gamma^\dagger(x)(B_w) \in [0, 1]$ by recursion on the length of w:

$$\begin{aligned} \gamma^\dagger(x)(B_\varepsilon) &= 1 \\ \gamma^\dagger(x)(B_{aw}) &= \begin{cases} 0 & \text{if } \gamma(x)(a) = \bot \\ \mathsf{mass}(a, x) \cdot \gamma^\dagger(\mathsf{next}(a, x))(B_w) & \text{if } \gamma(x)(a) \neq \bot \end{cases} \end{aligned} \tag{5.7}$$

One needs to show that this specifies each function $\gamma^\dagger$ as a finitely additive function on the generators of the Borel algebra, that the resulting function $\gamma^\dagger$ is a convex algebra morphism as well as a G-coalgebra morphism, and finally that it is the unique such map.

▶ Remark 5.10. It is also true that (forgetting the convex algebra structure) $\mathsf{Prob}(A^\omega)$ is the final coalgebra of the functor $H : \mathsf{Set} \to \mathsf{Set}$ mentioned in Remark 5.6. This provides a way to define the stream semantics of LMCs using finality (Proposition 2.3), i.e., without the convex algebra structure. However, other ingredients in our completeness proof do require convex algebras.

Determinization: Connecting LMCs and G-coalgebras

Earlier in this section, we mentioned that one can think of G-coalgebras as *deterministic* counterparts to LMCs. We now make the relationship between LMCs and G-coalgebras precise. Using the universal property of free convex algebras and the correspondence between finitely supported probability distributions $\theta \in \mathcal{D}(A \times -)$ and functions $f \colon A \to \mathcal{D}_\bot(-)$ satisfying the mass-splitting property, we can construct a *determinization functor* $\Delta \colon \mathsf{Coalg}_{\mathsf{Set}}(\mathcal{D}(A \times -)) \to \mathsf{Coalg}_{\mathsf{CA}}(G)$ as follows.

First, we define the natural transformation $\lambda_Y : \mathcal{D}(A \times Y) \to G(\mathcal{D}(Y))$ by

$$\lambda_Y(\theta)(a) = \begin{cases} \bot & \text{if } s_a = 0 \\ s_a \cdot (\frac{1}{s_a}\theta(a, -)) & \text{otherwise} \end{cases} \tag{5.8}$$

for each set Y, $\theta \in \mathcal{D}(A \times Y)$, and $a \in A$, with $s_a = \sum_{y \in Y} \theta(a, y)$. After making the identification $\mathcal{D}(X)_\bot = \mathcal{D}_\bot(X)$, this amounts to $\lambda_Y(\theta)(a)(x) = \theta(a, x)$. A routine check verifies that λ_Y is natural in Y and that for any $\theta \in \mathcal{D}(A \times Y)$, $\lambda_Y(\theta)$ satisfies the mass-splitting property.

Having constructed λ, we can now define the determinization $\Delta(Y, \beta)$ of the LMC (Y, β) to be the linear extension of the composition of λ_Y after β.

▶ **Definition 5.11.** *The* determinization functor $\Delta\colon \mathsf{Coalg}_{\mathsf{Set}}(\mathcal{D}(A\times -)) \to \mathsf{Coalg}_{\mathsf{CA}}(G)$ *is the functor given by* $\Delta(Y,\beta) = ((\mathcal{D}(Y),\mu_Y),\partial_\beta)$ *with* $\partial_\beta = (\lambda_Y\circ\beta)^\#$ *for any LMC* (Y,β)*, and* $\Delta(h) = \mathcal{D}(h)$ *for any coalgebra homomorphism* h *between LMCs.*

Moreover, we can show that λ is a natural isomorphism, by providing an inverse transformation $\chi_Y : G(\mathcal{D}(Y)) \to \mathcal{D}(A\times Y)$. For $h \in G(\mathcal{D}(Y))$ with $h(a) = r_a\cdot h_a$, define

$$\chi_Y(h)(a,y) = \begin{cases} 0 & h(a) = \bot \\ r_a h_a(y) & \text{otherwise} \end{cases} \tag{5.9}$$

▶ **Proposition 5.12.** *The natural transformations* λ *and* χ *are inverse to each other. Moreover, given a* G*-coalgebra* $((\mathcal{D}(Y),\mu_Y),\gamma)$*, let* $\beta : Y \to \mathcal{D}(A\times Y)$ *be given by* $\beta = \chi_Y\circ\gamma\circ\eta_Y$*. Then* $((\mathcal{D}(Y),\mu_Y),\gamma) = \Delta(Y,\beta)$*. As a result, a* G*-coalgebra is ffg iff it is a determinized finite LMC.*

By Theorem 5.9, $(\mathsf{Prob}(A^\omega),\zeta)$ is a final G-coalgebra, so from any LMC (Y,β), we may determinize to get a G-coalgebra $\Delta(Y,\beta)$ and then use finality to obtain a unique coalgebra homomorphism $\partial_\beta^\dagger\colon \Delta(Y,\beta) \to ((\mathsf{Prob}(A^\omega),\Sigma),\zeta)$. This yields a *determinized stream semantics* map $(\!|-|\!)_\beta : Y \to \mathsf{Prob}(A^\omega)$ by composition, i.e., $(\!|\, y\,|\!)_\beta = \partial_\beta^\dagger(1\cdot y)$. Fulfilling its intended purpose, determinized stream semantics does indeed coincide with stream semantics as we previously defined it.

▶ **Theorem 5.13.** *For every LMC* (X,β)*,* $(\!|-|\!)_\beta = [\![-]\!]_\beta$ *.*

Proof. Let $\alpha : \mathcal{D}(A\times\mathsf{Prob}(A^\omega)) \to \mathsf{Prob}(A^\omega)$ be given by $\alpha(\theta)(B_\varepsilon) = 1$, and for all $a\in A$, $w\in A^*$,

$$\alpha(\theta)(B_{aw}) = \sum_{\rho\in\mathsf{Prob}(A^\omega)} \theta(a,\rho)\,\rho(B_w) \tag{5.10}$$

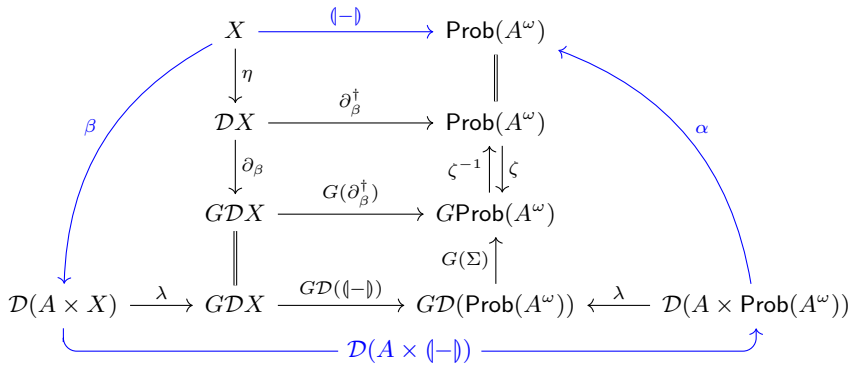
For a fixed $\theta\in\mathcal{D}(A\times\mathsf{Prob}(A^\omega))$, let us use the notation s_a for $\sum_{\rho\in\mathsf{Prob}(A^\omega)}\theta(a,\rho)$. Note that taking w in (5.10) to be the empty word ε gives $s_a = \alpha(\theta)(B_a)$.

Fix (X,β). Let us first check that a map $f : X \to \mathsf{Prob}(A^\omega)$ satisfies the equation mentioned in Proposition 2.3 if and only if $f = \alpha\circ\mathcal{D}(A\times f)\circ\beta$. That is, $f(x)(B_{aw}) = \sum_{y\in X}(\beta(x)(a,y))(f(y)(B_w))$ for all $a\in A$ and $w\in A^*$ if and only if $f = \alpha\circ\mathcal{D}(A\times f)\circ\beta$. This follows from:

$$\begin{aligned}(\alpha\circ\mathcal{D}(A\times f)\circ\beta)(x)(B_{aw}) &= \sum_{\rho\in\mathsf{Prob}(A^\omega)} (\mathcal{D}(A\times f)(\beta(x))(a,\rho))\rho(B_w)\\ &= \sum_{\rho\in\mathsf{Prob}(A^\omega)} \left(\sum_{y:f(y)=\rho}\beta(x)(a,y)\right)\rho(B_w)\\ &= \sum_{y\in X}(\beta(x)(a,y))(f(y)(B_w))\end{aligned}$$

where the first equality is by the definition of α, the second equality is the definition of $\mathcal{D}(A\times f)$, and the third only rearranges the sum.

In the notation of Proposition 2.3, the map $[\![-]\!] = [\![-]\!]_\beta$ is the unique map so that $[\![-]\!] = \alpha\circ\mathcal{D}(A\times[\![-]\!])\circ\beta$. So we shall show that the $(\!|-|\!)$ has this same property. We thus show the commutativity of the outer diagram below (with arrows in blue):



The top square commutes by definition of $(\!|-|\!)$, the left part commutes as $\partial_\beta \circ \eta = \lambda \circ \beta$ by definition of ∂_β, the middle square commutes because $\partial^\dagger_\beta$ is a coalgebra homomorphism, and the part on the bottom commutes by naturality of λ. The commutativity of the remaining two parts is shown below.

We first prove that $\zeta \circ \alpha = G(\Sigma) \circ \lambda$, giving commutativity of the part on the right. For $\theta \in \mathcal{D}(A \times \mathsf{Prob}(A^\omega))$, $a \in A$, and $w \in A^*$, we have, on the one hand:

$$\alpha(\theta)(B_{aw}) = \sum_{\rho \in \mathsf{Prob}(A^\omega)} \theta(a,\rho)\rho(B_w)$$

$$\zeta(\alpha(\theta))(a) = \begin{cases} \bot & \text{if } s_a = 0 \\ s_a \cdot (B_w \mapsto \sum_{\rho \in \mathsf{Prob}(A^\omega)} \frac{\theta(a,\rho)}{s_a}\rho(B_w)) & \text{if } s_a \neq 0 \end{cases} \tag{5.11}$$

We have used definitions of α from (5.10) and ζ from (5.6), that $aB_w = B_{aw}$ and that $\alpha(\theta)(B_a) = s_a$. On the other hand, we use the definitions of λ from (5.8) and Σ from (5.5):

$$\lambda_{\mathsf{Prob}(A^\omega)}(\theta)(a) = \begin{cases} \bot & \text{if } s_a = 0 \\ s_a \cdot (\rho \mapsto \frac{\theta(a,\rho)}{s_a}) & \text{if } s_a \neq 0 \end{cases}$$

$$G(\Sigma)(\lambda_{\mathsf{Prob}(A^\omega)}(\theta))(a) = \begin{cases} \bot & \text{if } s_a = 0 \\ s_a \cdot (B_w \mapsto \sum_{\rho \in \mathsf{Prob}(A^\omega)} \frac{\theta(a,\rho)}{s_a}\rho(B_w)) & \text{if } s_a \neq 0 \end{cases} \tag{5.12}$$

Equations (5.11) and (5.12) now give $\zeta \circ \alpha = G(\Sigma) \circ \lambda$.

We turn to the commutativity of the remaining square. First, affineness of the map $\partial^\dagger_\beta : \mathcal{D}X \to \mathsf{Prob}(A^\omega)$ yields $\partial^\dagger_\beta \circ \mu_X = \Sigma \circ \mathcal{D}(\partial^\dagger_\beta)$. We precompose with $\mathcal{D}(\eta_X)$, and use the monad law $\mu_X \circ \mathcal{D}(\eta_X) = \mathrm{id}_{\mathcal{D}X}$ along with the definition of $(\!|-|\!)$. Thus $\partial^\dagger_\beta = \Sigma \circ \mathcal{D}((\!|-|\!))$. Now apply G to see the desired commutativity. ◀

Returning to our blueprint for completeness in Section 4, Theorem 5.13 shows that $[\![-]\!]$ arises from the final coalgebra map of (PTerm, τ).

6 Step 2: PTerm/$\equiv$ as a G-coalgebra

The set $\mathsf{PTerm}/\equiv$ of provable equivalence classes of productive process terms inherits a canonical convex algebra structure from PTerm, given by $[e] \oplus_r [f] = [e \oplus_r f]$. These operations are well-defined because Fig. 1 includes the necessary axiom and they are indeed convex operations as Fig. 1 includes the convex algebra axioms. In this section, we show that $\mathsf{PTerm}/\equiv$ also carries a canonical G-coalgebra structure $(\mathsf{PTerm}/\equiv, \partial)$. We then focus on two goals: The first goal is to show that the stream semantics of a productive process

term e is equal to the stream distribution $\partial^\dagger([e])$ obtained from the finality of $(\mathsf{Prob}(A^\omega), \zeta)$. The second goal of this section is to show that $(\mathsf{PTerm}/{\equiv}, \partial)$ is locally fg and that every ffg G-coalgebra admits a unique coalgebra homomorphism into $(\mathsf{PTerm}/{\equiv}, \partial)$.

Defining ∂. Let $\tau(e) = \sum_{i=1}^n r_i \cdot (a_i, e_i)$ and write $s_a = \sum_{a_i = a} r_i$. We define the map $\partial\colon \mathsf{PTerm}/{\equiv} \to G(\mathsf{PTerm}/{\equiv})$ using the formulas

$$\mathsf{mass}_\partial(a, [e]) = \sum_{a_i = a} r_i \qquad \mathsf{next}_\partial(a, [e]) = \Big[\bigoplus_{i=1}^n (r_i/s_a) \cdot e_i\Big] \tag{6.1}$$

for any $\equiv$-equivalence class $[e] \in \mathsf{PTerm}/{\equiv}$ and $a \in A$. It can be shown by induction on derivations that (6.1) describes a well-defined map, i.e., $e \equiv f$ implies the right-hand sides of the equations in (6.1) agree.

The following characterization of $(\mathsf{PTerm}/{\equiv}, \partial)$ illustrates that this is a natural choice of G-coalgebra structure on $\mathsf{PTerm}/{\equiv}$.

▶ **Lemma 6.1.** *Given $e_1, e_2 \in \mathsf{PTerm}$, let*

$$\tau(e_1) = \sum_{i=1}^n r_i \cdot (a_i, f_i) \qquad \tau(e_2) = \sum_{i=1}^n s_i \cdot (a_i, f_i)$$

If $e_1 \equiv e_2$, then for any $a \in A$,

$$r_a = s_a \qquad \text{and} \qquad \bigoplus_{i=1}^n (r_i/r_a) \cdot f_i \equiv \bigoplus_{i=1}^n (s_i/s_a) \cdot f_i \tag{6.2}$$

where $r_a = \sum_{a_i = a} r_i$ and $s_a = \sum_{a_i = a} s_i$.

The proof of Lemma 6.1 is a rather long induction on the proof of $e \equiv f$. As an immediate consequence of this lemma, we obtain the following.

▶ **Lemma 6.2.** *Let $(\mathcal{D}(\mathsf{PTerm}), \partial_\tau) = \Delta(\mathsf{PTerm}, \tau)$ and $h_\Sigma = ([-])^\#$ be the linear extension of the quotient-by-$\equiv$ map. Then the following diagram commutes.*

$$\begin{array}{ccc} \mathcal{D}(\mathsf{PTerm}) & \xrightarrow{h_\Sigma} & \mathsf{PTerm}/{\equiv} \\ \downarrow{\scriptstyle \partial_\tau} & & \downarrow{\scriptstyle \partial} \\ G(\mathcal{D}(\mathsf{PTerm})) & \xrightarrow{G(h_\Sigma)} & G(\mathsf{PTerm}/{\equiv}) \end{array} \tag{6.3}$$

In particular, ∂ is a convex algebra homomorphism, and $(\mathsf{PTerm}/{\equiv}, \partial)$ is a homomorphic image of the determinized syntactic LMC.

▶ **Theorem 6.3.** *For any $e \in \mathsf{PTerm}$, $[\![e]\!] = \partial^\dagger([e])$.*

Proof. By Theorem 5.9 and Theorem 5.13, $[\![e]\!] = \partial_\tau^\dagger(1 \cdot e) = \partial^\dagger \circ h_\Sigma(1 \cdot e) = \partial^\dagger([e])$. ◀

▶ **Theorem 6.4.** *The G-coalgebra $(\mathsf{PTerm}/{\equiv}, \partial)$ is locally fg.*

Proof. It follows from results due to Stark and Smolka [29] that the syntactic LMC (PTerm, τ) is locally finite, in the sense that for any $e \in \mathsf{PTerm}$, there is a finite subcoalgebra (U, τ_U) of (PTerm, τ) containing e. So, let $[e] \in \mathsf{PTerm}/{\equiv}$ and find a finite subcoalgebra (U, τ_U) of (PTerm, τ) containing e. Then $\Delta(U, \tau_U)$ is a free fg subcoalgebra of $\Delta(\mathsf{PTerm}, \tau) = (\mathcal{D}(\mathsf{PTerm}), \partial_\tau)$ containing $1 \cdot e$. Taking the image of $\Delta(U, \tau_U)$ under h_Σ, we obtain a finite subcoalgebra $(V, \partial_V) = h_\Sigma(\Delta(U, \tau_U))$ of $(\mathsf{PTerm}/{\equiv}, \partial)$ containing $[e] = h_\Sigma(1 \cdot e)$, as a quotient of a free fg G-coalgebra. Thus, $[e]$ is contained in a fg subcoalgebra. ◀

Systems of equations from G-coalgebras and their unique solutions

The next goal is to show that every ffg G-coalgebra admits a unique coalgebra homomorphism into $(\mathsf{PTerm}/{\equiv}, \partial)$. As we remarked after Definition 5.11, every ffg G-coalgebra is of the form $\Delta(X, \beta)$ for some finite LMC (X, β). So, it suffices to show that every determinized finite LMC admits a unique coalgebra homomorphism into $\mathsf{PTerm}/{\equiv}$. As we will see, each coalgebra homomorphism $\Delta(X, \beta) \to (\mathsf{PTerm}/{\equiv}, \partial)$ corresponds to a solution to a particular system of equations.

▶ **Definition 6.5.** *The* guarded system of equations *corresponding to the finite LMC (X, β) is the set of formal equations*

$$\mathcal{S}(X, \beta) = \left\{ x = \bigoplus_{(a,y) \in A \times X} \beta(x)(a, y) \cdot ay \;\middle|\; x \in X \right\} \tag{6.4}$$

A solution *to the guarded system of equations (6.4) is a map*

$$\varphi \colon X \to \mathsf{PTerm} \qquad \text{such that} \qquad (\forall x \in X)\ \varphi(x) \equiv \bigoplus_{(a,y) \in A \times X} \beta(x)(a, y) \cdot a\varphi(y)$$

Two solutions φ, ψ are equivalent*, written $\varphi \equiv \psi$, if $\varphi(x) \equiv \psi(x)$ for all $x \in X$.*

The following theorem was a key component of Stark and Smolka's completeness proof for bisimilarity.

▶ **Theorem 6.6** (Stark-Smolka [29])**.** *Every guarded finite system of equations has a unique solution up to $\equiv$ without the use of the distributivity axiom $a(e \oplus_r f) = ae \oplus_r af$.*

An immediate consequence of the above theorem is the existence and uniqueness of solutions for systems of equations that arise from LMCs.

▶ **Corollary 6.7.** *Let (X, β) be a finite LMC. Then $\mathcal{S}(X, \beta)$ has a unique solution up to $\equiv$.*

Using the distributivity axiom, we can transform each equation in (6.4) into an equivalent system of equations of the form

$$x = \bigoplus_{a \in A} \mathsf{mass}(a, x) \cdot a\ \mathsf{next}(a, x)$$

where mass and next are derived from ∂_β. This tells us that a map $\varphi \colon X \to \mathsf{PTerm}$ is a solution to $\mathcal{S}(X, \beta)$ if and only if for all $x \in X$,

$$\varphi(x) \equiv \bigoplus_{a \in A} \mathsf{mass}(a, x) \cdot a\ \varphi(\mathsf{next}(a, x))$$

Solving systems of equations of this form is equivalent to finding G-coalgebra homomorphisms into $(\mathsf{PTerm}/{\equiv}, \partial)$.

▶ **Lemma 6.8.** *Let (X, β) be a finite LMC, and let $\varphi \colon X \to \mathsf{PTerm}$. Define $s_\beta \colon \mathcal{D}(X) \to \mathsf{PTerm}/{\equiv}$ to be the linear extension of the composition $[-] \circ \varphi \colon X \to \mathsf{PTerm}/{\equiv}$. Then φ is a solution to $\mathcal{S}(X, \beta)$ if and only if $s \colon \Delta(X, \beta) \to (\mathsf{PTerm}/{\equiv}, \partial)$ is a coalgebra homomorphism.*

We immediately obtain the following theorem.

▶ **Theorem 6.9.** *Let (X, β) be a finite LMC. There is a unique G-coalgebra homomorphism $s_\beta \colon \Delta(X, \beta) \to (\mathsf{PTerm}/{\equiv}, \partial)$.*

Hence, recalling that every ffg coalgebra arises via determinisation (see Proposition 5.12) yields that we have a unique homomorphism from any ffg coalgebra to $(\mathsf{PTerm}/{\equiv}, \partial)$.

7 Step 3: Properness of G

In this section, we finish the outline of completeness that we stated in Section 4 by establishing that G is finitary, preserves surjective affine maps, and is proper. By Theorems 4.4, 6.4, and 6.9, this allows us to conclude that $(\mathsf{PTerm}/{\equiv}, \partial)$ is the final locally fg G-coalgebra.

▶ **Lemma 7.1.** *G preserves pullbacks, and hence monomorphisms.*

Let us mention that monomorphisms in CA are exactly those affine maps which are injective as set functions. This follows from the fact that $U\colon \mathsf{CA} \to \mathsf{Set}$ is a right adjoint and thus preserves all limits, in particular all pullbacks. Recall that in any category monos are characterized as special pullbacks as in the square below. In particular, let $f : X \to Y$ be a monomorphism in CA. Then the square below is a pullback (and conversely) in CA.

$$\begin{array}{ccc} X & \xrightarrow{\mathrm{id}} & X \\ {\scriptstyle \mathrm{id}}\downarrow & & \downarrow{\scriptstyle f} \\ X & \xrightarrow{f} & Y \end{array}$$

Then its image under U is also a pullback and thus $U(f)$ is a monomorphism in Set: that is, f is an injective function.

For space reasons, we omit the proof that G preserves pullbacks. Using Lemma 7.1, we can establish the first required property of G.

▶ **Lemma 7.2.** *The functor $G : \mathsf{CA} \to \mathsf{CA}$ on CA is finitary.*

Proof. We are going to use the following results:

Fact 1. The forgetful functor $U\colon \mathsf{CA} \to \mathsf{Set}$ creates directed colimits.

Fact 2. Let C be a category equipped with a functor $U\colon \mathsf{C} \to \mathsf{S}$ that creates – hence, preserves and reflects – directed colimits. Let $G\colon \mathsf{C} \to \mathsf{C}$ be a lifting of an endofunctor $H\colon \mathsf{S} \to \mathsf{S}$, i.e., $U \circ G = H \circ U$. Then, if H preserves directed colimits, so does G.

In our situation, G is defined in Eq. (5.2), $\mathsf{C} = \mathsf{CA}$, and $\mathsf{S} = \mathsf{Set}$. The proof of Fact 1 is routine, and similar to that of [1, Remark 3.4 (vii).(4)].

Let us briefly establish Fact 2. Let $D\colon (I, \leq) \to \mathsf{C}$ be a directed diagram in C, and let $(d_i\colon Di \to Y)_{i\in I}$ be a colimiting cocone for D. We want to show that $(G(d_i)\colon GDi \to GY)_{i\in I}$ is a colimiting cocone for $G \circ D$. Since U reflects colimits, it suffices to show that $(UG(d_i)\colon UGDi \to UGY)_{i\in I}$ is a colimiting cocone for $U \circ G \circ D$. To this end, consider the directed diagram $U \circ D\colon (I, \leq) \to \mathsf{S}$. Since U preserves directed colimits, $(U(d_i)\colon UDi \to UY)_{i\in I}$ is a colimiting cocone for $U \circ D$. Now, since H is finitary, i.e., it preserves directed colimits, $(HU(d_i)\colon HUDi \to HUY)_{i\in I}$ is a colimiting cocone of the directed diagram $H \circ U \circ D$. But $H \circ U = U \circ G$, so we can conclude that $(UG(d_i)\colon UGDi \to UGY)_{i\in I}$ is a colimiting cocone of the directed diagram $U \circ G \circ D$, as desired.

We can now proceed with the proof of the lemma. Recall from Remark 5.6 that G (from Eq. (5.2)) is a lifting of the endofunctor H. The functor H is finitary because for any set X, and any function $f \in HX$, there is the finite set $Z = \{x \in X \mid \exists a \in A \exists r > 0 \text{ such that } f(a) = (r, x)\}$ with $f \in HZ$. By Fact 1, the forgetful functor $U\colon \mathsf{CA} \to \mathsf{Set}$ creates directed colimits. Thus, the conditions of Fact 2 are satisfied, and we may conclude that G is finitary. ◀

▶ **Lemma 7.3.** *G preserves surjective affine maps.*

Proof. Let $h\colon X \to Y$ be a surjective affine map. Consider $G(h)\colon GX \to GY$. For $a \in A$, we have $G(h)(g)(a)(\bot) = \bot$ and $G(h)(g)(a)(r \cdot x) = r \cdot h(x)$.

Take $f \in GY$. For each $y \in Y$, denote by x_y an element of X with $y = h(x_y)$. Such exists since h is surjective. We define $g\colon A \to X_\bot$ as follows. For $a \in A$, if $f(a) = \bot$, set $g(a) = \bot$ and if $f(a) = r \cdot y$, set $g(a) = r \cdot x_y$. Then $g \in GX$ and $G(h)(g) = f$. ◀

The most interesting point in this section is the *properness* of G (see Definition 7.6). In order to verify that G is proper, we need a few lemmas regarding *bisimilarity* and *behavioural equivalence* for G-coalgebras.

▶ **Lemma 7.4.** *Let (X, γ) be a G-coalgebra on* CA*. Then bisimilarity (the largest bisimulation) on (X, c) coincides with behavioural equivalence, which in turn coincides with the final coalgebra semantics.*

Proof. Behavioural equivalence always coincides with the final coalgebra semantics if the functor admits a final coalgebra, which is the case for our functor G on CA.

CA is complete and cocomplete [2, § 9.3, Prop. 4] and the functor G preserves (weak) pullbacks by Lemma 7.1. So CA satisfies the requirements of [30, Theorem 4.1]. As a consequence: (1) every bisimulation is contained in a kernel bisimulation, and hence bisimilar states are behaviourally equivalent, and (2) every kernel bisimulation is a bisimulation, yielding that behaviourally equivalent states are bisimilar. ◀

We need one more lemma that characterises bisimilarity for G in concrete terms. The proof follows directly from the definition of bisimulation.

▶ **Lemma 7.5.** *Let (X, γ) and (Y, ϑ) be G-coalgebras. Let $R \subseteq X \times Y$ be a subalgebra of $X \times Y$. Then R is a bisimulation between (X, γ) and (Y, ϑ) if and only if the following holds: whenever $a \in A$ and $(x, y) \in R$, $\mathsf{mass}_\gamma(a, x) = \mathsf{mass}_\vartheta(a, y)$, and if $\mathsf{mass}_\gamma(a, x) = \mathsf{mass}_\vartheta(a, y) \neq 0$, then R contains $(\mathsf{next}_\gamma(a, x), \mathsf{next}_\vartheta(a, y))$.*

Without further ado, let us now proceed with the proof that G is a proper functor, in the following sense.

▶ **Definition 7.6.** *Let T be a finitary monad on* Set *and write* Set^T *for the Eilenberg-Moore category of T. A* zig-zag *in* $\mathsf{Coalg}_{\mathsf{Set}^T}(F)$ *is a diagram of the shape*

$$(X, c) \xrightarrow{f_1} (Z_1, e_1) \xleftarrow{f_2} (Z_2, e_2) \xrightarrow{f_3} (Z_3, e_3) \xleftarrow{f_4} \cdots \xrightarrow{f_{2n-1}} (Z_{2n-1}, e_{2n-1}) \xleftarrow{f_{2n}} (Y, d) \tag{7.1}$$

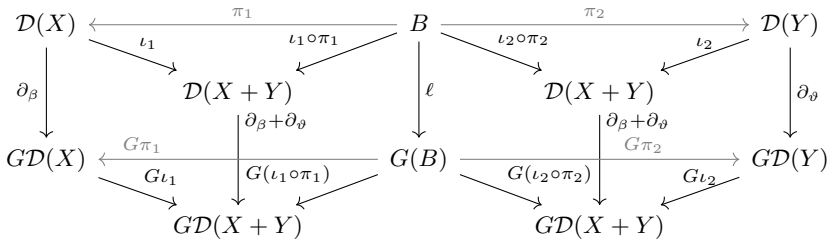
Write η for the unit of T. The zig-zag above relates $x \in X$ with $y \in Y$, *written $x \sim y$, if there exist elements $z_{2k} \in Z_{2k}$, $k = 1, \ldots, n-1$, with (setting $z_0 = x$ and $z_{2n} = y$)*

$$f_{2k}(z_{2k}) = f_{2k-1}(z_{2k-2}), \quad k = 1, \ldots, n$$

The endofunctor F is said to be proper *if the following statement holds: for any pair of ffg F-coalgebras $(T(X), c^X)$ and $(T(Y), c^Y)$ and any two elements $x \in X$ and $y \in Y$ with $\eta_X(x) \sim \eta_Y(y)$, there exists a zig-zag in $\mathsf{Coalg}_{\mathsf{Set}^T}(F)$ entirely consisting of ffg F-coalgebras that relates $\eta_X(x)$ with $\eta_Y(y)$. We may call such a zig-zag an ffg zig-zag.*

▶ **Theorem 7.7.** *The functor $G : \mathsf{CA} \to \mathsf{CA}$ is proper.*

Proof. Consider two ffg G-coalgebras $(\mathcal{D}(X), \partial_\beta)$ and $(\mathcal{D}(Y), \partial_\vartheta)$, with behaviourally equivalent states $\varphi \in \mathcal{D}(X)$ and $\psi \in \mathcal{D}(Y)$. We need to relate φ and ψ with a suitable, ffg, zig-zag. We are going to use bisimilarity B on the coproduct coalgebra[2] $(\mathcal{D}(X), \partial_\beta) + (\mathcal{D}(Y), \partial_\vartheta) \cong (\mathcal{D}(X+Y), \partial_\beta + \partial_\vartheta)$.



where ι_1, ι_2 denote the coproduct injections. It remains to show that B is finitely generated as a subalgebra of the product CA, $\mathcal{D}(X+Y) \times \mathcal{D}(X+Y)$. This follows from results below, using an analytic-algebraic characterization of finitely generated congruences (kernels of convex algebra homomorphisms) of finitely generated convex algebras.

In more detail, note that we can identify any ffg algebra $\mathcal{D}(X)$ with the simplex in the vector space $\mathbb{R}^X$. This can be done by seeing each Dirac delta $1 \cdot x$ as a unit vector in $\mathbb{R}^X$. Every congruence relation $R \subseteq \mathcal{D}(X) \times \mathcal{D}(X)$ of convex algebras is a subalgebra of $\mathcal{D}(X) \times \mathcal{D}(X)$, and so by extension can be identified with a (convex) subset of $\mathbb{R}^X \times \mathbb{R}^X \cong \mathbb{R}^{2X}$. In particular, our B can be identified with a convex subset of $\mathbb{R}^{2(X+Y)}$. As turns out, B is finitely generated as a subalgebra if and only if B is topologically closed in $\mathbb{R}^{2(X+Y)}$. The following theorem is a direct consequence of Sokolova-Woracek [27, Proposition 5.9].

▶ **Theorem 7.8.** *Let $R \subseteq \mathbb{R}^{2X}$ be a congruence on the ffg convex algebra $\mathcal{D}(X) \subseteq \mathbb{R}^X$. Then R is finitely generated as a subalgebra if and only if it is topologically closed (closed under limits of Cauchy sequences).*

▶ **Lemma 7.9.** *Let $(\mathcal{D}(X), \partial_\beta)$ be a G-coalgebra. Then for any $a \in A$, the maps $\partial_\beta(-)(a)$ and $\mathsf{mass}_\beta(a, -)$ are restrictions of $\mathbb{R}$-linear maps $\mathbb{R}^X \to \mathbb{R}^{X+1}$ and $\mathbb{R}^X \to \mathbb{R}$ respectively.*

Proof. Recall that we think of the Dirac distributions $1 \cdot x$ as the basis vectors of $\mathbb{R}^X$. We additionally have the unit vector $1 \cdot \bot$ in $\mathbb{R}^{X+1}$. For $x \in X$, write

$$\partial_\beta(x)(a) = \sum_{y \in X} r_{xy} \cdot y$$

and $r_{x\bot} = 1 - \sum_{y \in X} r_{xy}$. Define the matrix M by

$$M = \left[r_{x\xi} \;\middle|\; x \in X \text{ and } \xi \in X \cup \{\bot\} \right]$$

indexed by $X \times (X \cup \{\bot\})$. A quick calculation verifies that indeed, for $\theta \in \mathcal{D}(X)$, $\partial_\beta(\theta)(a) = M\theta$ by linear extension. Of course, here we are thinking of $\theta = \sum_{x \in X} q_x \cdot x$ as the column vector $[q_x \mid x \in X]$.

Similarly, define the row matrix $N = [1 \mid x \in X]$ of 1's. Then for $\theta = \sum_{x \in X} q_x \cdot x$,

$$N\theta = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} [q_x \mid x \in X] = \sum_{x \in X} q_x$$

[2] Left adjoints preserve colimits, so indeed the coproduct of free convex algebras is given by the formula $\mathcal{D}(X) + \mathcal{D}(Y) \cong \mathcal{D}(X+Y)$, where the "+" on the left hand side is the coproduct in CA.

We therefore have $\mathsf{mass}_\beta(a,\theta) = NM\theta$. Thus, both $\partial_\beta(-)(a)$ and $\mathsf{mass}_\beta(a,-)$ are restrictions of linear functions. ◀

▶ **Corollary 7.10.** *Let $(\mathcal{D}(X),\partial_\beta)$ be a G-coalgebra. Then for any $a \in A$, the maps $\partial_\beta(-)(a)$ and $\mathsf{mass}_\beta(a,-)$ are continuous.*

Proof. Follows directly from Lemma 7.9 and that $\mathbb{R}^X$, $\mathbb{R}^{X+1}$, and $\mathbb{R}$ are finite dimensional. ◀

▶ **Theorem 7.11.** *Let $(\mathcal{D}(X),\partial_\beta)$ and $(\mathcal{D}(Y),\partial_\vartheta)$ be free finitely generated G-coalgebras. Let (B,ℓ) be the largest bisimulation between $\mathcal{D}(X)$ and $\mathcal{D}(Y)$, and regard B as a subset of $\mathcal{D}(X+Y)\times\mathcal{D}(X+Y) \subseteq \mathbb{R}^{2(X+Y)}$. Then B is a closed set and thus is finitely generated as a subalgebra.*

Proof. We show that the topological closure $\overline{B}$ of $B \subseteq \mathbb{R}^{2(X+Y)}$ is a bisimulation between $(\mathcal{D}(X),\partial_\beta)$ and $(\mathcal{D}(Y),\partial_\vartheta)$. Since B is the largest bisimulation, $B \subseteq \overline{B} \subseteq B$.

We appeal to Lemma 7.5: Let $(\theta,\psi) \in \overline{B}$. Then there is a Cauchy sequence $(\theta_i,\psi_i)_{i\in\mathbb{N}}$ such that $(\theta_i,\psi_i) \to (\theta,\psi)$ as $i \to \infty$. This, in particular, means that $\theta_i \to \theta$ and $\psi_i \to \psi$ in the product topology. Now, for $a \in A$,

$$\begin{aligned}
\mathsf{mass}_\beta(a,\theta) &= \mathsf{mass}_\beta(a,\lim\theta_i) \\
&= \lim \mathsf{mass}_\beta(a,\theta_i) && \text{(Corollary 7.10)} \\
&= \lim \mathsf{mass}_\vartheta(a,\psi_i) && \text{(Lemma 7.5)} \\
&= \mathsf{mass}_\vartheta(a,\lim\psi_i) && \text{(Corollary 7.10)} \\
&= \mathsf{mass}_\vartheta(a,\psi)
\end{aligned}$$

This verifies the first condition. To verify the second, suppose that $\mathsf{mass}_\beta(a,\theta) = \mathsf{mass}_\vartheta(a,\psi) \neq 0$. Then there is an $N > 0$ such that for all $i > N$, $\mathsf{mass}_\beta(a,\theta_i) = \mathsf{mass}_\vartheta(a,\psi_i) > 0$. This allows for the following computation:

$$\mathsf{next}_\beta(a,\theta) = \frac{\partial_\beta(\theta)(a)}{\mathsf{mass}_\beta(a,\theta)} \overset{(*)}{=} \lim \frac{\partial_\beta(\theta_i)(a)}{\mathsf{mass}_\beta(a,\theta_i)} = \lim \mathsf{next}_\beta(a,\theta_i)$$

and similarly for ψ. Above, the step tagged (*) is due to the fact that a product of continuous functions is continuous on the intersection of their domain, which in this case contains all of the θ_i as well as θ. Simply put, we use a known rule for computing limits of sequences of fractions: The limit of the pointwise-fractions of two sequences is the quotient of the two limits, given that the denominator sequence has non-zero limit. This tells us that

$$(\mathsf{next}_\beta(a,\theta),\mathsf{next}_\vartheta(a,\psi)) = \lim(\mathsf{next}_\beta(a,\theta_i),\mathsf{next}_\vartheta(a,\psi_i)) \in \overline{B}$$

By Lemma 7.5, $\overline{B}$ is a bisimulation, as desired. ◀

At long last, we complete the proof of Theorem 7.7 with an appeal to Theorem 7.11. ◀

Recap of the proof of completeness, Theorem 3.6

We have taken the approach outlined in Section 4 to showing that the axioms in Fig. 1 are complete with respect to the stream semantics of probabilistic process terms (Proposition 2.3). In Step 1, we observed that the semantics map $[\![-]\!]$ coincides with determinized stream semantics $(\!|-|\!)$ (Theorem 5.13), and that in particular this meant that the final G-coalgebra

homomorphism $\partial^\dagger : (\mathsf{PTerm}/{\equiv}, \partial) \to (\mathsf{Prob}(A^\omega), \zeta)$ satisfies $[\![e]\!] = \partial^\dagger([e])$ for each $e \in \mathsf{PTerm}$ (Theorem 6.3). Thus, it suffices to show that $\partial^\dagger$ is injective. To this end, we observed in Section 4 that it suffices to construct a left inverse k to q in the diagram below.

$$(\mathsf{PTerm}/{\equiv}, \partial) \underset{k}{\overset{q}{\rightleftarrows}} (J, \rho) \xhookrightarrow{\iota} (\mathsf{Prob}(A^\omega), \zeta) \qquad \partial^\dagger : (\mathsf{PTerm}/{\equiv}, \partial) \to (\mathsf{Prob}(A^\omega), \zeta) \tag{7.2}$$

The left inverse k in (7.2) exists if $(\mathsf{PTerm}/{\equiv}, \partial^\dagger)$ is the final locally fg G-coalgebra. In Step 2, we saw that $(\mathsf{PTerm}/{\equiv}, \partial^\dagger)$ satisfies a slightly weaker universal property, that every ffg G-coalgebra admits a unique coalgebra homomorphism into it (Theorem 6.9). In Step 3, we verified the hypotheses of Theorem 4.4, in particular Theorem 7.7, which tells us that in fact, $(\mathsf{PTerm}/{\equiv}, \partial^\dagger)$ is the final locally fg coalgebra, as desired. This finishes the proof of completeness, Theorem 3.6.

8 Discussion and Related Work

We present the first sound and complete axiomatization of *infinite* trace semantics for generative probabilistic transition systems, settling a recent conjecture of Schmid, Noquez, and Moss [21]. Our completeness theorem on infinite traces is a new direction in a series of coalgebraic completeness theorems on finite trace semantics for probabilistic process calculi [25, 18], thus expanding the scope of this line of work. Our approach is categorical, and we build on recent results on proper functors over convex sets. In our proof, we use an analytic-algebraic result about convex congruences to show properness of G. The particular functor which we prove to be proper has not been studied before, and the properness proof technique of [28] does not apply to it, but remarkably we could use a result concerning the geometry of convex congruences due to Sokolova and Woracek [27].

We provide a characterization of infinite traces as the final coalgebra semantics of a functor over convex algebras. Infinite traces have been studied in the context of semantics of (variants of GPTS) before: via a largest homomorphism in the (order enriched) Kleisli category of the Giry monad [32] due to Urabe and Hasuo, via a greatest fixpoint in a category of generalised relations [4] due to Cîrstea, as a final coalgebra on a free positive convex algebra (a convex algebra with a distinguished element, i.e., in the Kleisli category of the subdistribution monad) due to Kerstan and König [9], and as a subcoalgebra of the final Moore automaton on a positive convex algebra (in the Eilenberg-Moore category of the subdistribution monad) due to Goy and Rot [5, 6]. We offer a fourth characterization as a final coalgebra semantics for a new functor on convex algebras (i.e., in the Eilenberg-Moore category of the finite probability distribution monad) in Section 5. It is also the final coalgebra of a set functor.

In the future, we want to explore whether the argument we provided for properness generalizes to other endofunctors on CA and to endofunctors on the category of positive convex algebras used in [25, 18]. We would like to expand our completeness theorem to incorporate hypotheses, especially in the context [21] where actions are interpreted concretely as contractions on a space: If the space and the contractions are fixed, the actions might satisfy additional relations. More speculatively, it might be interesting to also go in the opposite direction: Given a set of hypotheses, can one construct a canonical space and a contraction interpretation of the actions that satisfies the hypotheses? We would also like to consider different syntax for specifying LMCs and stream measures, such as the so-called formal language of recursion [7], which connects nicely to iterative algebra. Orthogonally, we would like to explore axiomatizations of behavioural distances, in the style of quantitative equational theories [12]. Last but not least, we would like to explore unifying the results of Silva and Sokolova [25] with those of this paper.

References

1 J. Adámek and J. Rosicky. *Locally Presentable and Accessible Categories*. London Mathematical Society Lecture Note Series. Cambridge University Press, 1994.

2 M. Barr and Ch. Wells. *Toposes, Triples and Theories*. Springer, Berlin, 1985. Revised and corrected version available from URL: `www.cwru.edu/artsci/math/wells/pub/ttt.html`.

3 F. Bartels, A. Sokolova, and E.P. de Vink. A hierarchy of probabilistic system types. *Theoretical Computer Science*, 327:3–22, 2004. `doi:10.1016/J.TCS.2004.07.019`.

4 Corina Cîrstea. From branching to linear time, coalgebraically. *Fundam. Informaticae*, 150(3-4):379–406, 2017. `doi:10.3233/FI-2017-1474`.

5 Alexandre Goy. Trace semantics via determinization for probabilistic transition systems. *CoRR*, abs/1802.09084, 2018. `arXiv:1802.09084`.

6 Alexandre Goy and Jurriaan Rot. (In)finite trace equivalence of probabilistic transition systems. In Corina Cîrstea, editor, *Coalgebraic Methods in Computer Science - 14th IFIP WG 1.3 International Workshop, CMCS 2018*, volume 11202 of *Lecture Notes in Computer Science*, pages 100–121. Springer, 2018. `doi:10.1007/978-3-030-00389-0_7`.

7 Antonius J. C. Hurkens, Monica McArthur, Yiannis N. Moschovakis, Lawrence S. Moss, and Glen T. Whitney. The logic of recursive equations. *J. Symb. Log.*, 63(2):451–478, 1998. `doi:10.2307/2586843`.

8 Bart Jacobs. A bialgebraic review of deterministic automata, regular expressions and languages. In Kokichi Futatsugi, Jean-Pierre Jouannaud, and José Meseguer, editors, *Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday*, volume 4060 of *Lecture Notes in Computer Science*, pages 375–404. Springer, 2006. `doi:10.1007/11780274_20`.

9 Henning Kerstan and Barbara König. Coalgebraic Trace Semantics for Continuous Probabilistic Transition Systems. *Logical Methods in Computer Science*, Volume 9, Issue 4, December 2013. `doi:10.2168/LMCS-9(4:16)2013`.

10 S. C. Kleene. Representation of events in nerve nets and finite automata. In Claude Shannon and John McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, Princeton, NJ, 1956.

11 Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991. `doi:10.1016/0890-5401(91)90030-6`.

12 Radu Mardare, Prakash Panangaden, and Gordon Plotkin. Quantitative algebraic reasoning. In *2016 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, 2016.

13 Stefan Milius. A sound and complete calculus for finite stream circuits. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010*, pages 421–430. IEEE Computer Society, 2010. `doi:10.1109/LICS.2010.11`.

14 Stefan Milius. Proper functors and fixed points for finite behaviour. *Log. Methods Comput. Sci.*, 14(3), 2018. `doi:10.23638/LMCS-14(3:22)2018`.

15 Robin Milner. A complete inference system for a class of regular behaviours. *J. Comput. Syst. Sci.*, 28(3):439–466, 1984. `doi:10.1016/0022-0000(84)90023-0`.

16 Michael O. Rabin. Probabilistic automata. *Inf. Control.*, 6(3):230–245, 1963. `doi:10.1016/S0019-9958(63)90290-0`.

17 Alexander Moshe Rabinovich. A complete axiomatisation for trace congruence of finite state behaviors. In Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics, 9th International Conference, New Orleans, LA, USA, April 7-10, 1993, Proceedings*, volume 802 of *Lecture Notes in Computer Science*, pages 530–543. Springer, 1993. `doi:10.1007/3-540-58027-1_25`.

18 Wojciech Rozowski and Alexandra Silva. A completeness theorem for probabilistic regular expressions. In Pawel Sobocinski, Ugo Dal Lago, and Javier Esparza, editors, *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2024, Tallinn, Estonia, July 8-11, 2024*, pages 66:1–66:14. ACM, 2024. `doi:10.1145/3661814.3662084`.

19 Walter Rudin. *Real and Complex Analysis*. McGraw-Hill, 1966.

20 Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000. doi:10.1016/S0304-3975(00)00056-6.

21 Todd Schmid, Victoria Noquez, and Lawrence S. Moss. Fractals from regular behaviours. In Paolo Baldan and Valeria de Paiva, editors, *10th Conference on Algebra and Coalgebra in Computer Science, CALCO 2023, June 19-21, 2023, Indiana University Bloomington, IN, USA*, volume 270 of *LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. Also available at https://arxiv.org/pdf/2306.03894. doi:10.4230/LIPICS.CALCO.2023.14.

22 Todd Schmid, Jurriaan Rot, and Alexandra Silva. On star expressions and coalgebraic completeness theorems. In Ana Sokolova, editor, *Proceedings 37th Conference on Mathematical Foundations of Programming Semantics, MFPS 2021, Hybrid: Salzburg, Austria and Online, 30th August - 2nd September, 2021*, volume 351 of *EPTCS*, pages 242–259, 2021. doi:10.4204/EPTCS.351.15.

23 Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Log. Methods Comput. Sci.*, 9(1), 2013. doi:10.2168/LMCS-9(1:9)2013.

24 Alexandra Silva, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Non-deterministic kleene coalgebras. *Log. Methods Comput. Sci.*, 6(3), 2010. URL: http://arxiv.org/abs/1007.3769.

25 Alexandra Silva and Ana Sokolova. Sound and complete axiomatization of trace semantics for probabilistic systems. In Michael W. Mislove and Joël Ouaknine, editors, *Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics, MFPS 2011, Pittsburgh, PA, USA, May 25-28, 2011*, volume 276 of *Electronic Notes in Theoretical Computer Science*, pages 291–311. Elsevier, 2011. doi:10.1016/j.entcs.2011.09.027.

26 A. Sokolova and E.P. de Vink. Probabilistic automata: system types, parallel composition and comparison. In C. Baier, B.R. Haverkort, H. Hermanns, J.-P. Katoen, and M. Siegle, editors, *Validation of Stochastic Systems: A Guide to Current Research*, pages 1–43. LNCS 2925, 2004. doi:10.1007/978-3-540-24611-4_1.

27 Ana Sokolova and Harald Woracek. Congruences of convex algebras. *Journal of Pure and Applied Algebra*, 219(8):3110–3148, 2015. doi:10.1016/j.jpaa.2014.10.005.

28 Ana Sokolova and Harald Woracek. Proper semirings and proper convex functors. In *FoSSaCS 2018*, pages 331–347. LNCS 10803, 2018.

29 Eugene W. Stark and Scott A. Smolka. A complete axiom system for finite-state probabilistic processes. In Gordon D. Plotkin, Colin Stirling, and Mads Tofte, editors, *Proof, Language, and Interaction, Essays in Honour of Robin Milner*, pages 571–596. The MIT Press, 2000.

30 Sam Staton. Relating coalgebraic notions of bisimulation. In *CALCO 2009*, volume 5728, pages 191–205. LNCS 5728, 2009. doi:10.1007/978-3-642-03741-2_14.

31 T. Świrszcz. Monadic functors and convexity. *Bull. Acad. Polon. Sci. Sér. Sci. Math. Astronom. Phys.*, 22:39–42, 1974.

32 Natsuki Urabe and Ichiro Hasuo. Coalgebraic infinite traces and Kleisli simulations. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:15)2018.

Simple Types for Probabilistic Termination

Willem Heijltjes
Department of Computer Science, University of Bath, UK

Georgina Majury
Department of Computer Science, University of Bath, UK

Abstract

We present a new typing discipline to guarantee the probability of termination in probabilistic lambda-calculi. The main contribution is a particular naturality and simplicity: our probabilistic types are as simple types, but generated from probabilities as base types, representing a least probability of termination. Simple types are recovered by restricting probabilities to one.

Our vehicle is the Probabilistic Event Lambda-Calculus by Dal Lago, Guerrieri, and Heijltjes, which presents a solution to the issue of confluence in probabilistic lambda-calculi. Our probabilistic type system provides an alternative solution to that using counting quantifiers by Antonelli, Dal Lago, and Pistone, for the same calculus.

The problem that both type systems address is to give a lower bound on the probability that terms head-normalize. Following the recent Functional Machine Calculus by Heijltjes, our development takes the (simplified) Krivine machine as primary, and proceeds via an extension of the calculus with sequential composition and identity on the machine. Our type system then gives a natural account of termination probability on the Krivine machine, reflected back onto head-normalization for the original calculus. In this way we are able to avoid the use of counting quantifiers, while improving on the termination bounds given by Antonelli, Dal Lago, and Pistone.

2012 ACM Subject Classification Theory of computation → Lambda calculus; Theory of computation → Type theory; Theory of computation → Probabilistic computation

Keywords and phrases lambda-calculus, probabilistic termination, simple types

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.31

Acknowledgements We would like to thank Melissa Antonelli, Ugo Dal Lago, and Paolo Pistone for the constructive conversation about both our approaches, and the anonymous referees for their helpful commentary.

1 Introduction

While the study of probabilistic computation can be traced to the 1950s [11], the first study of the probabilistic λ-calculus in particular is considered to be by Saheb-Djahromi in the late 70s [30]. In the near half century since, many variations on higher-order probabilistic computation have been considered [10, 12, 19, 20, 26, 29]. In recent years, perhaps due to the potential for applications in machine learning and modelling of probabilistic systems, the area has seen a return to popularity [5, 6, 16, 17, 24, 31]. An important computational phenomenon in its own right, the study of probabilistic choice can also provide a "foot in the door" for understanding how more general effects might manifest, leading for instance to the recent *Functional Machine Calculus* (FMC) as a confluent λ-calculus with effects [3, 18] that will play a central role in our development.

In this paper we consider the problem of *probabilistic termination*, the probability that a given reduction mechanism reaches the normal form of a term, a key consideration for probabilistic computation and the subject of significant recent attention [4, 7, 8, 15, 22]. [1]

[1] Note that this objective is distinct from *almost-sure* termination: it considers *exact* probabilities, not those converging in the limit. Almost-sure termination is more commonly studied for iterative constructs [21, 27]; extending the lambda-calculus with an almost-surely terminating iterator is the subject of future work.

Termination being one of the prime considerations of type systems in general, the question of type systems for probabilistic termination is pertinent [1, 7]. Our contribution in this paper is a new typing discipline for probabilistic termination, following the recent line of work on the *probabilistic event λ-calculus* [9] and the FMC. Their prime features are *confluence* for probabilistic computation and other effects, and the encoding of multiple reduction strategies within a single calculus.

One of the greatest challenges facing the study of probabilistic computation is confluence. In most variants the outcome of duplicating a probabilistic term is strategy dependent. Consider the instruction "write down the result of flipping a coin twice": it is ambiguous whether "twice" refers to "write" or "flipping", and the choice of interpretation changes the possible outcomes. In most of the literature results are therefore either restricted to a single strategy, such as call–by–value or call–by–name, or rely on a modified definition of reduction. This raises the question whether it is possible to have confluence for probabilistic computation, while expressing different strategies, and rewriting without restriction on contexts, properties which lend elegance to the λ-calculus. The *probabilistic event λ-calculus* [9] proposes a solution to this problem by decomposing a probabilistic sum $M \oplus N$ into a *choice* operation $M\, a\, N$, understood as a conditional "*if* a *then* M *else* N", and a probabilistic *generator* $\boxed{a}.\, P$, representing a coin toss whose outcome is bound to the boolean variable a in the term P, which then determines the choice for $M\, a\, N$. When in argument position $\boxed{a}.\, M$ acts as a wrapper, similar to *thunking* in call–by–push value [25] and the bang-calculus [14, 15], protecting the operator from evaluation. These factors combine to return confluence to the calculus, allowing for arbitrary evaluation strategies with an unrestricted β-reduction.

The probabilistic event λ-calculus (PEΛ) was introduced in [9] with a simple type system, corresponding to that on the lambda-calculus. This system ignored the probabilistic elements of the calculus, and thus gave little insight into the properties of this extension. In a deterministic calculus a type system provides various safety guarantees. These may be qualitative: termination, outputs, composability; or quantitative: run time, term size. In the probabilistic setting, however, it is natural to wonder what guarantees can be made, when the behaviour of a single term can vary between iterations. If, as in the type system mentioned, the probabilities are ignored, strong results can be obtained, albeit on a fairly uninteresting fragment of the calculus. Once probability is acknowledged by the type system the guarantees made become similarly qualified: probability of termination, almost-sure termination, expected run time. Here, we consider the first of these.

One proposed type system for the PEΛ [1], labelled C$\lambda_{\to}$, introduces *counting quantifiers* as a Curry–Howard correspondent to an intuitionistic counting propositional logic. These provide a mechanism to express "proportion of truth" by quantifying the number of satisfying assignments to a formula within a given model, in the way that existential and universal quantification may be understood via the existence of a satisfying assignment, respectively the satisfaction of all assignments. By taking the assignments to a formula to describe the branches of a probabilistic computation, counting quantifiers may be used to describe probability bounds. By this mechanism, C$\lambda_{\to}$ provides a lower bound on the probability of head normalisation. However, as illustrated in [1], the bounds provided by C$\lambda_{\to}$ are not tight, even in situations where this would be expected (see Example 8).

In this paper we present an alternative approach to the same problem, for the same calculus, using a different inspiration, to provide improved termination bounds. Deriving from the PEΛ, the FMC provides an additional feature that, to us, seemed crucial to a natural account of probabilistic termination via the type system. First, the "machine" in question is the (simplified) *Krivine machine* [23], whose evaluation is closely related to head

normalisation. The FMC here adds an intriguing aspect: it introduces *sequential composition* and *identity* on the machine, where the latter, the imperative *skip*, provides a notion of *successful termination*, notably absent from the machine for standard λ-calculus. Moreover, this becomes the primary interpretation of types: a type derivation in the FMC is a proof that the machine successfully terminates. Our main question for this work was how to adapt this to capture *probabilistic termination*. The answer, described in Section 6, is an extended *sequential probabilistic event* λ*-calculus* $\mathsf{SPE}\Lambda$, with a type system $\mathsf{SPE}\Lambda^{\mathbb{Q}\Rightarrow}$ that naturally describes probabilistic termination of the machine, as well as probabilistic termination of head reduction.

The type system $\mathsf{SPE}\Lambda^{\mathbb{Q}\Rightarrow}$ for the extended sequential calculus reflects back onto a natural type system for the original probabilistic event λ-calculus, $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$, which gives probabilistic head normalization in the following way: types generalize simple types by replacing the base type with a *probability*. Formally,

$$A, B \ ::= \ p \ \mid \ A \to B$$

where $p \in [0,1] \cap \mathbb{Q}$ is a rational number between zero and one inclusive. The intuition is that the base type for simple types, o, may be understood as signifying successful evaluation, even if it is uninhabited. After all, a constant base type such as for booleans or integers signifies the successful return of a corresponding value. This further matches the interpretation in the FMC, where the type o is inhabited by *skip*, and corresponds to termination of the machine without producing a result. Our approach, then, is to replace *certain* termination with a *probability* of termination, replacing the base type o with a probability p. We consider the striking simplicity and natural intuition of this approach one of our main contributions.

2 The probabilistic event lambda-calculus

We recall the *probabilistic event lambda-calculus* $\mathsf{PE}\Lambda$ from [9]. Assume countable sets of *variables*, ranged over by x, y, z, and of *events*, ranged over by a, b, c. The former are *term* variables, to be instantiated by terms of the calculus, and the latter are *boolean* variables, to be instantiated by $\top$ (*true*) or $\bot$ (*false*). The $\mathsf{PE}\Lambda$ extends the λ-calculus with a *generator* $\boxed{a}.\, M$, which flips a coin and binds the result ($\top$ or $\bot$) to a, and a *choice* or *conditional* $M\, a\, N$, which evaluates to M if a is *true*, and to N otherwise. A traditional probabilistic sum of terms, $M \oplus N$, may be encoded as $\boxed{a}.\, M\, a\, N$. Generators are normally *fair*, with equal probability for $\top$ or $\bot$, though on occasion we may need a *biased* generator $\boxed{a}_p$ which chooses $\top$ with probability p, and $\bot$ with $1-p$.

▶ **Definition 1.** Terms *are given by the following grammar,*

$$M, N \ ::= \ x \ \mid \ \lambda x.\, M \ \mid \ M\, N \ \mid \ \boxed{a}.\, M \ \mid \ M\, a\, N$$

with, from left to right: a variable*; an* abstraction*, which binds* x *in* M*; an* application*; a* generator*, which binds* a *in* M*; and a* choice*.*

The *free variables* and *free events* of a term M are written $\mathsf{fv}(M)$ and $\mathsf{fe}(M)$ respectively. Substitution is written prefix: $\{N/x\}M$ is the capture-avoiding substitution of N for x in M. For an event a we define two *projection* functions $\pi^a_\top$ and $\pi^a_\bot$, which apply the effect of instantiating a with *true* and *false* respectively, to a term M.

▶ **Definition 2.** *The* projection *functions* π_i^a *for an event* a *and* $i \in \{\bot, \top\}$ *are given as follows, where* $a \neq b$.

$$\begin{aligned} \pi_i^a x &= x \\ \pi_i^a(\lambda x.\, M) &= \lambda x.\, \pi_i^a M \\ \pi_i^a(M\, N) &= (\pi_i^a M)\,(\pi_i^a N) \end{aligned} \qquad \begin{aligned} \pi_\top^a(M\, a\, N) &= \pi_\top^a M \\ \pi_\bot^a(M\, a\, N) &= \pi_\bot^a N \end{aligned} \qquad \begin{aligned} \pi_i^a(M\, b\, N) &= (\pi_i^a M)\, b\, (\pi_i^a N) \\ \pi_i^a(\boxed{b}.\, M) &= \boxed{b}.\, \pi_i^a M \end{aligned}$$

We use the standard notions of *context*, *head context*, and *applicative context* to define the reduction relations. Note that a head context is of the form $\lambda x_1 \ldots \lambda x_n.\, \{\}\, M_1 \ldots M_m$ where m and n are potentially zero.

▶ **Definition 3.** Contexts C, head *contexts* H, *and* applicative *contexts* A *are defined as follows. A context* C *with the hole replaced by* M, *capturing variables, is written* $C\{M\}$.

$$\begin{aligned} C \;::=\; & \{\} \mid \lambda x.\, C \mid C\, M \mid M\, C \\ & \mid \boxed{a}.\, C \mid C\, a\, M \mid M\, a\, C \end{aligned} \qquad \begin{aligned} H \;::=\; & \lambda x.\, H \mid A \\ A \;::=\; & A\, M \mid \{\} \end{aligned}$$

A *probability* p, q is a rational number between 0 and 1 inclusive. Probabilistic reduction will return a *multi-(sub-)distribution* [2], a finite multiset of weighted terms $\mathcal{M}$ written as $[p_1 \cdot M_1, \ldots, p_n \cdot M_n]$ whose *weight* $|\mathcal{M}| = \sum_{i \leq n} p_i$ is (at most) one.[2] For simplicity, we will refer to these as *distributions*, and we convert implicitly between terms M and the singleton distribution $[1 \cdot M]$. Multiset union is written $\mathcal{S} + \mathcal{T}$, and the empty multiset as $\varnothing$. A distribution $\mathcal{M}$ is scaled to $p\mathcal{M}$ by multiplying each weight in $\mathcal{M}$ by p. The underlying probability (sub-)distribution of $\mathcal{M}$, the finite function from terms to probabilities obtained by collecting like terms $p \cdot M$ and $q \cdot M$ as $(p{+}q) \cdot M$, is written $\lfloor \mathcal{M} \rfloor$.

▶ **Definition 4.** Beta-reduction $\twoheadrightarrow_\beta$ *and* head β-reduction $\twoheadrightarrow_{\beta \mathsf{h}}$ *are given by closing the beta-rule below under all contexts* C *respectively under head contexts* H, *and implicitly return a singleton distribution.*

$$(\lambda x.\, M)N \;\twoheadrightarrow_\beta\; \{N/x\}M$$

Projective reduction $\twoheadrightarrow_\pi$ *is the following reduction relation from terms to distributions.*

$$H\{\boxed{a}.\, M\} \;\twoheadrightarrow_\pi\; [\tfrac{1}{2} \cdot H\{\pi_\top^a M\}\,,\, \tfrac{1}{2} \cdot H\{\pi_\top^a M\}]$$

Head reduction $(\twoheadrightarrow_{\mathsf{h}}) = (\twoheadrightarrow_{\beta \mathsf{h}}) \cup (\twoheadrightarrow_\pi)$ *is the union of head* β*-reduction and projective reduction. Reduction is lifted to distributions of terms in the expected way: if* $M \twoheadrightarrow \mathcal{N}$ *then* $[p \cdot M] + \mathcal{M} \twoheadrightarrow (p\mathcal{N}) + \mathcal{M}$. *We write* $\twoheadrightarrow\!\!\!\!\twoheadrightarrow$ *for the reflexive-transitive closure of a reduction relation* $\twoheadrightarrow$.

The PEΛ features a second notion of reduction, *permutative reduction* $\twoheadrightarrow_{\mathsf{p}}$ [9], which gives a more fine-grained evaluation of probabilistic sums. It is this reduction for which confluence is particularly significant. The effect of permutative reduction is to bridge the gap between the decomposed operators $\boxed{a}$ and $M\, a\, N$ and the standard probabilistic sum $M \oplus N$, encoded as $\boxed{a}.\, M\, a\, N$, by internalizing the reduction of $\boxed{a}.\, M$ to the sum of its two projections [9, Proposition 29]:

$$\boxed{a}.\, M \;\twoheadrightarrow\!\!\!\!\twoheadrightarrow_{\mathsf{p}}\; \boxed{a}.\, (\pi_\top^a M)\; a\; (\pi_\bot^a M) \quad (\text{if } a \in \mathsf{fe}(M))\,.$$

[2] We use multi-distributions rather than distributions to accommodate the reduction measure for the proof of head normalization in Appendix A.2.

$$\frac{}{\Gamma,\, x\colon A \vdash x\colon A} \qquad \frac{\Gamma,\, x\colon A \vdash M\colon B}{\Gamma \vdash \lambda x.\, M\colon A \to B} \qquad \frac{\Gamma \vdash M\colon A \to B \qquad \Gamma \vdash N\colon A}{\Gamma \vdash M\, N\colon B}$$

$$\frac{\Gamma \vdash M\colon A \qquad \Gamma \vdash N\colon A}{\Gamma \vdash M\, a\, N\colon A} \qquad \frac{\Gamma \vdash M\colon A}{\Gamma \vdash \boxed{a}.\, M\colon A}$$

Figure 1 The simple type system $\mathsf{PE}\Lambda^{\to}$.

In this paper we will work with projective reduction, since we are interested in termination of head reduction. We will, on occasion, consider terms where probabilistic sums are of the traditional form $M \oplus N = \boxed{a}.\, M\, a\, N$ with $a \notin \mathsf{fe}(M), \mathsf{fe}(N)$. As per the above, these are effectively the normal forms of permutative reduction.

3 Probabilistic types

Before introducing our probabilistic type system, we recall *simple types* for the $\mathsf{PE}\Lambda$ [9].

▶ **Definition 5.** Simple types *are given by the following grammar.*

$$A, B \;::=\; o \;\mid\; A \to B$$

A typing context Γ *is a finite function from term variables to types, written as a sequence* $x_1\colon A_1, \ldots, x_n\colon A_n$. *A* typing judgment $\Gamma \vdash M\colon A$ *assigns the type* A *to the term* M *in the context* Γ. *The* simply-typed probabilistic event λ-calculus $\mathsf{PE}\Lambda^{\to}$ *is given by the typing rules in Figure 1.*

Simple types ignore the probabilistic constructs of the calculus, *generator* and *choice*, requiring only that branches of a choice have equal types. This gives the expected result: typed terms are strongly normalizing, since every possible branch of the computation is typed. For *probabilistic* termination, we wish to capture that a given fraction of all branches of a computation, terminates – where our notion of *termination* is given by *head normalization*.

Our approach derives from the following idea: if the base type o of simple types, even if not inhabited, denotes *certainty* of termination, then we may generalise this to *probability* of termination by replacing base types with arbitrary probabilities. This yields the following notion of types.

▶ **Definition 6.** Probabilistic types *are given by the following grammar, where* $p \in [0,1] \cap \mathbb{Q}$.

$$A, B \;::=\; p \;\mid\; A \to B$$

The intuitive meaning of, for example, assigning a term M a type $A \to B \to p$ is: given inputs of type A and B, M terminates with probability at least p. The identity term $I = \lambda x.\, x$ may be assigned any type $p \to p$: given an input N that terminates with probability p, the term $I\, N$ does so as well. The type system will include an axiom that any term, in particular a non-terminating one such as $\Omega = (\lambda x.\, xx)(\lambda x.\, xx)$, may be assigned a termination probability of zero. This is expressed by a type $A_1 \to \ldots \to A_n \to 0$: for any inputs A_1 through A_n, the term will terminate with probability at least zero.

Note that these types generalise simple types with a single, uninhabited base type o. A probabilistic system with multiple base types, say integers and booleans, would pair the return type with a probability, for instance an integer with $\frac{1}{2}$ probability, or a boolean with $\frac{1}{4}$ probability. The extension of the calculus with sequencing, in Sections 5 and 6, will give a more concrete account of how probabilities interact with return values and return types.

$$\frac{}{E \mid \Gamma, x\colon A \vdash x\colon A}\,\mathsf{var} \qquad \frac{}{E \mid \Gamma \vdash M\colon \overline{A} \to 0}\,\mathsf{zero} \qquad \frac{E \mid \Gamma \vdash M\colon \overline{A} \to q}{E \mid \Gamma \vdash M\colon \overline{A} \to p}\,\mathsf{low}\ (p < q)$$

$$\frac{E \mid \Gamma, x\colon A \vdash M\colon B}{E \mid \Gamma \vdash \lambda x.\, M\colon A \to B}\,\mathsf{abs} \qquad \frac{E \mid \Gamma \vdash M\colon A \to B \qquad E \mid \Gamma \vdash N\colon A}{E \mid \Gamma \vdash M\,N\colon B}\,\mathsf{app}$$

$$\frac{E, a \mapsto i \mid \Gamma \vdash M_i\colon A}{E, a \mapsto i \mid \Gamma \vdash M_\top\, a\, M_\bot\colon A}\,\mathsf{chc} \qquad \frac{E, a \mapsto \top \mid \Gamma \vdash M\colon \overline{A} \to p \qquad E, a \mapsto \bot \mid \Gamma \vdash M\colon \overline{A} \to q}{E \mid \Gamma \vdash \boxed{a}.\, M\colon \overline{A} \to \tfrac{1}{2}p + \tfrac{1}{2}q}\,\mathsf{gen}$$

Figure 2 The probabilistic type system $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$.

The term $\boxed{a}.\, M\, a\, \Omega$ gives a fair probabilistic choice between M and Ω. For the computation to be well-typed regardless of the choice, M and Ω should have the same input types, so that they can be applied to the same arguments. Hence, if M has type $A \to B \to p$, we may choose $A \to B \to 0$ for Ω. Then the type for $\boxed{a}.\, M\, a\, \Omega$ should be $A \to B \to \frac{1}{2}p$: given arguments of type A and B, the computation chooses M with probability $\frac{1}{2}$ and so terminates with probability at least $\frac{1}{2}p$.

These considerations motivate the shape of our probabilistic type system, with one further aspect to explain. An arbitrary generator term $\boxed{a}.\, M$ reduces with a fair probability to either $\pi^a_\top M$ or $\pi^a_\bot M$, which may then terminate with different probabilities. However, using projections would mean the type system loses the property of being inductive on terms. We will instead record the assignment of a truth value to a in an additional context E, that we call an *event valuation*. The type derivation then projects a term $M\, a\, N$ to M or to N according to the value of a in the event valuation E.

▶ **Definition 7.** *The* probabilistically typed probabilistic event λ-calculus $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ *is given by the typing rules in Figure 2, using the following definitions. An* event valuation E *is a finite function from events to* $\{\bot, \top\}$*, written as a sequence* $a_1 \mapsto i_1, \ldots, a_n \mapsto i_n$*. A* typing context Γ *is a finite function from variables to types, written* $x_1\colon A_1, \ldots, x_n\colon A_n$*. A* probabilistic typing judgement $E \mid \Gamma \vdash M\colon A$ *assigns a type* A *to the term* M *in the context of* E *and* Γ*. A sequence of antecedents* $A_1 \to \ldots \to A_n \to p$ *is abbreviated with vector notation as* $\overline{A} \to p$*.*

The typing rules are syntax-driven, except for the low rule to lower a given bound. The rule is not essential to the type system, since it is already implied in the meaning of types as giving a lower bound; but for the same reason, since it is implied, including it brings more clarity than omitting it. Note further that the typing rule gen assumes a fair coin toss for the generator $\boxed{a}$, which gives the resulting probability of $\frac{1}{2}p + \frac{1}{2}q$. An unfair toss $\boxed{a}_r$ with ratio r would give a probability $rp + (1-r)q$.

▶ **Example 8.** To demonstrate our type system we reprise the example $t[a, b]$ from [1, Section 6.2], written here as the term $\boxed{a}.\, \boxed{b}.\, \boxed{c}.\, T$ below. In Figure 3 we derive the following type.

$$\boxed{a}.\, \boxed{b}.\, \boxed{c}.\, T\colon 1 \to \tfrac{3}{8} \quad \text{where} \quad T = ((I\, c\, \Omega)\, b\, \Omega)\, a\, (\Omega\, b\, I)$$

We will discuss a number of aspects of our type system. First, as a particularly natural feature, observe that the rules var, abs and app make it conservative over standard simply-typed λ-calculus – which, interestingly, is agnostic to the choice of base types.

$$D_{\bot,\bot,\bot} = \dfrac{\dfrac{\dfrac{\dfrac{}{a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid x\colon 1\vdash x\colon 1}\,\mathsf{var}}{a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid \vdash \lambda x.\,x\colon 1\to 1}\,\mathsf{abs}}{a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid \vdash I\,c\,\Omega\colon 1\to 1}\,\mathsf{chc}}{a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid \vdash (I\,c\,\Omega)\,b\,\Omega\colon 1\to 1}\,\mathsf{chc}}{a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid \vdash ((I\,c\,\Omega)\,b\,\Omega)\,a\,(\Omega\,b\,I)\colon 1\to 1}\,\mathsf{chc}$$

$$D_{\bot,\bot} = \dfrac{D_{\bot,\bot,\bot}\quad a\mapsto\bot,b\mapsto\bot,c\mapsto\bot \mid \vdash T\colon 1\to 1 \qquad \dfrac{}{a\mapsto\bot,b\mapsto\bot,c\mapsto\top \mid \vdash T\colon 1\to 0}\,\mathsf{zero}}{a\mapsto\bot,b\mapsto\bot \mid \vdash \boxed{c}.\,T\colon 1\to\tfrac{1}{2}}\,\mathsf{gen}$$

$$D_{\bot} = \dfrac{D_{\bot,\bot}\quad a\mapsto\bot,b\mapsto\bot \mid \vdash \boxed{c}.\,T\colon 1\to\tfrac{1}{2} \qquad \dfrac{}{a\mapsto\bot,b\mapsto\top \mid \vdash \boxed{c}.\,T\colon 1\to 0}\,\mathsf{zero}}{a\mapsto\bot \mid \vdash \boxed{b}.\,\boxed{c}.\,T\colon 1\to\tfrac{1}{4}}\,\mathsf{chc}$$

$$D_{\top,\top,i} = \dfrac{\dfrac{\dfrac{\dfrac{}{a\mapsto\top,b\mapsto\top,c\mapsto i \mid x\colon 1\vdash x\colon 1}\,\mathsf{var}}{a\mapsto\top,b\mapsto\top,c\mapsto i \mid \vdash \lambda x.\,x\colon 1\to 1}\,\mathsf{abs}}{a\mapsto\top,b\mapsto\top,c\mapsto i \mid \vdash \Omega\,b\,I\colon 1\to 1}\,\mathsf{chc}}{a\mapsto\top,b\mapsto\top,c\mapsto i \mid \vdash ((I\,c\,\Omega)\,b\,\Omega)\,a\,(\Omega\,b\,I)\colon 1\to 1}\,\mathsf{chc}$$

$$D_{\top,\top} = \dfrac{D_{\top,\top,\bot}\quad a\mapsto\top,b\mapsto\top,c\mapsto\bot \mid \vdash T\colon 1\to 1 \qquad D_{\top,\top,\top}\quad a\mapsto\top,b\mapsto\top,c\mapsto\top \mid \vdash T\colon 1\to 1}{a\mapsto\top,b\mapsto\top \mid \vdash \boxed{c}.\,T\colon 1\to 1}\,\mathsf{gen}$$

$$D_{\top} = \dfrac{\dfrac{}{a\mapsto\top,b\mapsto\bot \mid \vdash \boxed{c}.\,T\colon 1\to 0}\,\mathsf{zero} \qquad D_{\top,\top}\quad a\mapsto\top,b\mapsto\top \mid \vdash \boxed{c}.\,T\colon 1\to 1}{a\mapsto\top \mid \vdash \boxed{b}.\,\boxed{c}.\,T\colon 1\to\tfrac{1}{2}}\,\mathsf{gen}$$

$$D = \dfrac{D_{\bot}\quad a\mapsto\bot \mid \vdash \boxed{b}.\,\boxed{c}.\,T\colon 1\to\tfrac{1}{4} \qquad D_{\top}\quad a\mapsto\top \mid \vdash \boxed{b}.\,\boxed{c}.\,T\colon 1\to\tfrac{1}{2}}{\mid \vdash \boxed{a}.\,\boxed{b}.\,\boxed{c}.\,T\colon 1\to\tfrac{3}{8}}\,\mathsf{gen}$$

Figure 3 Typing derivations for Example 8, where $T = ((I\,c\,\Omega)\,b\,\Omega)\,a\,(\Omega\,b\,I)$.

Second, a key difference with the simple type system of Figure 1 is that probabilistic branching occurs in the *generator* rule, whereas for simple types it is the *choice* rule. This is due to aiming for *head normalization* instead of *strong normalization*. Consider the example term $\boxed{a}.\, M\, a\, (\Omega\, a\, N)$. Head reduction projects it to M and N, removing Ω, which will not be reduced. This is reflected in the probabilistic type system, where the branches of the generator typing rule project to M and N via the event valuation. The simple type system, reflecting strong normalization, would require also Ω to be typed – which of course it cannot.

For terms of the form $M \oplus N = \boxed{a}.\, M\, a\, N$ this difference is moot: both type systems branch similarly for this construct, to M and N. This leaves as only distinction the generalisation of types themselves, from a single base type o to probabilities p. In accordance with the meaning of a base type as the probability of termination, simple types are then recovered by restricting to base type $p = 1$. This rules out the rules zero and low, assigning a zero-weighted type $\overline{A} \to 0$ and reducing the weight of a type. It is easy to observe that the remaining rules preserve the restriction $p = 1$, to give the following proposition.

▶ **Proposition 9.** *For the fragment below left, the simply-typed* PEΛ *coincides with the probabilistically-typed* PEΛ *restricted to the types below right.*

$$M,\, N \;::=\; x \mid \lambda x.\, M \mid M\, N \mid \boxed{a}.\, M\, a\, N \qquad\qquad A,\, B \;::=\; 1 \mid A \to B$$

Our main result for the probabilistically-typed PEΛ is that a type $\overline{A} \to p$ guarantees head normalization with probability at least p. Formally, this is stated by a head reduction to a distribution of which a proportion of at least p is in head-normal form.

▶ **Theorem 10.** *For closed M, if $M : \overline{A} \to p$ then $M \twoheadrightarrow_{\mathsf{h}} \mathcal{N}_0 + \mathcal{N}_1$ where all terms in $\mathcal{N}_0$ are head normal and $|\mathcal{N}_0| \geq p$.*

The result will follow directly from the corresponding Theorem 22 for the expanded probabilistic calculus with sequential composition, introduced in Section 5.

4 Comparison with counting quantifiers

In this section we will give a close comparison with the type system C$\lambda_{\to}$ of Antonelli, Dal Lago, and Pistone [1], and demonstrate that our approach gives tighter bounds on the probability of termination.

The first distinction between C$\lambda_{\to}$ and PEΛ$^{\mathbb{Q}\to}$ is that the former uses indexed event variables x_a^i for $i \in \mathbb{N}$ instead of events a. However, there is no formal need for this; since the indices i are static, we may replace x_a^i and x_a^j for distinct i and j simply with distinct events a and b. This simplification extends to the semantics. Probabilities in C$\lambda_{\to}$ are given by boolean formulas $\mathscr{b}$ over the variables x_a^i, indicating a subset of the space $(2^{\mathbb{N}})^X$ where X is a finite set of events. However, since the indices i are fixed and bounded, it is sufficient to consider finite spaces 2^X. The elements of this set are the event valuations E with domain X, and a boolean formula $\mathscr{b}$ over X indicates the set of event valuations $[\![\mathscr{b}]\!]_X = \{E \in 2^X \mid E \models \mathscr{b}\}$, where $E \models \mathscr{b}$ is characterized syntactically as expected:

$$\frac{}{E, a \mapsto \top \;\models\; a} \qquad \frac{}{E, a \mapsto \bot \;\models\; \neg a} \qquad \frac{}{E \;\models\; \top} \qquad \frac{E \;\models\; \mathscr{b} \qquad E \;\models\; \mathscr{c}}{E \;\models\; \mathscr{b} \wedge \mathscr{c}} \qquad \frac{E \;\models\; \mathscr{b}}{E \;\models\; \mathscr{b} \vee \mathscr{c}}$$

For a set $\mathcal{E} \subseteq 2^X$ the measure $\mu_X(\mathcal{E})$ is given by

$$\mu_X(\mathcal{E}) = \frac{|\mathcal{E}|}{2^{|X|}}$$

where $|S|$ denotes the size of a set S. Then $\mu(\mathscr{b})$ is $\mu_X([\![\mathscr{b}]\!]_X)$ where X is the domain of $\mathscr{b}$. This coincides with the measure $\mu(\mathscr{b})$ over $(2^{\mathbb{N}})^X$ in [1], as the latter makes no essential use of the infinity offered by $\mathbb{N}$.

The second difference is the syntax of types: $\mathrm{C}\lambda_{\to}$ introduces probabilities through counting quantifiers C^p, where $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ has probabilities as base types. Types are nevertheless isomorphic: $\mathrm{C}\lambda_{\to}$ types $\mathfrak{s}$ are of the form $\mathrm{C}^p(\mathfrak{s}_1 \to \cdots \to \mathfrak{s}_n \to o)$, with a fixed outer counting quantifier, and map 1-to-1 onto $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ types A of the form $A_1 \to \ldots \to A_n \to p$ by associating the probability p instead with the consequent o. Formally, we encode types $\mathfrak{s}$ and typing contexts Γ of $\mathrm{C}\lambda_{\to}$ into our setting as follows.

$$\begin{aligned}[\![\mathrm{C}^q(o)]\!] &= q \\ [\![\mathrm{C}^q(\mathfrak{s} \Rightarrow \tau)]\!] &= [\![\mathfrak{s}]\!] \to [\![\mathrm{C}^q(\tau)]\!]\end{aligned} \qquad [\![x_1 : \mathfrak{s}_1, \ldots, x_n : \mathfrak{s}_n]\!] = x_1 : [\![\mathfrak{s}_1]\!], \ldots, x_n : [\![\mathfrak{s}_n]\!]$$

Having connected boolean formulas $\mathscr{b}$ to event valuations E, and $\mathrm{C}\lambda_{\to}$ types to $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ types, we may state the following conservativity result of $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ over $\mathrm{C}\lambda_{\to}$.

▶ **Proposition 11.** *If* $E \in [\![\mathscr{b}]\!]_X$ *then*

$$\Gamma \vdash^X M : \mathscr{b} \rightarrowtail \mathfrak{s} \quad \textit{implies} \quad E \mid [\![\Gamma]\!] \vdash M : [\![\mathfrak{s}]\!] \ .$$

Proof. By induction on the typing derivation for $\Gamma \vdash^X M : \mathscr{b} \rightarrowtail \mathfrak{s}$. ◀

▶ **Corollary 12.** *For a given closed term* M *the type system* $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ *gives the same or higher termination bounds than* $\mathrm{C}\lambda_{\to}$.

In the reverse direction, Example 8 and its counterpart in [1, Section 6.2] show that the two type systems do not give the exact same bounds, and in some cases $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ gives a strictly higher bound. The reason is that $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ locates branching between alternatives at the gen-rule for $\boxed{a}.\,M$, where $\mathrm{C}\lambda_{\to}$ branches for *choice* terms $M\,a\,N$ or in a contraction rule. Crucially, the gen-rule allows branches with different termination bounds. We illustrate this further by attempting to simulate the rule for $\boxed{a}.\,M$ in $\mathrm{C}\lambda_{\to}$.

$$\frac{E, a \mapsto \top \mid [\![\Gamma]\!] \vdash M : \overline{A} \to p \qquad E, a \mapsto \bot \mid [\![\Gamma]\!] \vdash M : \overline{A} \to q}{E \mid [\![\Gamma]\!] \vdash \boxed{a}.\,M : \overline{A} \to \tfrac{1}{2}p + \tfrac{1}{2}q}\ \mathsf{gen}$$

The branching at this rule in $\mathrm{C}\lambda_{\to}$ is captured with a contraction on the two premisses, which requires the probabilities to be equal, i.e. $p = q$. The derivation is as follows, where $\mathscr{d} = \top$ and $\mu(\mathscr{d}) = 1$ for the counting rule.

$$\frac{\dfrac{\Gamma \vdash^{X\cup\{a\}} M : \mathscr{b} \wedge a \rightarrowtail \mathrm{C}^p\sigma \qquad \Gamma \vdash^{X\cup\{a\}} M : \mathscr{b} \wedge \neg a \rightarrowtail \mathrm{C}^p\sigma \qquad \mathscr{b} \models (\mathscr{b} \wedge a) \vee (\mathscr{b} \wedge \neg a)}{\Gamma \vdash^{X\cup\{a\}} M : \mathscr{b} \rightarrowtail \mathrm{C}^p\sigma}}{\Gamma \vdash^X \boxed{a}.M : \mathscr{b} \rightarrowtail \mathrm{C}^p\sigma}$$

This is the issue illustrated by Example 8 and its counterpart in [1, Section 6.2], for which $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ gives the actual termination probability of $\frac{3}{8}$, while $\mathrm{C}\lambda_{\to}$ gives a best approximation of $\frac{1}{4}$. Reprising the example in $\mathrm{C}\lambda_{\to}$, the two sub-derivations for $\boxed{b}.\,\boxed{c}.\,T$, given in condensed form below, assign probabilities of $\frac{1}{4}$ and $\frac{1}{2}$. These may only be combined in a contraction by lowering the first probability to match the $\frac{1}{4}$ of the first.

$$\frac{\dfrac{\vdash^{\{a,b,c\}} T : a \wedge b \wedge c \rightarrowtail \mathrm{C}^1\sigma}{\vdash^{\{a,b\}} \boxed{c}.\,T : a \wedge b \rightarrowtail \mathrm{C}^{\frac{1}{2}}\sigma}}{\vdash^{\{a\}} \boxed{b}.\,\boxed{c}.\,T : a \rightarrowtail \mathrm{C}^{\frac{1}{4}}\sigma} \qquad \frac{\dfrac{\vdash^{\{a,b,c\}} T : \neg a \wedge \neg b \rightarrowtail \mathrm{C}^1\sigma}{\vdash^{\{a,b\}} \boxed{c}.\,T : \neg a \wedge \neg b \rightarrowtail \mathrm{C}^1\sigma}}{\vdash^{\{a\}} \boxed{b}.\,\boxed{c}.\,T : \neg a \rightarrowtail \mathrm{C}^{\frac{1}{2}}\sigma}$$

The above analysis suggests that this issue with $\mathrm{C}\lambda_{\to}$ may be fixed by adopting the generator typing rule of $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$, adjusted appropriately as follows. The key here is that different branches feature the dual atoms a and $\neg a$, not seen in either the simple type system $\mathsf{PE}\Lambda^{\mathbb{Q}\to}$ nor the intersection type system in [1].

$$\frac{\Gamma \vdash^{X\cup\{a\}} M : \mathscr{c} \wedge a \rightarrowtail \mathrm{C}^{p}\sigma \qquad \Gamma \vdash^{X\cup\{a\}} M : \mathscr{d} \wedge \neg a \rightarrowtail \mathrm{C}^{q}\sigma \qquad \mathscr{b} \models \mathscr{c} \vee \mathscr{d}}{\Gamma \vdash^{X} \boxed{a}.\, M : \mathscr{b} \rightarrowtail \mathrm{C}^{\frac{1}{2}p+\frac{1}{2}q}\sigma}$$

5 Sequencing

Two observations about probabilistic λ-calculi motivate the developments in the remainder of this paper. The first is the primary role of *head reduction*. It is well known that head normalization corresponds closely to evaluation on the Krivine Machine [23], and sometimes the machine gives a more natural model of what is being studied. The second is that probabilistic evaluation in λ-calculi needs to account for the difference between call–by–value (cbv) and call–by–name (cbn), to which end additional constructions are introduced, sometimes ad-hoc. The $\mathsf{PE}\Lambda$ is an example, as is the separate consideration of a cbv- and a cbn-probabilistic sum by Faggian and Ronchi Della Rocca [16], while Antonelli, Dal Lago, and Pistone [1] add to the standard cbn-application a second cbv-application.

These observations prompted us to consider probabilistic termination from the perspective of the *Functional Machine Calculus* (FMC) [18], a λ-calculus with computational effects. Firstly, the Krivine Machine plays a central role in the FMC (indeed it is the "M" in "FMC"), while the FMC provides the machine with a new notion of *successful termination*, absent from the standard λ-calculus. It is then a natural question if and how this may be used to capture the *probability* of successful termination. Secondly, the FMC may express both cbn and cbv behaviour, with the cbn λ-calculus a fragment and the cbv λ-calculus encoded in the syntax. The need for ad-hoc constructs to control reduction behaviour is thus avoided.

The Krivine Machine, simplified by replacing *environments* with *substitution*, evaluates a λ-term in the presence of a stack of input terms. An abstraction $\lambda x.\, M$ pops the top off the stack, say N, and continues as $\{N/x\}M$, while an application $M\, N$ pushes its argument N and continues as M. A λ-term may thus be viewed as a language of instruction sequences for this machine: application–*push*, abstraction–*pop*, variable–*execute*.

The FMC then extends the λ-calculus with *sequential composition* $M\,;N$ and its unit, the imperative *skip* $\star$, with the expected semantics: *concatenation* of machine instructions and the *empty* instruction. As in models of imperative languages, *skip* indicates successful termination of the machine. This gives a fragment called the *sequential λ-calculus*.

We adopt these modifications in the $\mathsf{PE}\Lambda$ to give the *sequential probabilistic event λ-calculus* ($\mathsf{SPE}\Lambda$), defined below. Following [18] we render abstraction as $\langle x\rangle.\, M = \lambda x.\, M$ and application as $[N].\, M = M\, N$ to emphasise the machine behaviour of *pop* and *push*, retaining the standard syntax as a shorthand. In particular, the new notation clarifies the interaction between *push* and *sequencing*: the following three terms are equivalent, rendered first in standard notation and second with prefix application.

$$(\star\, N)\,;M \;\sim\; (\star\,;M)\, N \;\sim\; M\, N \qquad\qquad ([N].\,\star)\,;M \;\sim\; [N].\,(\star\,;M) \;\sim\; [N].\,M$$

The full FMC further generalises to a machine with multiple independent stacks, addressed by a set of *locations*, in which *pop* and *push* are then parameterised to operate on the corresponding stack. This allows us to encode the effects of mutable higher-order store, input/output, and indeed probabilistic computation: the generator $\boxed{a}.\, M$ of the $\mathsf{PE}\Lambda$ is, in the FMC, an abstraction $\mathsf{rnd}\langle\, a\,\rangle.\, M$ parameterised to draw from a stream of random values labelled rnd. The $\mathsf{SPE}\Lambda$ is thus a fragment of the FMC with two locations.

▶ **Definition 13.** *The* sequential probabilistic event λ-calculus $\mathsf{SPE}\Lambda$ *is given as follows.*

$$M, N, P \ ::= \ x \ \mid \ \langle x \rangle . M \ \mid \ [N] . M \ \mid \ \boxed{a} . M \ \mid \ M \, a \, N \ \mid \ \star \ \mid \ N \, ; M$$

Prefixing binds tighter than sequencing, $[N] . M \, ; P = ([N] . M) \, ; P$, and sequencing associates right, $M \, ; N \, ; P = M \, ; (N \, ; P)$. *Projections* and *contexts* extend to the $\mathsf{SPE}\Lambda$ as below; *head contexts* and *applicative contexts* are as for the $\mathsf{PE}\Lambda$.

$$\pi_i^a \star \ = \ \star \qquad \pi_i^a (N \, ; M) \ = \ \pi_i^a N \, ; \pi_i^a M \qquad C \ ::= \ \ldots \ \mid \ C \, ; M \ \mid \ M \, ; C$$

The interaction between sequentiality and the λ-calculus is governed by the following *sequencing* reduction rules. These make sequential composition right-associative, and let the *prefixing* of *push*, *pop*, and the generator propagate past it (as in a standard list concatenation algorithm). The result is to make the first such action on the abstract machine the leading construct.

$$\star \, ; P \ \to\!\!\!\!\!\rightarrow_\sigma \ P \qquad \begin{array}{ll} (N \, ; M) \, ; P \to\!\!\!\!\!\rightarrow_\sigma \ N \, ; (M \, ; P) & \langle x \rangle . M \ ; \ P \to\!\!\!\!\!\rightarrow_\sigma \ \langle x \rangle . (M \, ; P) \quad (x \notin \mathsf{fv}(P)) \\ [N] . M \ ; \ P \to\!\!\!\!\!\rightarrow_\sigma \ [N] . (M \, ; P) & \boxed{a} . M \ ; \ P \to\!\!\!\!\!\rightarrow_\sigma \ \boxed{a} . (M \, ; P) \quad (a \notin \mathsf{fe}(P)) \end{array}$$

The *sequencing* relation $\to\!\!\!\!\!\rightarrow_\sigma$ is given by closing these rules under all contexts C, and *head sequencing* $\to\!\!\!\!\!\rightarrow_{\sigma\mathsf{h}}$ by closing under head contexts H only. The β-reduction rule in $\mathsf{SPE}\Lambda$ notation is as below left, with $\to\!\!\!\!\!\rightarrow_\beta$ given by closing under all contexts and $\to\!\!\!\!\!\rightarrow_{\beta\mathsf{h}}$ by closing under head contexts. Projective reduction, below right, is as previously.

$$[N] . \langle x \rangle . M \ \to\!\!\!\!\!\rightarrow_\beta \ \{N/x\} M \qquad H\{\boxed{a} . M\} \ \to\!\!\!\!\!\rightarrow_\pi \ [\, \tfrac{1}{2} \cdot H\{\pi_\top^a M\} \, , \, \tfrac{1}{2} \cdot H\{\pi_\top^a M\} \,]$$

Head reduction $\to\!\!\!\!\!\rightarrow_\mathsf{h}$ is the union of all three head relations:

$$\to\!\!\!\!\!\rightarrow_\mathsf{h} \ = \ \to\!\!\!\!\!\rightarrow_{\beta\mathsf{h}} \cup \to\!\!\!\!\!\rightarrow_{\sigma\mathsf{h}} \cup \to\!\!\!\!\!\rightarrow_\pi \ .$$

We round off by observing the shape of head-normal forms, assuming no free event variables.

▶ **Proposition 14.** *The head-normal forms of event-closed* $\mathsf{SPE}\Lambda$*-terms are of one of the three forms* $H\{\star\}$*,* $H\{x\}$*, and* $H\{x \, ; M\}$*.*

5.1 Encoding call–by–value

Sequential composition provides an essential element that the λ-calculus lacks, and which is at the heart of the cbv/cbn dichotomy: *control over execution*. The cbv behaviour of an application $M \, N$ is encoded almost as $N \, ; M$: *first* evaluate the argument N, *then* evaluate the function M (the full encoding, below, includes an extra part to also *execute* M).

We demonstrate the encoding of the cbv-probabilistic λ-calculus $\Lambda_\oplus^{\mathsf{cbv}}$ of Faggian and Ronchi Della Rocca [16]. The encoding of cbv λ-terms is standard: see [13, 18, 28]. *Values* V, W and *terms* M, N encode by the translations $-_\mathsf{v}$ and $-_\mathsf{t}$ respectively, below.

$$\begin{array}{lrlrl} \text{Values } V, W: & x_\mathsf{v} & = x & \text{Terms } M, N: & V_\mathsf{t} = [V_\mathsf{v}] . \star \\ & (\lambda x.M)_\mathsf{v} & = \langle x \rangle . M_\mathsf{t} & & (M \, N)_\mathsf{t} = N_\mathsf{t} \, ; M_\mathsf{t} \, ; \langle x \rangle . x \\ & & & & (M \oplus N)_\mathsf{t} = \boxed{a} . M_\mathsf{t} \, a \, N_\mathsf{t} \end{array}$$

The operational intuition is that a *push* represents a *return value*: a term M_t evaluates until it is of the form $V_\mathsf{t} = [V_\mathsf{v}] . \star$, at which point V_v is pushed to the stack and the machine terminates. Then β-reduction is simulated as follows.

$$\begin{array}{rcl}((\lambda x.M)\,V)_{\mathsf{t}} & = & [V_{\mathsf{v}}].\ \star\ ;\ [\langle x\rangle.\,M_{\mathsf{t}}].\ \star\ ;\ \langle y\rangle.\,y \\ & \twoheadrightarrow_{\sigma} & [V_{\mathsf{v}}].\,[\langle x\rangle.\,M_{\mathsf{t}}].\,\langle y\rangle.\,y \\ & \to_{\beta} & [V_{\mathsf{v}}].\,\langle x\rangle.\,M_{\mathsf{t}} \\ & \to_{\beta} & \{V_{\mathsf{v}}/x\}M_{\mathsf{t}} \\ & = & (\{V/x\}M)_{\mathsf{t}}\end{array}$$

The probabilistic reduction rule of $\Lambda^{\mathsf{cbv}}_{\oplus}$ is that of projective reduction, under the given encoding, but it applies in *surface contexts*:

$$S \ ::= \ \{\} \ \mid \ M\,S \ \mid \ S\,M$$

Then the translation of $S\{M \oplus N\}$ indeed does not place the probabilistic redex inside a *push*, the requirement for correct behaviour.

5.2 The abstract machine

The small-step operational semantics of the SPEΛ is given by the following abstract machine. A *state* is a triple (S, M, K), where M is a term and S and K are stacks of terms. S is the *operand stack*, with the head to the right as $S\,N$, and K is the *continuation stack*, with the head to the left as $N\,K$. In both cases the empty stack is written ε, and concatenation by juxtaposition, $S\,T$. *Transitions* or *steps* are probabilistic: a transition rule is written as below left, expressing that the machine transitions from a state (S, M, K) to (T, N, L) with probability p. We may omit p when $p = 1$. A *run* is a sequence of steps, written as below centre, where probabilities are multiplied, i.e. p below is the product of the probabilities of all steps. A run is *successful* if it terminates with *skip* and an empty continuation stack, as below right; the stack T then holds the *return values* of the computation.

$$\text{step: } p\frac{(\,S\ ,\ M\ ,\ K\,)}{(\,T\ ,\ N\ ,\ L\,)} \qquad \text{run: } p\frac{(\,S\ ,\ M\ ,\ K\,)}{(\,T\ ,\ N\ ,\ L\,)} \qquad \text{successful run: } p\frac{(\,S\ ,\ M\ ,\ K\,)}{(\,T\ ,\ \star\ ,\ \varepsilon\,)}$$

▶ **Definition 15.** *The* sequential probabilistic machine *(SPM) is given by the following probabilistic* transitions*.*

$$\frac{(\,S\ ,\ [N].\,M\ ,\ K\,)}{(\,S\,N\ ,\ M\ ,\ K\,)} \qquad \frac{(\,S\ ,\ N\,;M\ ,\ K\,)}{(\,S\ ,\ N\ ,\ M\,K\,)} \qquad \frac{1}{2}\frac{(\,S\ ,\ \boxed{a}.\,M\ ,\ K\,)}{(\,S\ ,\ \pi^a_{\top}M\ ,\ K\,)}$$

$$\frac{(\,S\,N\ ,\ \langle x\rangle.\,M\ ,\ K\,)}{(\,S\ ,\ \{N/x\}M\ ,\ K\,)} \qquad \frac{(\,S\ ,\ \star\ ,\ M\,K\,)}{(\,S\ ,\ M\ ,\ K\,)} \qquad \frac{1}{2}\frac{(\,S\ ,\ \boxed{a}.\,M\ ,\ K\,)}{(\,S\ ,\ \pi^a_{\bot}M\ ,\ K\,)}$$

5.3 Big-step semantics

Running the machine for a given term and input stack gives a distribution of return stacks. We use the following notation, extending from that for distributions over terms.

$$\mathcal{T} \ = \ [p_1\cdot T_1, \ldots, p_n\cdot T_n] \ = \ [p_i\cdot T_i]_{i\le n}$$

▶ **Definition 16.** *The* evaluation *relation $S, M \Downarrow \mathcal{T}$ is defined inductively by the following rules.*

$$\frac{}{S, \star \Downarrow [1\cdot S]} \qquad \frac{S, \{N/x\}M \Downarrow \mathcal{T}}{S\,N, \langle x\rangle.\,M \Downarrow \mathcal{T}} \qquad \frac{R, M \Downarrow [p_i\cdot S_i]_{i\le n} \qquad (\,S_i, N \Downarrow \mathcal{T}_i\,)_{i\le n}}{R, M\,;N \Downarrow \sum_{i\le n} p_i\mathcal{T}_i}$$

$$\frac{}{S, M \Downarrow \varnothing} \qquad \frac{S\,N, M \Downarrow \mathcal{T}}{S, [N].\,M \Downarrow \mathcal{T}} \qquad \frac{S, \pi^a_{\top}M \Downarrow \mathcal{T}_{\top} \qquad S, \pi^a_{\bot}M \Downarrow \mathcal{T}_{\bot}}{S, \boxed{a}.\,M \Downarrow \frac{1}{2}\mathcal{T}_{\top} + \frac{1}{2}\mathcal{T}_{\bot}}$$

We demonstrate that small-step and big-step semantics agree.

▶ **Proposition 17.** *$S, M \Downarrow \mathcal{T}$ if and only if there is a finite collection of n distinct runs*

$$p_i \frac{(S\ ,\ M\ ,\ \varepsilon)}{(T_i\ ,\ \star\ ,\ \varepsilon)} \quad (i \leq n) \quad \textit{such that} \quad \mathcal{T} = [p_i \cdot T_i]_{i \leq n}\ .$$

Proof. ($\Longrightarrow$) By induction on $S, M \Downarrow \mathcal{T}$. ($\Longleftarrow$) By induction each run in the collection of n runs. ◀

6 Sequential probabilistic types

Types for the sequential λ-calculus are of the form below, with the meaning: given an input stack of terms typed by A_1 through A_n, the machine will terminate successfully and return a stack with types B_1 through B_m.

$$A_n \dots A_1 \Rightarrow B_1 \dots B_m$$

For the SPEΛ, we parameterize this with the *probability* of successful termination.

▶ **Definition 18.** Sequential probabilistic types *are given by the following grammars, where p is a probability.*

$$\begin{array}{rcll} A, B, C & ::= & \overline{A} \overset{p}{\Rightarrow} \overline{C} & \textit{(types)} \\ \overline{A} & ::= & A_1 \dots A_n & \textit{(type vectors)} \end{array}$$

A typing judgement $E \mid \Gamma \vdash M \colon A$ *assigns a term M the type A in the context of E and Γ, and $E \mid \Gamma \vdash S \colon \overline{A}$ assigns a stack of terms S a type vector $\overline{A}$. The sequential probabilistic type system* $\mathsf{SPE}\Lambda^{\mathbb{Q}\Rightarrow}$ *is given by the typing rules in Figure 4. We may omit p when $p = 1$.*

There are no base types: their rôle is subsumed by types with empty vectors ($\overset{p}{\Rightarrow}$). Observe that because stacks are last-in-first-out, the identity term on two elements is $\langle x \rangle . \langle y \rangle . [y] . [x] . \star$, i.e. with the order of x and y reversed between popping and pushing. We match this reversal in types, and assign this term the type $AB \Rightarrow BA$. Since we want identity types to be of the form $\overline{A} \Rightarrow \overline{A}$, in a type $\overline{A} \overset{p}{\Rightarrow} \overline{C}$ we consider the antecedent type vector $\overline{A}$ to be reversed, i.e.

$$\overline{A} \Rightarrow \overline{A} \quad = \quad A_n \dots A_1 \Rightarrow A_1 \dots A_n\ .$$

Probabilistic PEΛ-types embed into sequential types by $\overline{A} \to p \quad = \quad \overline{A} \overset{p}{\Rightarrow} \varepsilon$. With this identification, for PEΛ-terms and -types the two type systems coincide.

▶ **Proposition 19.** *For M a* PEΛ*-term, $E \mid \Gamma \vdash M \colon \overline{A} \to p$ if and only if $E \mid \Gamma \vdash M \colon \overline{A} \overset{p}{\Rightarrow} \varepsilon$.*

Every type is inhabited by a closed term. For a type A, define the *zero term* 0_A as follows: for $A = B_1 \dots B_m \overset{p}{\Rightarrow} C_1 \dots C_n$,

$$0_A \ = \ \langle x_1 \rangle \dots \langle x_m \rangle . [0_{C_1}] \dots [0_{C_n}] . \star\ .$$

▶ **Proposition 20** (Type inhabitation)**.** *Every type A is inhabited by its zero term, $\vdash 0_A \colon A$.*

Proof. By induction on the type A, using the low rule to lower the termination probability from 1 to an arbitrary p. ◀

$$\frac{}{E \mid \Gamma,\, x\colon \overline{A} \overset{p}{\Rightarrow} \overline{C} \vdash x\colon \overline{A}\,\overline{B} \overset{p}{\Rightarrow} \overline{B}\,\overline{C}}\ \mathsf{var} \qquad \frac{}{E \mid \Gamma \vdash M\colon \overline{A} \overset{0}{\Rightarrow} \overline{C}}\ \mathsf{zero}$$

$$\frac{E \mid \Gamma,\, x\colon A \vdash M\colon \overline{B} \overset{p}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash \langle x\rangle.\, M\colon A\,\overline{B} \overset{p}{\Rightarrow} \overline{C}}\ \mathsf{abs} \qquad \frac{E \mid \Gamma \vdash N\colon A \qquad E \mid \Gamma \vdash M\colon A\,\overline{B} \overset{p}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash [N].\, M\colon \overline{B} \overset{p}{\Rightarrow} \overline{C}}\ \mathsf{app}$$

$$\frac{E, a{\mapsto} i \mid \Gamma \vdash M_i\colon A}{E, a{\mapsto} i \mid \Gamma \vdash M_\top\, a\, M_\bot\colon A}\ \mathsf{chc} \qquad \frac{E, a{\mapsto}\top \mid \Gamma \vdash M\colon \overline{A} \overset{p}{\Rightarrow} \overline{C} \qquad E, a{\mapsto}\bot \mid \Gamma \vdash M\colon \overline{A} \overset{q}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash \boxed{a}.\, M\colon \overline{A} \overset{\frac{1}{2}p+\frac{1}{2}q}{\Rightarrow} \overline{C}}\ \mathsf{gen}$$

$$\frac{}{E \mid \Gamma \vdash \star\colon \overline{A} \overset{1}{\Rightarrow} \overline{A}}\ \mathsf{skip} \qquad \frac{E \mid \Gamma \vdash M\colon \overline{A} \overset{p}{\Rightarrow} \overline{B} \qquad E \mid \Gamma \vdash N\colon \overline{B} \overset{q}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash M\,;N\colon \overline{A} \overset{pq}{\Rightarrow} \overline{C}}\ \mathsf{seq}$$

$$\frac{E \mid \Gamma \vdash M\colon \overline{A} \overset{q}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash M\colon \overline{A} \overset{p}{\Rightarrow} \overline{C}}\ \mathsf{low}\ (p{<}q) \qquad \frac{(E \mid \Gamma \vdash M_i\colon A_i)_{\,i\le n}}{E \mid \Gamma \vdash \varepsilon\, M_1 \ldots M_n\colon A_1 \ldots A_n}\ \mathsf{stk}$$

Figure 4 The sequential probabilistic type system $\mathsf{SPE}\Lambda^{\mathbb{Q}\Rightarrow}$.

Type inhabitation is an important justification for the interpretation of types as a guarantee for successful machine termination: it means that for a typed term $M\colon A \overset{p}{\Rightarrow} \overline{C}$, a suitable input stack $S : \overline{A}$ always exists. Our main theorem, below, formalizes this interpretation: for a term $M\colon A \overset{p}{\Rightarrow} \overline{C}$ and stack $S : \overline{A}$, the machine successfully returns a stack $T : \overline{C}$ with probability at least p.

▶ **Theorem 21.** *For a typed, closed term $M\colon \overline{A} \overset{p}{\Rightarrow} \overline{C}$ and stack $S : \overline{A}$ there is a finite set of distinct successful runs*

$$p_i \frac{(S\ ,\ M\ ,\ \varepsilon)}{(T_i\ ,\ \star\ ,\ \varepsilon)} \qquad (i \le n)$$

with sum probability $\sum_{i\le n} p_i \ge p$.

Proof. See Appendix A.1. ◀

The lower bound given by the theorem is an exact bound when two conditions are met: the typing derivation does not use the low rule to lower the termination bound, and every use of the zero rule to assign a zero probability applies to a term that is in fact non-terminating. In such a case, such as Example 8, types can be seen to give an exact bound.

Since head reduction as a strategy closely follows machine evaluation, it is no surprise that the termination bound for the machine is also a lower bound for probabilistic head reduction. This is formalised in the following theorem.

▶ **Theorem 22.** *For a closed M, if $M\colon \overline{A} \overset{p}{\Rightarrow} \overline{C}$ then $M \twoheadrightarrow_{\mathsf{h}} \mathcal{N}_0 + \mathcal{N}_1$ where all terms in $\mathcal{N}_0$ are head normal and $|\mathcal{N}_0| \ge p$.*

Proof. See Appendix A.2. ◀

Observe that our main theorem on probabilistic termination for the PEΛ, Theorem 10, follows immediately from the corresponding Theorem 22 above by conservativity of the SPEΛ over the PEΛ. This is despite the fact that the proof of Theorem 22 makes essential use of the notion of successful termination made available by sequencing, absent from the PEΛ.

7 Conclusions

To us, what stands out about our approach is the simplicity and the natural intuition of our type system. This manifests in the transparent reasoning in our proofs, which despite using deep techniques such as abstract reducibility, give a direct and clear connection between types, machine behaviour, and reduction.

Compared with the approach in [1], the simplicity of our approach manifests in several advantages. First is to avoid the need for *counting quantifiers*, associating probabilities instead with base types, and the use of simple event valuations over boolean formulas. Second, it is clear that the expression of both call–by–name and call–by–value behaviour is an essential ingredient for a probabilistic calculus. We eschew the introduction of ad-hoc constructs, relying instead on a principled interpretation of `cbv` via sequential composition. Finally, where the main example [1, Example 6.2] requires intersection types to produce the exact bound, our approach does so directly in Example 8, while morally remaining within the realm of simple types, and strictly improving on the bounds provided by $\mathrm{C}\lambda_{\to}$.

References

1 Melissa Antonelli, Ugo Dal Lago, and Paolo Pistone. Curry and howard meet borel. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'22)*, pages 45:1–45:13. ACM, 2022. doi:10.1145/3531130.3533361.

2 Martin Avanzini, Ugo Dal Lago, and Akihisa Yamada. On probabilistic term rewriting. In John P. Gallagher and Martin Sulzmann, editors, *Proc. Functional and Logic Programming - 14th International Symposium, FLOPS 2018*, volume 10818 of *Lecture Notes in Computer Science*, pages 132–148. Springer, 2018. doi:10.1007/978-3-319-90686-7_9.

3 Chris Barrett, Willem Heijltjes, and Guy McCusker. The Functional Machine Calculus II: Semantics. In *31st EACSL Annual Conference on Computer Science Logic (CSL 2023)*, volume 252 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:18, 2023. doi:10.4230/LIPIcs.CSL.2023.10.

4 Raven Beutner and Luke Ong. On probabilistic termination of functional programs with continuous distributions. In Stephen N. Freund and Eran Yahav, editors, *Proceedings 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI)*, pages 1312–1326. ACM, 2021. doi:10.1145/3453483.3454111.

5 Flavien Breuvart and Ugo Dal Lago. On intersection types and probabilistic lambda calculi. In *roceedings of the 20th International Symposium on Principles and Practice of Declarative Programming, PPDP 2018*, pages 8:1–8:13. ACM, 2018. doi:10.1145/3236950.3236968.

6 Fredrik Dahlqvist and Dexter Kozen. Semantics of higher-order probabilistic programs with conditioning. *Proceedings of the ACM on Programming Languages*, 4(POPL):57:1–57:29, 2020. doi:10.1145/3371125.

7 Ugo Dal Lago, Claudia Faggian, and Simona Ronchi Della Rocca. Intersection types and (positive) almost-sure termination. *Proceedings of the ACM on Programming Languages*, 5(POPL):1–32, 2021. doi:10.1145/3434313.

8 Ugo Dal Lago and Charles Grellois. Probabilistic termination by monadic affine sized typing. *ACM Transactions on Programming Languages and Systems*, 41(2):10:1–10:65, 2019. doi:10.1145/3293605.

9 Ugo Dal Lago, Giulio Guerrieri, and Willem Heijltjes. Decomposing probabilistic lambda-calculi. In *Proceedings of Foundations of Software Science and Computation Structures (FoSSaCS)*, volume 12077 of *LNCS*, pages 136–156, 2020. doi:10.1007/978-3-030-45231-5_8.

10 Ugo Dal Lago and Margherita Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO - Theoretical Informatics and Applications*, 46(3):413–450, 2012. doi:10.1051/ita/2012012.

11 Karel de Leeuw, Edward F. Moore, Claude E. Shannon, and Norman Shapiro. Computability by probabilistic machines. *Automata studies*, 34:183–198, 1956.

12 Ugo De'Liguoro and Adolfo Piperno. Non deterministic extensions of untyped lambda-calculus. *Information and Computation*, 122(2):149–177, 1995. doi:10.1006/inco.1995.1145.

13 Rémi Douence and Pascal Fradet. A systematic study of functional language implementations. *ACM Transactions on Programming Languages and Systems*, 20(2):344–387, 1998. doi:10.1145/276393.276397.

14 Thomas Ehrhard and Giulio Guerrieri. The bang calculus: An untyped lambda-calculus generalizing call-by-name and call-by-value. In *Proceedings of the 18th International Symposium on Principles and Practice of Declarative Programming (PPDP'16)*, pages 174–187, 2016. doi:10.1145/2967973.2968608.

15 Thomas Ehrhard and Christine Tasson. Probabilistic call by push value. *Logical Methods in Computer Science*, 15(1), 2019. doi:10.23638/LMCS-15(1:3)2019.

16 Claudia Faggian and Simona Ronchi Della Rocca. Lambda calculus and probabilistic computation. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019*, pages 1–13, 2019. doi:10.1109/LICS.2019.8785699.

17 Jean Goubault-Larrecq. A probabilistic and non-deterministic call-by-push-value language. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019*, pages 1–13. IEEE Computer Society, 2019. doi:10.1109/LICS.2019.8785809.

18 Willem Heijltjes. The functional machine calculus. In *Proceedings of the 38th Conference on the Mathematical Foundations of Programming Semantics, MFPS XXXVIII*, volume 1 of *ENTICS*, 2022. doi:10.46298/ENTICS.10513.

19 C. Jones and G. D. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89)*, pages 186–195. IEEE Computer Society, 1989. doi:10.1109/LICS.1989.39173.

20 Achim Jung and Regina Tix. The troublesome probabilistic powerdomain. *Electronic Notes in Theoretical Computer Science*, 13:70–91, 1998. doi:10.1016/S1571-0661(05)80216-6.

21 Benjamin Kaminski. *Advanced Weakest Precondition Calculi for Probabilistic Programs*. PhD thesis, Fakultät für Mathematik, Informatik und Naturwissenschaften, RWTH Aachen University, 2019. doi:10.18154/RWTH-2019-01829.

22 Naoki Kobayashi, Ugo Dal Lago, and Charles Grellois. On the termination problem for probabilistic higher-order recursive programs. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019*, pages 1–14. IEEE, 2019. doi:10.1109/LICS.2019.8785679.

23 Jean-Louis Krivine. A call-by-name lambda-calculus machine. *Higher-Order and Symbolic Computation*, 20(3):199–207, 2007. doi:10.1007/s10990-007-9018-9.

24 Thomas Leventis. A deterministic rewrite system for the probabilistic λ-calculus. *Mathematical Structures in Computer Science*, 29(10):1479–1512, 2019. doi:10.1017/S0960129519000045.

25 Paul Blain Levy. *Call-by-push-value: A functional/imperative synthesis*, volume 2 of *Semantic Structures in Computation*. Springer Netherlands, 2003. doi:10.1007/978-94-007-0954-6.

26 Udi Manber and Martin Tompa. Probabilistic, nondeterministic, and alternating decision trees. In *14th Annual ACM Symposium on Theory of Computing*, pages 234–244, 1982. doi:10.1145/800070.802197.

27 Annabelle McIver and Carroll Morgan. Abstraction and refinement in probabilistic systems. *SIGMETRICS Perform. Eval. Rev.*, 32(4):41–47, March 2005. doi:10.1145/1059816.1059824.

28 A.J. Power and Hayo Thielecke. Closed Freyd- and κ-categories. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 1644 of *Lecture Notes in Computer Science*, pages 625–634. Springer, 1999. doi:10.1007/3-540-48523-6_59.

29 Norman Ramsey and Avi Pfeffer. Stochastic lambda calculus and monads of probability distributions. In *Conference Record of POPL 2002: The 29th SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '02, pages 154–165, 2002. doi:10.1145/503272.503288.

30 Nasser Saheb-Djahromi. Probabilistic LCF. In *Mathematical Foundations of Computer Science 1978, Proceedings, 7th Symposium*, volume 64 of *Lecture Notes in Computer Science*, pages 442–451. Springer, 1978. doi:10.1007/3-540-08921-7_92.

31 Davide Sangiorgi and Valeria Vignudelli. Environmental bisimulations for probabilistic higher-order languages. In *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016*, pages 595–607, 2016. doi:10.1145/2837614.2837651.

A Proofs for Section 6

A.1 Probabilistic machine termination

Our main theorem for the typed SPEΛ will be that a type $\overline{A} \overset{p}{\Rightarrow} \overline{B}$ guarantees a probability of termination of the machine of at least p, given an input stack of type $\overline{A}$, and returning a stack of type $\overline{B}$. The proof is a direct application of abstract reducibility. For each type we define a set $\textsc{run}(A)$ that holds the terms with the above property, and we proceed to prove that every term of type A belongs to this set.

We write $\mathcal{D}_p(X)$ for the set of finite distributions $\mathcal{X}$ of weight $|\mathcal{X}| \geq p$ over a set X.

▶ **Definition 23.** *The set* $\textsc{run}(-)$ *is defined by mutual induction on types* A *and type vectors* $\overline{A}$ *as a set of closed terms respectively of closed stacks, as follows.*

$$\textsc{run}(\overline{A} \overset{p}{\Rightarrow} \overline{B}) = \{\ M \mid \forall S \in \textsc{run}(\overline{A}).\, \exists \mathcal{T} \in \mathcal{D}_p(\textsc{run}(\overline{B}))\,.\ S, M \Downarrow \mathcal{T}\ \}$$
$$\textsc{run}(A_1 \dots A_n) = \{\ \varepsilon\, M_1 \dots M_n \mid M_i \in \textsc{run}(A_i)\ \}$$

For the proof of Lemma 25, the main reducibility lemma, we need the following notation, as well as an additional lemma. We write σ for a substitution map $\{M_i/x_i\}_{i\leq n}$, where σM is the application of σ to M. The set $\textsc{run}(-)$ then extends to contexts Γ as follows. Note that the definition implies that σ is closed (i.e. each substituting term is closed).

$$\textsc{run}(x_1 \colon A_1, \dots, x_n \colon A_n) = \{\ \sigma \mid \sigma x_i \in \textsc{run}(A_i)\ ,\ 1 \leq i \leq n\ \}$$

For an event valuation $E = (a_k \mapsto i_k)_{k\leq n}$ we write $\pi_E M$ for the projection on each a_i in E.

$$\pi_E M = \pi^{a_1}_{i_1}(\dots(\pi^{a_n}_{i_n} M)\dots)$$

We *expand* the stacks in a distribution $\mathcal{T}$ by prepending a stack S as $S\,\mathcal{T}$, where $S\,[p_i \cdot T_i]_{i\leq n} = [p_i \cdot S\,T_i]_{i\leq n}$.

▶ **Lemma 24.** *If* $S, M \Downarrow \mathcal{T}$ *then* $R\,S, M \Downarrow R\,\mathcal{T}$ *for any stack* R.

Proof. By induction on the definition of $\Downarrow$. ◀

The main reducibility lemma is then the following.

▶ **Lemma 25.** *If* $E \mid \Gamma \vdash M \colon A$ *and* $\sigma \in \textsc{run}(\Gamma)$ *then* $\pi_E(\sigma M) \in \textsc{run}(A)$.

Proof. By induction on the typing derivation for M. Note that since σ is closed, $\pi_E(\sigma M) = \sigma(\pi_E(M))$. We cover three key cases (sequencing, abstraction, and generator); the remaning are similar.

Sequencing case:

$$\frac{E \mid \Gamma \vdash N \colon \overline{A} \overset{p}{\Rightarrow} \overline{B} \quad E \mid \Gamma \vdash M \colon \overline{B} \overset{q}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash N\,; M \colon \overline{A} \overset{pq}{\Rightarrow} \overline{C}}\ \textsf{seq}$$

Let $N' = \pi_E(\sigma N)$ and $M' = \pi_E(\sigma M)$. Given $R \in \textsc{run}(\overline{A})$, by induction we have $R, N' \Downarrow [p_i \cdot S_i]_{i \le n}$ with each $S_i \in \textsc{run}(\overline{B})$ and $\sum_{i \le n} p_i \ge p$. Again by induction, for each $i \le n$ we have $S_i, M' \Downarrow \mathcal{T}_i$ with $\mathcal{T}_i \in \mathcal{D}_q(\textsc{run}(\overline{C}))$. The definition of $\Downarrow$ gives $R, (N' ; M') \Downarrow \mathcal{T}$ for $\mathcal{T} = \sum_{i \le n} p_i \mathcal{T}_i$. Finally, observe that $\mathcal{T} \in \mathcal{D}_{pq}(\textsc{run}(\overline{C}))$ since $|\mathcal{T}| = \sum_{i \le n} p_i |\mathcal{T}_i| \ge pq$ and since each $\mathcal{T}_i$ is a distribution over $\textsc{run}(\overline{C})$.

Abstraction case:

$$\frac{E \mid \Gamma, x \colon A \vdash M \colon \overline{B} \overset{p}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash \langle x \rangle . M \colon A\,\overline{B} \overset{p}{\Rightarrow} \overline{C}}\,\mathsf{abs}$$

Let $\sigma \in \textsc{run}(\Gamma)$ and $S\,N \in \textsc{run}(\overline{B}\,A)$. We need to demonstrate a distribution $\mathcal{T} \in \mathcal{D}_p(\textsc{run}(\overline{C}))$ such that $S\,N, \pi_E(\sigma(\langle x \rangle . M)) \Downarrow \mathcal{T}$. Let $M' = \pi_E(\sigma M)$ and note that since σ is not defined on x and σ and N are closed, $\pi_E(\sigma\{N/x\}M) = \{N/x\}M'$ and $\pi_E(\sigma(\langle x \rangle . M) = \langle x \rangle . M'$. The inductive hypothesis for the premise $E \mid \Gamma, x \colon A \vdash M \colon \overline{B} \overset{p}{\Rightarrow} \overline{C}$, with $\sigma\{N/x\} \in \textsc{run}(\Gamma, x \colon A)$, gives the desired $\mathcal{T} \in \mathcal{D}_p(\textsc{run}(\overline{C}))$ with evaluation $S, \{N/x\}M' \Downarrow \mathcal{T}$. Then by the definition of evaluation, $S\,N, M' \Downarrow \mathcal{T}$, and hence $\langle x \rangle . M' \in \textsc{run}(A\,\overline{B} \overset{p}{\Rightarrow} \overline{C})$.

Generator case:

$$\frac{E, a \mapsto \top \mid \Gamma \vdash M \colon \overline{A} \overset{p}{\Rightarrow} \overline{C} \qquad E, a \mapsto \bot \mid \Gamma \vdash M \colon \overline{A} \overset{q}{\Rightarrow} \overline{C}}{E \mid \Gamma \vdash \boxed{a} . M \colon \overline{A} \overset{\frac{1}{2}p + \frac{1}{2}q}{\Rightarrow} \overline{C}}\,\mathsf{gen}$$

Let $\sigma \in \textsc{run}(\Gamma)$ and $S \in \textsc{run}(\overline{A})$. Let $M_i = \pi_{E, a \mapsto i}(\sigma M)$ for $i \in \top, \bot$. By the inductive hypothesis, $M_\top \in \textsc{run}(\overline{A} \overset{p}{\Rightarrow} \overline{C})$ and $M_\bot \in \textsc{run}(\overline{A} \overset{q}{\Rightarrow} \overline{C})$, which gives $S, M_\top \Downarrow \mathcal{T}_\top$ and $S, M_\bot \Downarrow \mathcal{T}_\bot$ for some $\mathcal{T}_\top \in \mathcal{D}_p(\textsc{run}(\overline{C}))$ and $\mathcal{T}_\bot \in \mathcal{D}_q(\textsc{run}(\overline{C}))$. Let $r = \frac{1}{2}(p + q)$; then $\mathcal{T} = \frac{1}{2}(\mathcal{T}_\top + \mathcal{T}_\bot) \in \mathcal{D}_r(\textsc{run}(\overline{C}))$ and by definition of evaluation, $S, \pi_E(\sigma(\boxed{a} . M)) \Downarrow \mathcal{T}$. We conclude that $\pi_E(\sigma(\boxed{a} . M)) \in \textsc{run}(\overline{A} \overset{r}{\Rightarrow} \overline{C})$. ◀

Our main theorem, the probability of machine termination, is a direct consequence.

▶ **Theorem 21 (restatement).** *For a typed, closed term $M \colon \overline{A} \overset{p}{\Rightarrow} \overline{C}$ and stack $S : \overline{A}$ there is a finite set of distinct successful runs*

$$p_i \frac{(S\ ,\ M\ ,\ \varepsilon)}{(T_i\ ,\ \star\ ,\ \varepsilon)} \qquad (i \le n)$$

with sum probability $\sum_{i \le n} p_i \ge p$.

Proof. By Lemma 25 we have $M \in \textsc{run}(\overline{A} \overset{p}{\Rightarrow} \overline{C})$ and $S \in \textsc{run}(\overline{A})$. By the definition of $\textsc{run}(-)$ we then have $S, M \Downarrow \mathcal{T}$ with $|\mathcal{T}| \ge p$. Proposition 17 then gives the desired set of machine runs. ◀

A.2 Probabilistic head normalization

Finally, we will relate machine termination to head reduction. For a redex in a head context $\lambda x_1 \ldots \lambda x_n . \{\}\, M_1 \ldots M_m$ the machine runs as follows: after the abstractions consume the top n elements off the stack, and the applications push the terms M_i onto it, then the redex itself is the first part of the term to be evaluated on the machine. Machine evaluation thus corresponds tightly to head reduction, with the same order of evaluation of redexes.

However, in this correspondence, the machine halts with a *variable*, while successful termination in our setting requires a *skip* with an empty continuation stack. We will thus use a different approach.

$$\frac{}{S, \star \Downarrow_0 [1 \cdot S]} \qquad \frac{S, \{N/x\}M \Downarrow_w \mathcal{T}}{S\,N, \langle x \rangle .\, M \Downarrow_{w+1} \mathcal{T}} \qquad \frac{R, M \Downarrow_w [p_i \cdot S_i]_{i \leq n} \qquad (\, S_i, N \Downarrow_{v_i} \mathcal{T}_i \,)_{i \leq n}}{R, M\,;N \Downarrow_{(w + \sum_{i \leq n} v_i)} \sum_{i \leq n} p_i \mathcal{T}_i}$$

$$\frac{}{S, M \Downarrow_0 \varnothing} \qquad \frac{S\,N, M \Downarrow_w \mathcal{T}}{S, [N] .\, M \Downarrow_{w+1} \mathcal{T}} \qquad \frac{S, \pi^a_\top(M) \Downarrow_v \mathcal{T}_\top \qquad S, \pi^a_\bot(M) \Downarrow_w \mathcal{T}_\bot}{S, \boxed{a} .\, M \Downarrow_{v+w+1} \frac{1}{2}\mathcal{T}_\top + \frac{1}{2}\mathcal{T}_\bot}$$

Figure 5 The weighted probabilistic evaluation relation.

The main idea is that reduction *shortens* the runs of the machine, by removing consecutive *push* and *pop* operations, or in the case of projective reduction, removing a generator. By annotating the evaluation relation to count abstractions, applications, and generators we may then observe that this measure reduces under head reduction.

▶ **Definition 26.** *The* weighted *evaluation relation* $S, M \Downarrow_w \mathcal{T}$ *is given in Figure 5, where the* weight w *is a natural number. We extend it to a distribution of terms* $\mathcal{M}$ *by the following rule, carrying a multiset of weights* $\mathcal{W}$*.*

$$\frac{(\, S, M_i \Downarrow_{w_i} \mathcal{T}_i \,)_{i \leq n}}{S, [p_i \cdot M_i]_{i \leq n} \Downarrow_{[w_i]_{i \leq n}} \sum_{i \leq n} p_i \mathcal{T}_i}$$

The core lemma establishes that the weight in the evaluation relation decreases for $\to_{\beta \mathsf{h}}$ and $\to_\pi$ reduction steps, and is stable under $\to_{\sigma \mathsf{h}}$. However, this does not apply to terms introduced by the zero-rule, as $S, M \Downarrow_0 \varnothing$, which are the potentially non-terminating terms. Crucially for our purpose, when the evaluation returns a non-empty distribution, head-reduction on the term progresses towards a head-normal form.

▶ **Lemma 27.** *Let* $S, M \Downarrow_w \mathcal{T}$ *where* $\mathcal{T} \neq \varnothing$*.*

- *If* $M \to_{\sigma \mathsf{h}} N$ *then* $S, N \Downarrow_w \mathcal{T}$*.*
- *If* $M \to_{\beta \mathsf{h}} N$ *then* $S, N \Downarrow_v \mathcal{T}$ *with* $v < w$*.*
- *If* $M \to_\pi \mathcal{N}$ *then* $S, \mathcal{N} \Downarrow_{\mathcal{W}} \mathcal{T}$ *with* $v < w$ *for every* v *in* $\mathcal{W}$*.*

Proof. We first consider the reduction steps themselves, and then their closure under head contexts. The proof is by induction on $\Downarrow$. We consider three cases; the remaining are similar.

Beta $[N] .\, \langle x \rangle .\, M \to_\beta \{N/x\}M$ Since $\mathcal{T}$ is non-empty, the derivation must be as below left, as none of the inferences may be replaced by a zero-rule. The case is then as follows.

$$\frac{\dfrac{S, \{N/x\}M \Downarrow_w \mathcal{T}}{S\,N, \langle x \rangle .\, M \Downarrow_{w+1} \mathcal{T}}}{S, [N] .\, \langle x \rangle .\, M \Downarrow_{w+2} \mathcal{T}} \quad \to_\beta \quad S, \{N/x\}M \Downarrow_w \mathcal{T}$$

Projection $\boxed{a} .\, M \to_\pi [\, \frac{1}{2} \cdot \pi^a_\top M \,,\, \frac{1}{2} \cdot \pi^a_\bot M \,]$ Since $\mathcal{T}$ is non-empty, the derivation for the redex is the first below. The derivation for the reduct, second below, uses the evaluation rule for distributions.

$$\frac{S, \pi^a_\top M \Downarrow_v \mathcal{T}_\top \qquad S, \pi^a_\bot M \Downarrow_w \mathcal{T}_\bot}{S, \boxed{a} .\, M \Downarrow_{v+w+1} \frac{1}{2}\mathcal{T}_\top + \frac{1}{2}\mathcal{T}_\bot}$$

$$\frac{S, \pi^a_\top M \Downarrow_v \mathcal{T}_\top \qquad S, \pi^a_\bot M \Downarrow_w \mathcal{T}_\bot}{S, [\frac{1}{2} \cdot \pi^a_\top M, \frac{1}{2} \cdot \pi^a_\bot M] \Downarrow_{[v,w]} \frac{1}{2}\mathcal{T}_\top + \frac{1}{2}\mathcal{T}_\bot}$$

Sequence–generator $\boxed{a}.\, M\,;P \;\to_\sigma\; \boxed{a}.\,(M\,;P)$ $(a \notin \mathsf{fe}(P))$ The derivations are as follows, with premises stacked for space. In the second derivation $w' = w + \sum_{i\le n} u_i$ and $v' = v + \sum_{n<i\le m} u_i$.

$$\dfrac{\dfrac{\begin{array}{c}R, \pi^a_\top M \Downarrow_w [p_i \cdot S_i]_{i\le n}\\ R, \pi^a_\bot M \Downarrow_v [p_i \cdot S_i]_{n<i\le m}\end{array}}{R, \boxed{a}.\, M \Downarrow_{w+v+1} \frac{1}{2}[p_i \cdot S_i]_{i\le m}} \qquad (S_i, P \Downarrow_{u_i} \mathcal{T}_i)_{i\le m}}{R, \boxed{a}.\, M\,;P \Downarrow_{(w+v+1+\sum_{i\le m} u_i)} \sum_{i\le m} \frac{1}{2} p_i \mathcal{T}_i}$$

$$\dfrac{\dfrac{R, \pi^a_\top M \Downarrow_w [p_i \cdot S_i]_{i\le n} \qquad (S_i, P \Downarrow_{u_i} \mathcal{T}_i)_{i\le n}}{R, \pi^a_\top M\,;P \Downarrow_{w'} \sum_{i\le n} p_i \mathcal{T}_i} \qquad \dfrac{R, \pi^a_\bot M \Downarrow_v [p_i \cdot S_i]_{n<i\le m} \qquad (S_i, P \Downarrow_{u_i} \mathcal{T}_i)_{n<i\le m}}{R, \pi^a_\bot M\,;P \Downarrow_{v'} \sum_{n<i\le m} p_i \mathcal{T}_i}}{R, \boxed{a}.\,(M\,;P) \Downarrow_{(w+v+1+\sum_{i\le m} u_i)} \sum_{i\le m} \frac{1}{2} p_i \mathcal{T}_i}$$

This concludes the reduction steps. The remaining cases consider the closure under head contexts. For sequencing reduction, which leaves the weight unchanged, this is immediate, and the cases for beta-steps are straightforward and omitted. For projective reduction, let $M \to_\pi [\, \frac{1}{2} \cdot N_\top \; \frac{1}{2} \cdot N_\bot \,]$, and consider the remaining cases.

Application $[P].\, M \to_\pi [\, \frac{1}{2} \cdot [P].\, N_\top \; \frac{1}{2} \cdot [P].\, N_\bot \,]$ The derivation for $[P].\, M$ is as follows.

$$\dfrac{S\, P, M \Downarrow_w \mathcal{T}}{S, [P].\, M \Downarrow_{w+1} \mathcal{T}}$$

By induction, for the premise of this derivation we get one for the distribution $\mathcal{N} = [\frac{1}{2} \cdot N_\top, \frac{1}{2} \cdot N_\bot]$, below, where it follows that $\mathcal{T} = \frac{1}{2}\mathcal{T}_\top + \frac{1}{2}\mathcal{T}_\bot$ and $u, v < w$.

$$\dfrac{S\, P, N_\top \Downarrow_u \mathcal{T}_\top \qquad S\, P, N_\bot \Downarrow_v \mathcal{T}_\bot}{S\, P, \mathcal{N} \Downarrow_{[u,v]} \mathcal{T}}$$

Then for $\mathcal{P} = [\, \frac{1}{2} \cdot [P].\, N_\top \; \frac{1}{2} \cdot [P].\, N_\bot \,]$ we get the following derivation.

$$\dfrac{\dfrac{S\, P, N_\top \Downarrow_u \mathcal{T}_\top}{S, [P].\, N_\top \Downarrow_{u+1} \mathcal{T}_\top} \qquad \dfrac{S\, P, N_\bot \Downarrow_v \mathcal{T}_\bot}{S, [P].\, N_\bot \Downarrow_{v+1} \mathcal{T}_\bot}}{S, \mathcal{P} \Downarrow_{[u+1,v+1]} \mathcal{T}}$$

Abstraction $\langle x \rangle.\, M \to_\pi [\, \frac{1}{2} \cdot \langle x \rangle.\, N_\top \; \frac{1}{2} \cdot \langle x \rangle.\, N_\bot \,]$ The derivation for $\langle x \rangle.\, M$ is as follows.

$$\dfrac{S, \{P/x\}M \Downarrow_w \mathcal{T}}{S\, P, \langle x \rangle.\, M \Downarrow_{w+1} \mathcal{T}}$$

Given the reduction for M, we also have the following, where we abbreviate the reduct as $\mathcal{P}$.

$$\{P/x\}M \to_\pi [\, \tfrac{1}{2} \cdot \{P/x\}N_\top \; \tfrac{1}{2} \cdot \{P/x\}N_\bot \,] = \mathcal{P}$$

By induction this gives us the following derivation for $\mathcal{P}$, where again $\mathcal{T} = \frac{1}{2}\mathcal{T}_\top + \frac{1}{2}\mathcal{T}_\bot$ and $u, v < w$.

$$\dfrac{S, \{P/x\}N_\top \Downarrow_u \mathcal{T}_\top \qquad S, \{P/x\}N_\bot \Downarrow_v \mathcal{T}_\bot}{S, \mathcal{P} \Downarrow_{[u,v]} \mathcal{T}}$$

Then for $\mathcal{N} = [\,\frac{1}{2}\cdot\langle x\rangle.\,N_\top\ \frac{1}{2}\cdot\langle x\rangle.\,N_\bot\,]$ we get the required derivation.

$$\dfrac{\dfrac{S, \{P/x\}N_\top \Downarrow_u \mathcal{T}_\top}{S\,P, \langle x\rangle.\,N_\top \Downarrow_{u+1} \mathcal{T}_\top} \qquad \dfrac{S, \{P/x\}N_\bot \Downarrow_v \mathcal{T}_\bot}{S\,P, \langle x\rangle.\,N_\bot \Downarrow_{v+1} \mathcal{T}_\bot}}{S, \mathcal{N} \Downarrow_{[u+1,v+1]} \mathcal{T}}$$ ◀

We apply the above lemma to relate machine evaluation to head reduction in the following way: if evaluation returns a distribution of weight p, then head-reduction terminates with probability at least p.

▶ **Lemma 28.** *If $S, \mathcal{M} \Downarrow_{\mathcal{W}} \mathcal{T}$ then $\mathcal{M} \twoheadrightarrow_{\mathsf{h}} \mathcal{N}_0 + \mathcal{N}_1$ where all terms in $\mathcal{N}_0$ are head normal and $|\mathcal{N}_0| \geq |\mathcal{T}|$.*

Proof. Let $\mathcal{M} = [p_i \cdot M_i]_{i\leq n}$ evaluate by the following derivation.

$$\dfrac{(\,S, M_i \Downarrow_{w_i} \mathcal{T}_i\,)_{i\leq n}}{S, [p_i\cdot M_i]_{i\leq n} \Downarrow_{[w_i]_{i\leq n}} \sum_{i\leq n} p_i\mathcal{T}_i}$$

First, we head-reduce any M_i where $\mathcal{T}_i \neq \varnothing$. This process terminates by induction on $\mathcal{W}$, using Lemma 27: a β-step or projective step reduces $\mathcal{W}$, while sequencing reduction alone is terminating and does not increase it.

Then the distribution $\mathcal{N}_0 = [\,p_i\cdot M_i \mid i\leq n,\ \mathcal{T}_i \neq \varnothing\,]$ is head-normal. The weights of $\mathcal{N}_0$ and $\mathcal{T}$ are as follows.

$$|\mathcal{N}_0| = \sum[\,p_i \mid i\leq n,\ \mathcal{T}_i \neq \varnothing\,] \qquad |\mathcal{T}| = \sum_{i\leq n} p_i|\mathcal{T}_i|$$

It follows that $|\mathcal{N}_0| \geq |\mathcal{T}|$. ◀

Our final theorem then ties everything together: typing gives successful evaluation on the machine, which in turn gives termination of head reduction.

▶ **Theorem 22 (restatement).** *For a closed M, if $M : \overline{A} \overset{p}{\Rightarrow} \overline{C}$ then $M \twoheadrightarrow_{\mathsf{h}} \mathcal{N}_0 + \mathcal{N}_1$ where all terms in $\mathcal{N}_0$ are head normal and $|\mathcal{N}_0| \geq p$.*

Proof. We provide M with an input stack consisting only of zero terms 0_A. Let S be the stack of zero terms for $\overline{A}$. By Proposition 20 zero-terms inhabit their types, so that $S : \overline{A}$.

By Lemma 25 we have $S \in \textsc{run}(\overline{A})$ and $M \in \textsc{run}(\overline{A} \overset{p}{\Rightarrow} \overline{C})$. By the definition of $\textsc{run}(-)$ this gives an evaluation $S, M \Downarrow \mathcal{T}$ where $|\mathcal{T}| \geq p$.

For the corresponding weighted derivation $S, M \Downarrow_w \mathcal{T}$, Lemma 28 gives a head reduction $M \twoheadrightarrow_{\mathsf{h}} \mathcal{N}_0 + \mathcal{N}_1$ where all terms in $\mathcal{N}_0$ are head normal and $|\mathcal{N}_0| \geq |\mathcal{T}| \geq p$. ◀

A Mixed Linear and Graded Logic: Proofs, Terms, and Models

Victoria Vollmer
School of Computing, University of Kent, UK

Danielle Marshall
School of Computing, University of Kent, UK

Harley Eades III
Computer Science, Augusta University, GA, USA

Dominic Orchard
School of Computing, University of Kent, UK
Department of Computer Science and Technology, University of Cambridge, UK

Abstract

Graded modal logics generalise standard modal logics via families of modalities indexed by an algebraic structure whose operations mediate between the different modalities. The graded "of-course" modality $!_r$ captures how many times a proposition is used and has an analogous interpretation to the of-course modality from linear logic; the of-course modality from linear logic can be modelled by a linear exponential comonad and graded of-course can be modelled by a graded linear exponential comonad. Benton showed in his seminal paper on Linear/Non-Linear logic that the of-course modality can be split into two modalities connecting intuitionistic logic with linear logic, forming a symmetric monoidal adjunction. Later, Fujii et al. demonstrated that every graded comonad can be decomposed into an adjunction and a "strict action". We give a similar result to Benton, leveraging Fujii et al.'s decomposition, showing that graded modalities can be split into two modalities connecting a graded logic with a graded linear logic. We propose a sequent calculus, its proof theory and categorical model, and a natural deduction system which we show is isomorphic to the sequent calculus system. Interestingly, our system can also be understood as Linear/Non-Linear logic composed with an action that adds the grading, further illuminating the shared principles between linear logic and a class of graded modal logics.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases linear logic, graded modal logic, adjoint decomposition

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.32

Related Version *Full version with appendices*: `https://arxiv.org/abs/2401.17199` [40]

Funding This work was supported in part by the EPSRC grant EP/T013516/1 (*Verifying Resource-like Data Use in Programs via Types*). The second author received support through Schmidt Sciences, LLC.

Acknowledgements We thank all the anonymous reviewers of this, and previous versions, of this paper. We are also grateful for discussions with Peter Hanukaev and helpful comments from Paulo Torrens on a draft of this manuscript.

1 Introduction

Intuitionistic logic has a central role in the foundations of programming language theory, providing a logical basis for type theories and type systems, and other program reasoning principles. A significant amount of the expressivity of proof systems for intuitionistic logic (both natural deduction and sequent calculus forms) lies within the structure of the

hypotheses – the *context*. Probing the foundations of this part of the logic has, perhaps surprisingly, yielded the very fertile field of *substructural logics* [39] including influential logics such as linear logic [14] and its variants, and the Lambek calculus [25].

By restricting the manipulation of hypotheses in the context we typically arrive at logics which align more closely with physical reality, where propositions are instead "resources" that cannot necessarily be copied, discarded, or reordered. Such restricted logics have been used to construct type systems for safely manipulating values that should be treated in a resourceful way, such as file handlers, pointers to mutable memory, or channels [43, 42].

However, such pervasive restrictions often hamper expressivity and thus some substructural logics then seek to carefully control the reintroduction of structural rules. For example, linear logic provides the ! modality ("of course") for reintroducing weakening and contraction of propositions, which linear logic otherwise prohibits. However, this modality is coarse-grained: for those propositions under the modality, it re-enables all the structural rules that have been removed in linear logic. *Subexponentials* instead aim to be more fine-grained, offering families of modalities capturing specific structural rules [8, 22]. The related notion of *grading* [12, 31] gives an alternate view, providing an indexed family of modalities whose indices are subject to an algebra which accounts for any structural rules applied: structural rules are "counted" by the algebra (whose operations mirror the shape of structural rules). Bounded Linear Logic [15] is a special case where the family of modalities $!_n A$ uses indices n which are natural numbers (or polynomial terms over naturals) counting the upper bound on usage of the proposition A. Various systems generalise this approach to arbitrary semirings to capture data-flow properties [1, 2, 5, 12, 13, 24, 28, 31, 33, 34, 36]. Such graded systems annotate hypotheses/variables in the context with elements of the semiring ("grades") denoting their usage, e.g., $x :_0 A \vdash t : B$ types a term t which does not use x and $y :_{1+1} A \vdash t' : B$ types a term t' in which y is used in two different subterms once each, accounted for by the semiring addition. A graded modality *internalises* the semiring grade, causing a multiplication to the grades of any captured dependencies when the graded modality is introduced, e.g., $y :_{0*(1+1)} A \vdash \Box t' : \Box_0 B$.

We seek here to further understand the underlying structure of graded modal logics by following an "adjoint resolution" approach à la Benton's seminal "A Mixed Linear and Non-Linear Logic: Proofs, Terms and Models" at CSL 1994 [3]. Benton showed that the exponential modality of linear logic (modelled by a comonad) can be decomposed into an adjunction, defining a pair of "adjoint" logics (a linear logic and a non-linear intuitionistic, or "Cartesian", logic) which embed into each other [3]. This provides a beautiful reduction of the core features of linear logic and its non-linearity modality. *Adjoint logic* applies the same idea but to subexponentials [37, 38]. We follow the same scheme, via the adjoint decomposition of graded modalities which generalise linear logic's ! and which are traditionally modelled by graded exponential comonads [5, 6, 12, 13, 24, 34]. Whilst Benton's work has a pair of adjoint modalities mediating between the two sublogics, we have a pair of a modality Lin and a *graded modality* Grd_r. We give a categorical model, showing that these are captured by an LNL-like adjunction paired with a "strict action" for incorporating the grading, following the Fujii-Katsumata-Melliès adjoint decomposition of graded (co)monads [10, 24]. The result is a pair of logics which serve to explain and clarify the relationship between linearity and grading. We call our system Mixed Graded/Linear (mGL) Logic.

This pair of logics also shines light on a relationship between two styles of graded system in the literature: those which take linear types as their basis augmented with a graded modality [6, 12, 31] versus those with no base notion of linearity where grading is pervasive, tracking all substructurality [1, 2, 4, 7, 28, 30, 34]. Our linear fragment is analogous to the

former whilst our graded fragment is analogous to the latter. The mutual embedding shows that these two styles of graded logics have a similar relationship to the adjoint relationship of intuitionistic logic and linear logic.

Aside from the internal motivation of better understanding the relationship between grading and linearity, an external motivation for this work is that it can provide a basis for flexible, safe programming with resources. By separating out the linear fragment from an intuitionistic graded fragment, one could avoid the strictures of linearity for working with standard data types which need not be linear, working only in the linear fragment for handling resources like file handles. The mutual embedding would allow the programmer to move smoothly between these two subcalculi, as seen also in other adjunction-based calculi, e.g., for concurrent programming [35]. The focus here however is on the core theory rather than developing these applications yet.

Since our focus is on the relationship between grading and linearity, we consider the semiring-graded modalities that generalise linear logic's !. Other flavours of graded modality (e.g., graded monads for capturing side effecting behaviour [23, 32]) are not considered here.

Roadmap

Section 2 defines a pair of sequent calculi, the mixed fragment MS, which has both linear and graded assumptions, and the graded fragment GS, which has only graded assumptions and no function arrow. As described above, these calculi have a mutual embedding via modalities between the two. Section 3 considers the categorical model of mGL leveraging recent work on the adjoint resolution of graded comonads [10, 24]. Section 4 provides the natural deduction formulation of the calculus, which is proved equivalent to the sequent calculus version. Section 5 discusses how this work gives a view on the landscape of graded systems in the literature and considers other related work and future applications.

A version of this paper with the appendices providing full proof details can be found on the arXiv [40].

2 Mixed Graded/Linear Logic: Proofs and Terms

We present first a sequent calculus for Mixed Graded/Linear logic, which comes in the form of a term assignment. Figure 1 collects the term syntax for reference; it will also be used in Section 4 for the natural deduction formulation. The syntax is explained with reference to its associated proof rules in the next section.

$$
\begin{array}{lll}
\text{(GS/GT)} & t ::= & x \mid \mathsf{j} \mid \mathsf{let}\,\mathsf{j} = t_1 \,\mathsf{in}\, t_2 \\
\textit{graded} & \mid & (t_1, t_2) \mid \mathsf{let}\,(x, y) = t_1 \,\mathsf{in}\, t_2 \\
& \mid & \mathsf{Lin}\; l
\end{array}
\qquad
\begin{array}{lll}
\text{(MS/MT)} & l ::= & x \mid \mathsf{i} \mid \mathsf{let}\,\mathsf{i} = l_1 \,\mathsf{in}\, l_2 \\
\textit{linear} & \mid & (l_1, l_2) \mid \mathsf{let}\,(x, y) = l_1 \,\mathsf{in}\, l_2 \\
& \mid & \lambda x.l \mid l_1\; l_2 \\
& \mid & \mathsf{Grd}\; r\; t \mid \mathsf{let}\,\mathsf{Grd}\; r\; x = l_1 \,\mathsf{in}\, l_2 \\
& \mid & \mathsf{Unlin}\; z \\
& \mid & \mathsf{let}\,\mathsf{j} = z \,\mathsf{in}\, l \mid \mathsf{let}\,(x, y) = z \,\mathsf{in}\, l
\end{array}
$$

Variables are ranged over by x, y, z in both fragments.

Terms are mostly grouped above with introduction forms followed by elimination forms, though note that in the last two lines of syntax for l there are additional eliminators: for the linear modality (Unlin), for units j, and for tensors coming from the graded context.

Figure 1 Collected term syntax.

Benton's approach has two proof systems [3]: one system of linear propositions (the L of LNL) with two contexts for linear and non-linear propositions respectively, and one system of non-linear propositions (the NL in LNL). We generalise this approach to the graded setting by replacing the non-linear parts with *graded* notions. Thus, our system (mGL) has two analogous proof systems: one of linear propositions with two contexts for linear and graded propositions, with judgments subscripted as $\vdash_{\mathsf{MS}}$ (for "Mixed (linear/graded) Sequent"), and one system of graded propositions, with judgments subscripted as $\vdash_{\mathsf{GS}}$ ("Graded Sequent").

The syntax of Benton's propositions is split into two, "*conventional*" (i.e., Cartesian / non-linear) and *linear* [3]. The syntax of our propositions is analogously split into two, *graded* and *linear*:

$$\begin{array}{ll}\text{(Graded)} & X, Y, Z ::= \mathsf{J} \mid X \boxtimes Y \mid \mathsf{Lin}\, A \\ \text{(Linear)} & A, B, C ::= \mathsf{I} \mid A \otimes B \mid A \multimap B \mid \mathsf{Grd}_r\, X\end{array}$$

where J and I are unit types, and $\boxtimes$ and $\otimes$ are tensor (product) operators in their respective domains. In the case of the linear domain, the product is the standard multiplicative conjunction. The Lin modality encapsulates a linear proposition as a graded proposition, and the Grd_r modality encapsulates a graded proposition (at grade r, whose structure is defined below) as a linear proposition. Thus, the two logics are interconnected by $\mathsf{Grd}_r\, X$ and $\mathsf{Lin}\, A$. Using these two modalities we will later define graded modalities $\Box_r A$ as $\mathsf{Grd}_r\, (\mathsf{Lin}\, A)$, similarly to how the of-course modality $!A$ can be defined in LNL logic as the composition of two adjoint modalities.

▶ **Definition 1.** *Grades (ranged over by r, s) are drawn from a semiring parameterizing the system $(\mathcal{R}, 1, *, 0, +, \leq)$ with preorder $(\mathcal{R}, \leq)$ such that both $*$ and $+$ are monotonic wrt $\leq$.*

The semiring governs the structural rules: the additive part of the semiring is involved in weakening and contraction, and the multiplicative part in usage and composition. Various concrete examples of interesting semirings are given at the end of Subsection 2.1.

Section 4 develops an equivalent natural deduction formulation of mGL. We then show that the natural deduction and the sequent calculus are interderivable without modifying the term witnessing a derivation. Thus, any semantic model of one is a model of the other. We opt to focus on the sequent calculus form for now without loss of generality.

2.1 Sequent Calculus

We first define contexts used in the judgments:

▶ **Definition 2** (Graded contexts)**.** *Suppose $(\mathcal{R}, 1, *, 0, +, \leq)$ is a preordered semiring (Def. 1). Then* grade vectors *δ are sequences of $\mathcal{R}$, contexts Δ are sequences of graded formulas X, and contexts Γ are sequences of linear formulas:*

$$\delta := \emptyset \mid \delta, r \qquad \Delta := \emptyset \mid \Delta, x : X \qquad \Gamma := \emptyset \mid \Gamma, x : A$$

The comma operator is overloaded for sequence concatenation, i.e., we can write δ_1, δ_2 and Δ_1, Δ_2, which further requires that Δ_1 and Δ_2 are disjoint contexts.

A graded context *$\delta \odot \Delta$ is a pairing of a grade vector and a context defined as follows:*

$$\emptyset \odot \emptyset = \emptyset \qquad (\delta, r) \odot (\Delta, x : X) = (\delta \odot \Delta), x : (r \odot X)$$

where $r \odot X$ pairs a formula with a grade r capturing (by the rules of the system) how the formula X (named x) is used to form a judgment.

We lift the operations of semirings to grade vectors, forming a semimodule, with the pointwise addition and scalar multiplication defined in a standard way:

$$\emptyset + \emptyset = \emptyset \qquad\qquad r * \emptyset = \emptyset$$
$$(\delta_1, r_1) + (\delta_2, r_2) = (\delta_1 + \delta_2), (r_1 + r_2) \qquad\qquad r * (\delta, s) = (r * \delta), (r * s)$$

Addition of grade vectors requires the vectors to be of the same length.

The judgment form for our fully graded logic $\delta \odot \Delta \vdash_{\mathsf{GS}} t : X$ captures a concluding proposition X under the graded context of assumptions $\delta \odot \Delta$. Mixed graded/linear logic judgments $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l : A$ are similar but also have a context Γ of linear assumptions which, being linear, do not have a corresponding grade vector.

The two judgments $\vdash_{\mathsf{GS}}$ and $\vdash_{\mathsf{MS}}$ (also called *sub-logics* or *fragments*) are defined by mutual induction. We present conceptually related rules from both systems side-by-side where possible, or one-after-the-other, in the order GS then MS.

The identity (axiom) rules are:

$$\textsc{id}_{\mathsf{GS}}\ \frac{}{1 \odot x : X \vdash_{\mathsf{GS}} x : X} \qquad\qquad \textsc{id}_{\mathsf{MS}}\ \frac{}{\emptyset \odot \emptyset; x : A \vdash_{\mathsf{MS}} x : A}$$

The multiplicative identity 1 is the "default" grade for formulas in the graded logic GS (left), in the sense that we can think of the right-hand side of the judgment as also implicitly having grade 1. The graded identity rule says that a graded formula that is used must have the default grade. For example, in the natural number semiring $(\mathbb{N}, 1, *, 0, +, =)$ the multiplicative identity $1 \in \mathbb{N}$ captures linear usage. The mixed identity rule types linear assumption use, requiring just a singleton linear context (forcing a lack of weakening). It also requires that there are no graded formulas in context – the graded context is empty $\emptyset$.

The "cut" rules are:

$$\textsc{cut}_{\mathsf{GS}}\ \frac{\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t_1 : X \qquad (\delta_1, r, \delta_3) \odot (\Delta_1, x : X, \Delta_3) \vdash_{\mathsf{GS}} t_2 : Y}{(\delta_1, r * \delta_2, \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3) \vdash_{\mathsf{GS}} [t_1/x]t_2 : Y}$$

$$\textsc{cut}_{\mathsf{MS}}\ \frac{\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MS}} l_1 : A \qquad \delta_1 \odot \Delta_1; (\Gamma_1, x : A, \Gamma_3) \vdash_{\mathsf{MS}} l_2 : B}{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); (\Gamma_1, \Gamma_2, \Gamma_3) \vdash_{\mathsf{MS}} [l_1/x]l_2 : B} \qquad \textsc{gcut}_{\mathsf{MS}}\ \frac{\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t : X \qquad (\delta_1, r, \delta_3) \odot (\Delta_1, x : X, \Delta_3); \Gamma \vdash_{\mathsf{MS}} l : B}{(\delta_1, r * \delta_2, \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); \Gamma \vdash_{\mathsf{MS}} [t/x]l : B}$$

The $\textsc{cut}_{\mathsf{GS}}$ rule provides a cut through a graded proposition X of grade r in the receiving context (second premise). Thus, the resulting term uses semiring multiplication (lifted to contexts, Def. 2) to capture sequential usage, scaling the grade vector δ_2 of the cut term t_1 by r. The $\textsc{cut}_{\mathsf{MS}}$ rules provides a cut through a linear proposition A and has no effect on the graded contexts. However, MS has a further cut rule $\textsc{gcut}_{\mathsf{MS}}$ for graded propositions in its graded context, incurring a scaling similarly to $\textsc{cut}_{\mathsf{GS}}$. This pattern occurs throughout: operations applied to the graded context in GS have a sister rule in MS applying the same operation in the MS graded context.

Both sub-logics have free use of exchange:

$$\textsc{ex}_{\mathsf{GS}}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : Y, \Delta_2) \vdash_{\mathsf{GS}} t : Z}{(\delta_1, r_2, r_1, \delta_2) \odot (\Delta_1, y : Y, x : X, \Delta_2) \vdash_{\mathsf{GS}} t : Z}$$

$$\textsc{ex}_{\mathsf{MS}}\ \frac{\delta \odot \Delta; (\Gamma_1, x : A, y : B, \Gamma_2) \vdash_{\mathsf{MS}} l : C}{\delta \odot \Delta; (\Gamma_1, y : B, x : A, \Gamma_2) \vdash_{\mathsf{MS}} l : C} \qquad \textsc{gex}_{\mathsf{MS}}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : Y, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : B}{(\delta_1, r_2, r_1, \delta_2) \odot (\Delta_1, x : Y, y : X, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : B}$$

Exchanging graded propositions simultaneously exchanges their grades in the grade vector.

We can use weakening and contraction in the graded system and the mixed system within the graded contexts, with the semiring's 0 representing weakened hypotheses and the grades of contracted hypotheses combined via semiring addition $+$:

$$\textsc{weak}_{\mathsf{GS}}\ \frac{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2) \vdash_{\mathsf{GS}} t : Y}{(\delta_1, 0, \delta_2) \odot (\Delta_1, x : X, \Delta_2) \vdash_{\mathsf{GS}} t : Y} \qquad \textsc{cont}_{\mathsf{GS}}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : X, \Delta_2) \vdash_{\mathsf{GS}} t : Y}{(\delta_1, r_1 + r_2, \delta_2) \odot (\Delta_1, x : X, \Delta_2) \vdash_{\mathsf{GS}} [x/y]t : Y}$$

$$\textsc{weak}_{\mathsf{MS}}\ \frac{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : B}{(\delta_1, 0, \delta_2) \odot (\Delta_1, x : X, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : B} \qquad \textsc{cont}_{\mathsf{MS}}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : X, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : B}{(\delta_1, r_1 + r_2, \delta_2) \odot (\Delta_1, x : X, \Delta_2); \Gamma \vdash_{\mathsf{MS}} [x/y]l : B}$$

The left and right rules for units for the graded and mixed logics are akin to linear logic:

$$\textsc{unit}_L^{\mathsf{J}}\ \frac{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2) \vdash_{\mathsf{GS}} t : X}{(\delta_1, r, \delta_2) \odot (\Delta_1, x : \mathsf{J}, \Delta_2) \vdash_{\mathsf{GS}} \mathsf{let\,j} = x \,\mathsf{in}\, t : X} \qquad \textsc{unit}_R^{\mathsf{J}}\ \frac{}{\emptyset \odot \emptyset \vdash_{\mathsf{GS}} \mathsf{j} : \mathsf{J}}$$

$$\textsc{unit}_L^{\mathsf{I}}\ \frac{\delta \odot \Delta; (\Gamma_1, \Gamma_2) \vdash_{\mathsf{MS}} l : A}{\delta \odot \Delta; (\Gamma_1, x : \mathsf{I}, \Gamma_2) \vdash_{\mathsf{MS}} \mathsf{let\,i} = x \,\mathsf{in}\, l : A} \qquad \textsc{unit}_R^{\mathsf{I}}\ \frac{}{\emptyset \odot \emptyset; \emptyset \vdash_{\mathsf{MS}} \mathsf{i} : \mathsf{I}}$$

$$\textsc{unit}_L^{\mathsf{J}-\mathsf{MS}}\ \frac{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : A}{(\delta_1, r, \delta_2) \odot (\Delta_1, z : \mathsf{J}, \Delta_2); \Gamma \vdash_{\mathsf{MS}} \mathsf{let\,j} = z \,\mathsf{in}\, l : A}$$

Thus, in GS, we can eliminate a graded unit j at an arbitrary grade r, whereas the linear unit i in MS gets eliminated from the linear context. The additional left rule $(\textsc{unit}_L^{\mathsf{J}-\mathsf{MS}})$ for MS again similarly eliminates graded units J in the graded context.

Tensor products are then eliminated in each fragment as follows:

$$\boxtimes_L\ \frac{(\delta_1, r, r, \delta_2) \odot (\Delta_1, x : X, y : Y, \Delta_2) \vdash_{\mathsf{GS}} t : Z}{(\delta_1, r, \delta_2) \odot (\Delta_1, z : X \boxtimes Y, \Delta_2) \vdash_{\mathsf{GS}} \mathsf{let}\,(x, y) = z \,\mathsf{in}\, t : Z} \qquad \boxtimes_R\ \frac{\delta_1 \odot \Delta_1 \vdash_{\mathsf{GS}} t_1 : X \quad \delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t_2 : Y}{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2) \vdash_{\mathsf{GS}} (t_1, t_2) : X \boxtimes Y}$$

$$\otimes_L\ \frac{\delta \odot \Delta; (\Gamma_1, x : A, y : B, \Gamma_2) \vdash_{\mathsf{MS}} l : C}{\delta \odot \Delta; (\Gamma_1, z : A \otimes B, \Gamma_2) \vdash_{\mathsf{MS}} \mathsf{let}\,(x, y) = z \,\mathsf{in}\, l : C} \qquad \otimes_R\ \frac{\delta_1 \odot \Delta_1; \Gamma_1 \vdash_{\mathsf{MS}} l_1 : A \quad \delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MS}} l_2 : B}{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); (\Gamma_1, \Gamma_2) \vdash_{\mathsf{MS}} (l_1, l_2) : A \otimes B}$$

$$\boxtimes_{L-\mathsf{MS}}\ \frac{(\delta_1, r, r, \delta_2) \odot (\Delta_1, x : X, y : Y, \Delta_2); \Gamma \vdash_{\mathsf{MS}} l : A}{(\delta_1, r, \delta_2) \odot (\Delta_1, z : X \boxtimes Y, \Delta_2); \Gamma \vdash_{\mathsf{MS}} \mathsf{let}\,(x, y) = z \,\mathsf{in}\, l : A}$$

The left rule for $\boxtimes$ eliminates from the graded context at any grade r, where the components of the tensor product both inherit this grade in the premise. Reading instead top-down, the graded tensor product requires that both components are graded with the same grade; this is similar to linear products, where both components are linear.

Note that Benton has two left rules for (non-linear) tensor products, in the "projection" style. We instead must use the pattern matching style for the soundness of grading so that each component is bound to a variable with the same grade.

Only the mixed linear-graded system has implication, and only on linear propositions, thus we have $\multimap$ in MS with left and right rules:

$$\multimap_L \quad \frac{\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MS}} l_2 : A \qquad \delta_1 \odot \Delta_1; (\Gamma_1, x : B, \Gamma_3) \vdash_{\mathsf{MS}} l_1 : C}{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); (\Gamma_1, z : A \multimap B, \Gamma_2, \Gamma_3) \vdash_{\mathsf{MS}} [z\, l_2/x] l_1 : C} \qquad \multimap_R \quad \frac{\delta \odot \Delta; (\Gamma, x : A) \vdash_{\mathsf{MS}} l : B}{\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} \lambda x.l : A \multimap B}$$

In Lemma 4, we recover a graded implication through the modal operators of the system in the same way that Melliès did for (ungraded) LNL logic [29].

We now consider the modal operators Lin and Grd_r which connect the two sub-logics.

The right rule for the Lin modality transports a linear formula from the linear system MS into the graded system GS where it can be reasoned with non-linearly as accounted for by grading. The corresponding left rule is akin to *dereliction* from linear logic, enabling a linear assumption $x : A$ to treated as a (renamed) graded assumption $z : \mathsf{Lin}\, A$ at grade 1:

$$\mathsf{Lin}_L \quad \frac{\delta \odot \Delta; (x : A, \Gamma) \vdash_{\mathsf{MS}} l : B}{(\delta, 1) \odot (\Delta, z : \mathsf{Lin}\, A); \Gamma \vdash_{\mathsf{MS}} [\mathsf{Unlin}\, z/x] l : B} \qquad \mathsf{Lin}_R \quad \frac{\delta \odot \Delta; \emptyset \vdash_{\mathsf{MS}} l : B}{\delta \odot \Delta \vdash_{\mathsf{GS}} \mathsf{Lin}\, l : \mathsf{Lin}\, B}$$

The other modal operator Grd, or rather the family of modal operators Grd_r, transports a graded formula with its grade into the linear system where it can be reasoned with linearly:

$$\mathsf{Grd}_L \quad \frac{(\delta, r) \odot (\Delta, x : X); \Gamma \vdash_{\mathsf{MS}} l : C}{\delta \odot \Delta; (z : \mathsf{Grd}_r\, X, \Gamma) \vdash_{\mathsf{MS}} \mathsf{let}\, \mathsf{Grd}\, r\, x = z\, \mathsf{in}\, l : C} \qquad \mathsf{Grd}_R \quad \frac{\delta \odot \Delta \vdash_{\mathsf{GS}} t : X}{r * \delta \odot \Delta; \emptyset \vdash_{\mathsf{MS}} \mathsf{Grd}\, r\, t : \mathsf{Grd}_r\, X}$$

The right rule is akin to *promotion* for Grd_r where we subsequently scale the graded context by the grade r. The left rule "unboxes" a graded modality $\mathsf{Grd}_r\, X$ providing access to the X formula "inside", graded at r.

Perhaps the most remarkable property of these modal operators is that they decompose semiring-graded necessity modalities into $\Box_r A = \mathsf{Grd}_r\, (\mathsf{Lin}\, A)$ [24] within the mixed system. In fact, their introduction and elimination rules are derivable:

▶ **Lemma 3** (mGL Graded Necessity Modality). *The following are derivable:*

$$\Box_E \quad \frac{\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MS}} l_1 : \Box_r A \qquad (\delta_1, r, \delta_3) \odot (\Delta_1, x : \mathsf{Lin}\, A, \Delta_3); \Gamma_1 \vdash_{\mathsf{MS}} l_2 : B}{(\delta_1, \delta_2, \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); (\Gamma_1, \Gamma_2) \vdash_{\mathsf{MS}} \mathsf{let}\, \mathsf{Grd}\, r\, x = l_1\, \mathsf{in}\, l_2 : B} \qquad \Box_i \quad \frac{\delta \odot \Delta; \emptyset \vdash_{\mathsf{MS}} l : A}{(r * \delta) \odot \Delta; \emptyset \vdash_{\mathsf{MS}} \mathsf{Grd}\, r\, (\mathsf{Lin}\, l) : \Box_r A}$$

Proof. The elimination rule follows by applying Grd_L to the second premise and then applying cut with the first premise. The introduction rule follows by Lin_R then Grd_R. ◀

The previous lemma reveals a lot about the structure of existing graded type systems. First, graded hypotheses, usually denoted with their grade $x :_r A$ in the literature (e.g. [1, 31]), are graded hypotheses $r \odot \mathsf{Lin}\, A$ where the linear formula has been transported into the graded system. Second, the restriction to only graded variables in the promotion rule is very explicit in the introduction rule $\Box_i$ above. Third, the elimination rule (here $\Box_E$) is really the left rule for Grd followed by a cut. Thus, graded type systems, whilst typically of a natural deduction form, incorporate a little of the flavour of sequent calculi in the rules for graded modalities because of the integrated cut.

Using the modal operators we can derive a graded implication of the form $\mathsf{Grd}_r\, X \multimap A$:

▶ **Lemma 4** (mGL Graded Implication). *The following rules are derivable:*

$$\multimap_{GL}\ \frac{\begin{array}{c}\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t : X \\ (\delta_1, \delta_3) \odot (\Delta_1, \Delta_3); (\Gamma_1, x : A, \Gamma_2) \vdash_{\mathsf{MS}} l : B\end{array}}{(\delta_1, r * \delta_2, \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); (\Gamma_1, z : \mathsf{Grd}_r\, X \multimap A, \Gamma_2) \vdash_{\mathsf{MS}} [z\,(\mathsf{Grd}\, r\, t)/x]l : B}$$

$$\multimap_{GR}\ \frac{(\delta, r) \odot (\Delta, x : X); \Gamma \vdash_{\mathsf{MS}} l : A}{\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} \lambda y.(\mathsf{let}\ \mathsf{Grd}\, r\, x = y\, \mathsf{in}\, l) : \mathsf{Grd}_r\, X \multimap A}$$

Proof. The left rule follows by applying Grd_R to the first premise, and then $\multimap_L$ using the second premise. The right rule follows by applying Grd_L to the premise and then $\multimap_R$. ◀

The final rules are the approximation rules for grades:

$$\mathrm{SUB}_{\mathsf{GS}}\ \frac{\delta_1 \odot \Delta \vdash_{\mathsf{GS}} t : X \quad \delta_1 \leq \delta_2}{\delta_2 \odot \Delta \vdash_{\mathsf{GS}} t : X} \qquad \mathrm{SUB}_{\mathsf{MS}}\ \frac{\delta_1 \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l : B \quad \delta_1 \leq \delta_2}{\delta_2 \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l : B}$$

Approximation allows for the abstraction of grades along an ordering. For example, in the semiring $(\mathbb{N}, 1, *, 0, +, \leq)$ where the order is the usual ordering on natural numbers, then a grade r stands for "at-most r", generalising the notion of "affine" usage tracking. Disabling the ordering by forcing it to be true only in the case of reflexive pairs (i.e., an equality relation) results in exact usage tracking. This demonstrates that having a notion of approximation results in a more general usage tracking framework.

▶ **Example 5** (Derivation). As an example derivation in the natural numbers semiring $(\mathbb{N}, 1, *, 0, +, \leq)$, for "affine" usage tracking, the following copies an assumption to make a pair in the graded fragment, then uses an approximation, and then transports the pair into the linear fragment, scaling its grades further:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{1 \odot x : X \vdash_{\mathsf{GS}} x : X}\,\mathrm{ID}_{\mathsf{GS}} \quad \dfrac{}{1 \odot y : X \vdash_{\mathsf{GS}} y : X}\,\mathrm{ID}_{\mathsf{GS}}}{1, 1 \odot x : X, y : X \vdash_{\mathsf{GS}} (x, y) : X \boxtimes X}\,\boxtimes_R}{2 \odot x : X \vdash_{\mathsf{GS}} [x/y](x, y) : X \boxtimes X \quad 2 \leq 3}\,\mathrm{CONT}_{\mathsf{GS}}}{3 \odot x : X \vdash_{\mathsf{GS}} (x, x) : X \boxtimes X}\,\mathrm{SUB}_{\mathsf{GS}}}{2 * 3 \odot x : X; \emptyset \vdash_{\mathsf{MS}} \mathsf{Grd}\, 2\, (x, x) : \mathsf{Grd}_2\, (X \boxtimes X)}\,\mathsf{Grd}_R$$

▶ **Example 6** (None-One-Tons [28]). The semiring over $\{0, 1, \omega\}$ with $0 \leq \omega$ and $1 \leq \omega$, where $+$ and $*$ are saturating at ω, can be used to distinguish between linear and various non-linear uses: assigning $r = 1$ to linear usage, $r = 0$ to non-usage (when a resource is discarded), and $r = \omega$ to arbitrary usage.

Note, however, that even with the above semiring we are unable to exactly represent the exponential modality ! from linear logic via some particular grade r within the graded logic. This is because no matter which grade we choose, we are able to "push" the grade into the tensor product using this graded tensor elimination ($\boxtimes_L$), allowing derivation of $\mathsf{Grd}_r\,(X \boxtimes Y) \multimap (\mathsf{Grd}_r\,X \otimes \mathsf{Grd}_r\,Y)$, and yet in linear logic it is not possible to derive $!(A \otimes B) \multimap !A \otimes !B$. Therefore, our logic cannot reduce to Benton's LNL logic simply by taking the Cartesian (trivial) semiring, as one might expect at first glance. This quality is typical of *graded base* systems, so reconciling these with linear logic requires some additional structure on the semiring [18] (though this is not the focus here).

On the other hand, notice that we have another way to represent graded products: as linear products wrapped in the derived graded modality, or $\Box_r(A \otimes B)$. Importantly, here it is not possible to "push" the grade "through" the tensor as we can for the graded product; we cannot derive $(\Box_r A) \otimes (\Box_r B)$. This representation of graded products thus has behaviour more typical of a *linear base* graded type system, with our combined logic again giving us a clearer understanding of the relationship between these contrasting styles.

▶ **Example 7** (Security levels)**.** Information-Flow Control properties can be tracked by instantiating the semiring with a lattice of security levels [12], e.g., with $(\{\mathsf{Lo} \leq \mathsf{Hi}\}, \mathsf{Lo}, \wedge, \mathsf{Hi}, \vee)$ where Hi-graded inputs are treated as irrelevant: we cannot depend on any high-security inputs when building a low-security graded output $\mathsf{Grd}_{\mathsf{Lo}}A$.

▶ **Example 8** (Sensitivity)**.** The real number semiring $(\mathbb{R}, 1, *, 0, +, \leq)$ can be leveraged to capture a notion of numerical sensitivity in programs/logic [11, 9], where a program is k-sensitive (for $k \in \mathbb{R}$) in a variable if a change ϵ in its inputs to x produces at most a change of $k\epsilon$ in the output of the program. This instantiation of the system tracks sensitivities as grades where additional dependent-type-based mechanisms are needed to lift program values into the types, e.g., $\mathsf{scale} : (k : \mathbb{R}) \to \mathsf{Grd}_k\mathbb{R} \to \mathbb{R}$.

2.2 Metatheory

mGL enjoys a rich metatheory. First, it satisfies cut elimination, for which we give the full proof. The proof of cut reduction requires a generalization of the graded cut rules to graded *multicut* rules in order to accommodate the structural rule of graded contraction.[1]

Thus, throughout the cut elimination proof we use the following graded multicut rules:

MCUT

$$\frac{\begin{array}{c}\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t_1 : X \\ (\delta_1, \delta, \delta_3) \odot (\Delta_1, x^n : X^n, \Delta_3) \vdash_{\mathsf{GS}} t_2 : Y\end{array}}{(\delta_1, (\delta \circledast [{\delta_2}^n]), \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3) \vdash_{\mathsf{GS}} [t_1, \ldots, t_1/x_1, \ldots, x_n]t_2 : Y}$$

GMCUT

$$\frac{\begin{array}{c}\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} t : X \\ (\delta_1, \delta, \delta_3) \odot (\Delta_1, X^n, \Delta_3); \Gamma \vdash_{\mathsf{MS}} l : B\end{array}}{(\delta_1, (\delta \circledast [{\delta_2}^n]), \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); \Gamma \vdash_{\mathsf{MS}} [t, \ldots, t/x_1, \ldots, x_n]l : B}$$

Both rules compute the contraction of the n hypotheses involved in the multicut on the cut-formula X. To do this we use *row-vector matrix multiplication*. We denote the matrix consisting of n-copies of the row vector δ_2 by $[\delta_2^n]$. Then row-vector multiplication is:

[1] Whilst cut reduction can be proved for intuitionistic sequent calculus without multicut [41], we use the standard multicut approach as it relates well to the categorical models developed later.

$$\delta \circledast [{\delta_2}^n] = \bigoplus_{k=1}^{n} (\delta(k) * \delta_2)$$

where the $*$ on the right is the scalar multiplication derived from the semiring, $\bigoplus$ is the pointwise addition of vectors, and where $\delta(k)$ is the k-th element of the vector δ. This computes the usages of the hypotheses in Δ_2 as the multiplication of a matrix of size $1 \times n$ with a matrix of size $n \times |\Delta_2|$ to yield a matrix of size $1 \times |\Delta_2|$.

We now proceed with the proof of cut elimination. The *rank* $\mathsf{Rank}(X)$ and $\mathsf{Rank}(A)$ of a formula is the height of the input formula's syntax tree where constants are of rank 0. The *cut rank* $\mathsf{CutRank}\,(\Pi)$ of a derivation Π of some judgment is defined to be one more than the maximum rank of the cut formula's in Π, and 0 if Π is cut free. The *depth* $\mathsf{Depth}(\Pi)$ of a derivation Π is the length of the longest path in the proof tree of Π, and hence, the depth of an axiom is 0. We prove cut-elimination without term annotations on the rules, in keeping with traditional proofs.

▶ **Lemma 9** (Cut Reduction for mGL).

1. *(Graded) If* Π_1 *is a proof of* $\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} X$ *and* Π_2 *is a proof of* $(\delta_1, \delta, \delta_3) \odot (\Delta_1, X^n, \Delta_3) \vdash_{\mathsf{GS}} Y$ *with* $\mathsf{CutRank}\,(\Pi_1), \mathsf{CutRank}\,(\Pi_2) \leq \mathsf{Rank}(X)$*, then there exists a proof* Π *of* $(\delta_1, \delta \circledast [{\delta_2}^n], \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3) \vdash_{\mathsf{GS}} Y$ *with* $\mathsf{CutRank}\,(\Pi) \leq \mathsf{Rank}(X)$.
2. *(Graded/Mixed) If* Π_1 *is a proof of* $\delta_2 \odot \Delta_2 \vdash_{\mathsf{GS}} X$ *and* Π_2 *is a proof of* $(\delta_1, \delta, \delta_3) \odot (\Delta_1, X^n, \Delta_3); \Gamma \vdash_{\mathsf{MS}} B$ *with* $\mathsf{CutRank}\,(\Pi_1), \mathsf{CutRank}\,(\Pi_2) \leq \mathsf{Rank}(X)$*, then there exists a proof* Π *of* $(\delta_1, \delta \circledast [{\delta_2}^n], \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); \Gamma \vdash_{\mathsf{MS}} B$ *with* $\mathsf{CutRank}\,(\Pi) \leq \mathsf{Rank}(X)$.
3. *(Mixed) If* Π_1 *is a proof of* $\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MS}} A$ *and* Π_2 *is a proof of* $\delta_1 \odot \Delta_1; (\Gamma_1, A, \Gamma_3) \vdash_{\mathsf{MS}} B$ *with* $\mathsf{CutRank}\,(\Pi_1), \mathsf{CutRank}\,(\Pi_2) \leq \mathsf{Rank}(A)$*, then there exists a proof* Π *of* $(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); (\Gamma_1, \Gamma_2, \Gamma_3) \vdash_{\mathsf{MS}} B$ *with* $\mathsf{CutRank}\,(\Pi) \leq \mathsf{Rank}(A)$.

Proof. By mutual induction on $\mathsf{Depth}(\Pi_1) + \mathsf{Depth}(\Pi_2)$ (see Appendix C.1 [40] for proof). ◀

▶ **Lemma 10** (Decreasing Order of mGL). *If* Π *is a proof of* $\delta \odot \Delta \vdash_{\mathsf{GS}} X$ *or* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$ *with* $\mathsf{CutRank}\,(\Pi) > 0$*, then there is a proof* Π' *of* $\delta' \odot \Delta \vdash_{\mathsf{GS}} X$ *or* $\delta' \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$ *with* $\delta \leq \delta'$ *and* $\mathsf{CutRank}\,(\Pi') < \mathsf{CutRank}\,(\Pi)$.

Proof. By induction on $\mathsf{Depth}(\Pi)$ (see Appendix C.2 [40] for proof). ◀

▶ **Theorem 11** (Cut Elimination of mGL). *If* Π *is a proof of* $\delta \odot \Delta \vdash_{\mathsf{GS}} X$ *or* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$ *with* $\mathsf{CutRank}\,(\Pi) > 0$*, then there is an algorithm which yields a cut-free proof* Π' *of* $\delta \odot \Delta \vdash_{\mathsf{GS}} X$ *or* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$ *respectively.*

Proof. Follows immediately by induction on $\mathsf{CutRank}\,(\Pi)$ and the previous lemma. ◀

▶ **Lemma 12** (Subformula property).

1. *(Graded) Every formula occurring in a cut-free proof* Π *of a judgment,* $\delta \odot \Delta \vdash_{\mathsf{GS}} X$*, consists of subformulas of the formulas occurring in* $\delta \odot \Delta \vdash_{\mathsf{GS}} X$.
2. *(Mixed) Every formula occurring in a cut-free proof* Π *of a judgment,* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$*, consists of subformulas of the formulas occurring in* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} A$.

Proof. By induction on Π (See Appendix C.3 [40] for proof). ◀

Lastly, we define an equational theory for mGL:

▶ **Definition 13** (Equational theory $\equiv$, subset). *An equational theory on derivations accounts for equalities between proofs of the same sequent arising from the graded structure (where the terms are the same but the structure of the proof tree differs), as well as cut elimination, i.e., in* GS*, if cut elimination on derivation* Π_1 *of* $\delta \odot \Delta \vdash_{\mathsf{GS}} t : X$ *yields the cut-free derivation of* Π_2 *for* $\delta \odot \Delta \vdash_{\mathsf{GS}} t' : X$ *then the equational theory has* $\Pi_1 \equiv \Pi_2$*, and similarly for* MS*.*

As a sample of two equations from the GS *fragment, the following shows an equation leveraging the commutativity of contraction, and another on the interaction between weakening and contraction leveraging the left-unit of semiring addition:*

$$\dfrac{\dfrac{\delta_1, r, \delta_2 \odot \Delta_1, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}{\delta_1, 0, r, \delta_2 \odot \Delta_1, X, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}\, \mathit{WEAK}_{\mathsf{GS}}}{\delta_1, 0 + r, \delta_2 \odot \Delta_1, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}\, \mathit{CONT}_{\mathsf{GS}} \equiv \delta_1, r, \delta_2 \odot \Delta_1, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y \quad (\textsc{contr-unitL})$$

$$\dfrac{\dfrac{\delta_1, r, s, \delta_2 \odot \Delta_1, X, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}{\delta_1, s, r, \delta_2 \odot \Delta_1, X, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}\, \mathit{EX}_{\mathsf{GS}}}{\delta_1, s + r, \delta_2 \odot \Delta_1, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}\, \mathit{CONT}_{\mathsf{GS}} \equiv \dfrac{\delta_1, r, s, \delta_2 \odot \Delta_1, X, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}{\delta_1, r + s, \delta_2 \odot \Delta_1, X, X, \Delta_2 \vdash_{\mathsf{GS}} t : Y}\, \mathit{CONT}_{\mathsf{GS}} \quad (\textsc{contr-sym})$$

Appendix A.1 [40] gives the full definition of the equational theory.

▶ Remark 14 ("$\beta\eta$-equalities" and "Triangle identities" via cut reduction). One might wonder where β-equalities are in the above equational theory, e.g., that $(\lambda x.l)l'$ in MS is equal to the cut $[l'/x]l$. Such β-equalities are provided by the cut elimination procedure, which reduces away interacting pairs of right and left formulas (the principal vs. principal cases).

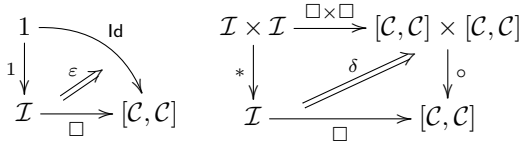
Similarly, η-equalities are equivalent to the identity expansion part of cut elimination procedure (where the cut of an identity axiom is transformed into an interacting left and right pair, with identity axioms expanded towards the leaves).

The internal derivations for the graded equivalent of the "triangle identities' (that one usually has associated with an adjunction) are also handled in the cut elimination procedure. The main feature of the derivations for both identities is that after one step the left and right rules for the modal operators match up. This leads to consecutive principal vs. principal cases where rules for the interacting left and right pairs in the two subproofs are removed by the reduction step.

3 Model

We detail a denotational model for mGL which is based on an adjoint decomposition of graded comonads. We introduce key definitions as needed.

A *graded comonad* can be summarised as a colax monoidal functor $\Box : \mathcal{I} \longrightarrow [\mathcal{C}, \mathcal{C}]$ where $\mathcal{I}$ is a preordered monoid $(\mathcal{I}, 1, *, \leq)$ treated as a monoidal category and $[\mathcal{C}, \mathcal{C}]$ is the category of endofunctors on $\mathcal{C}$ [33, 34]. Colax monoidality of $\Box$ means that the laws of a monoidal functor become 2-cells, providing the graded comonad operations:



which are thus natural transformations $\varepsilon_A : \Box_1 A \to A.$ and $\delta_{r,s,A} : \Box_{(r*s)} A \to \Box_r(\Box_s A)$.

Fujii et al. [10] gave a formal theory for graded monads, which can be easily dualised to graded comonads, showing that in an analogous way to an ordinary comonad, every graded comonad can be decomposed into an adjunction $\mathsf{Mny} \dashv \mathsf{Lin} : \mathcal{M} \longrightarrow \mathcal{C}$ and (key to *graded* comonads) a monoidal action $\odot : \mathcal{R} \times \mathcal{C} \longrightarrow \mathcal{C}$, and thus vice versa:

▶ **Lemma 15.** *(Resolution of a graded comonad [24, 10]) An adjunction* $\mathsf{L} \dashv \mathsf{R} : \mathcal{M} \longrightarrow \mathcal{C}$ *and a strict monoidal action* $\odot : \mathcal{R} \times \mathcal{C} \longrightarrow \mathcal{C}$ *together induce a graded comonad over the family of endofunctors defined by* $\Box_r = \mathsf{L}(r \odot (\mathsf{R}-)) : \mathcal{M} \longrightarrow \mathcal{M}$.

Along with some additional structure relating to substructurality (see below), this result provides a model of mGL with $\mathcal{C}$ providing a model for GS derivations, $\mathcal{M}$ providing a model for MS derivations, the type constructor Grd_r transporting from GS to MS modelled by $\mathsf{L}(r \odot -) : \mathcal{C} \to \mathcal{M}$, and type constructor Lin transporting MS to GS modelled by $\mathsf{R} : \mathcal{M} \to \mathcal{C}$.

However we need additional structure for the (sub)structural behaviour of our logic. In the literature on graded modal type theories, graded comonads are extended to *graded exponential comonads* (sometimes called *graded linear exponential comonads* [24]) defined as a colax monoidal functor $\Box : \mathcal{R} \longrightarrow [\mathcal{M}, \mathcal{M}]$ where $\mathcal{R}$ is a preordered semiring $(\mathcal{R}, 1, *, 0, +, \leq)$ (viewed as a category), $[\mathcal{M}, \mathcal{M}]$ is the category of symmetric lax monoidal endofunctors on a symmetric monoidal category $\mathcal{M}$, and $\Box$ has additional symmetric lax monoidal structure for the additional monoidality of $\mathcal{R}$ and $[\mathcal{M}, \mathcal{M}]$ [12]. This additional structure provides natural transformations $w_A : \Box_0 A \longrightarrow 1$ and $c_{r,s,A} : \Box_{(r+s)} A \longrightarrow (\Box_r A) \otimes (\Box_s A)$ capturing (graded) weakening and contraction, subject to comonoidal coherence conditions. This additional structure can be induced by the adjoint decomposition given an *exponential action*:

▶ **Definition 16** (Exponential action)**.** *Given a preordered semiring* $(\mathcal{R}, 1, *, 0, +, \leq)$ *and a symmetric monoidal category* $(\mathcal{C}, \mathsf{J}, \boxtimes)$*, we say that a bifunctor* $\odot : \mathcal{R} \times C \longrightarrow \mathcal{C}$ *is*

1. *a* strict action *(a* strict graded comonad*), if it satisfies the following equalities:*

$$\begin{aligned} \varepsilon_X &: \quad 1 \odot X = X \\ \delta_{X,r,s} &: \ (r * s) \odot X = r \odot (s \odot X) \end{aligned}$$

   Note that we treat these equalities as strict natural transformations named ε *and* δ*;*
2. symmetric lax monoidal *in the second argument if it has:*

$$\begin{aligned} m_{\mathsf{J},r} &: \quad \mathsf{J} \to r \odot \mathsf{J} \\ m_{\boxtimes,r,X,Y} &: \ (r \odot X) \boxtimes (r \odot Y) \to r \odot (X \boxtimes Y) \end{aligned}$$

   where m_{J} *is the unit of* $m_{\boxtimes}$ *and* $m_{\boxtimes}$ *is associative and commutative up to isomorphism;*
3. symmetric colax monoidal *between* $(\mathcal{R}, 0, +, \leq)$ *and* $(\mathcal{C}, \mathsf{J}, \boxtimes)$ *in the first argument if it has natural transformations:*

$$\begin{aligned} \mathsf{weak}_X &: \quad 0 \odot X \to \mathsf{J} \\ \mathsf{contr}_{r,s,X} &: (r + s) \odot X \to (r \odot X) \boxtimes (s \odot X) \end{aligned}$$

   where weak *is the unit of* contr*, e.g.* $\rho_{r \odot X} \circ (id \boxtimes \mathsf{weak}_X) \circ \mathsf{contr}_{r,0,X} = id$ *with right unitor* ρ*, and* contr *is associative and commutative, i.e., that* $\mathsf{contr}_{r,s,X} = c \circ \mathsf{contr}_{s,r,X}$*. Furthermore, these natural transformations must be preserved by the strict action and monoidal structure as described by the standard additional equations in Figure 2.*

If we have all of the above properties then we refer to $\odot$ as an *exponential action*. This terminology recalls the *exponential action* of Brunel et al. [6] which is the same as the above but where strictness is instead laxness in their definition. Our definition is also similar to linear exponential graded comonads (see e.g., [24, 12]), but here the graded comonad is uncurried (in the form of an action) and has equalities for its natural transformations (strictness).

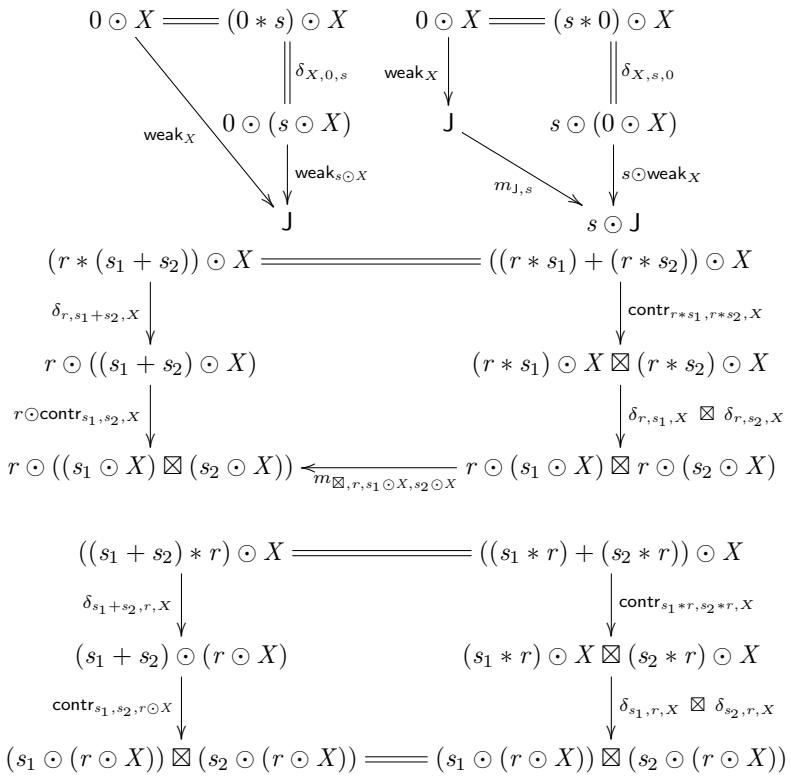


Figure 2 Further equations of a strict exponential action, interacting the colax symmetric monoidal structure, strict action, and (strict) monoidality.

We define a *strict exponential action* to be an exponential action as above but where the monoidal structure m_{J} and $m_{\boxtimes}$ is also strict, where for clarity (in the appendix) we sometimes orient the equality as a morphism, where in the opposite direction we denote these morphisms by $n_{\mathsf{J},r}$ and $n_{\boxtimes,r,X,Y}$ respectively. Strictness of the monoidal structure is needed for soundness of our model.

We now give the definition of the model of mGL, where we now use the opposite category $\mathcal{R}^{\mathsf{op}}$ to capture the correct polarity of the approximation rules.

▶ **Definition 17** (Mixed Graded/Linear model). *Suppose* $(\mathcal{C}, \mathsf{J}, \boxtimes)$ *and* $(\mathcal{M}, \mathsf{I}, \otimes)$ *are symmetric monoidal categories, where* $\mathcal{M}$ *is symmetric monoidal closed (with exponents* $\multimap$*), and* $(\mathcal{R}, 1, *, 0, +, \leq)$ *is a preordered semiring. Then a* Mixed Graded/Linear model *is a symmetric monoidal adjunction* $\mathsf{Mny} \dashv \mathsf{Lin} : \mathcal{M} \longrightarrow \mathcal{C}$ *along with an exponential action* $\odot : \mathcal{R}^{\mathsf{op}} \times \mathcal{C} \longrightarrow \mathcal{C}$.

Thus an mGL model is essentially an LNL model with a strict action. However, whilst Benton's LNL models are initially stated to require that $\mathcal{M}$ is Cartesian closed, he goes on to show that Cartesian properties are induced for the Eilenberg-Moore category of !-coalgebras for a symmetric monoidal category [3]. In our setting, the Cartesian structure is not needed since the MS logic is a mix of graded and linear logic, rather than Cartesian and linear logic. That is, graded propositions do not have arbitrary weakening and contraction, but instead these structural rules are controlled by grades (and corresponding underlying categorical structure [12, 24]). Therefore, a symmetric monoidal closed $\mathcal{M}$ suffices.

From Definition 17, we define our denotational model of mGL:

▶ **Definition 18** (Interpretation of Mixed Graded/Linear Logic.). *Given a Mixed Graded/Linear model (Def. 17) (with* $\mathsf{Mny} \dashv \mathsf{Lin} : \mathcal{M} \longrightarrow \mathcal{C}$ *and* $\odot : \mathcal{R}^{\mathsf{op}} \times \mathcal{C} \longrightarrow \mathcal{C}$*), we interpret by two mutually defined interpretations* $[\![-]\!]^{\mathsf{GS}}$ *and* $[\![-]\!]^{\mathsf{MS}}$ *on types and proofs (derivations):*

- *For every* GS *type* X *there is an object* $[\![X]\!]^{\mathsf{GS}} \in \mathcal{C}$ *and for every* MS *type* A *there is an object* $[\![A]\!]^{\mathsf{MS}} \in \mathcal{M}$*, mutually defined inductively as:*

$$\begin{array}{rlrl} [\![\mathsf{J}]\!]^{\mathsf{GS}} &= \mathsf{J} & [\![\mathsf{I}]\!]^{\mathsf{MS}} &= \mathsf{I} \\ [\![X \boxtimes Y]\!]^{\mathsf{GS}} &= [\![X]\!]^{\mathsf{GS}} \boxtimes [\![Y]\!]^{\mathsf{GS}} & [\![A \otimes B]\!]^{\mathsf{MS}} &= [\![A]\!]^{\mathsf{MS}} \otimes [\![B]\!]^{\mathsf{MS}} \\ [\![\mathsf{Lin}\ A]\!]^{\mathsf{GS}} &= \mathsf{Lin}[\![A]\!]^{\mathsf{MS}} & [\![A \multimap B]\!]^{\mathsf{MS}} &= [\![A]\!]^{\mathsf{MS}} \multimap [\![B]\!]^{\mathsf{MS}} \\ & & [\![\mathsf{Grd}_r\ X]\!]^{\mathsf{MS}} &= \mathsf{Mny}(r \odot [\![X]\!]^{\mathsf{GS}}) \end{array}$$

- *For every proof* Π *of a* GS *sequent* $(r_1, \ldots, r_n) \odot (x_1 : X_1, \ldots, x_n : X_n) \vdash_{\mathsf{GS}} t : X$ *there is a morphism in the category* $\mathcal{C}$*:*

$$[\![\Pi]\!]^{\mathsf{GS}} : (r_1 \odot [\![X_1]\!]^{\mathsf{GS}}) \boxtimes \ldots \boxtimes (r_n \odot [\![X_n]\!]^{\mathsf{GS}}) \longrightarrow [\![X]\!]$$

  (where an empty context is interpreted as $\emptyset^{\mathsf{GS}} = \mathsf{J}$*).*

- *For every proof* Π *of an* MS *sequent* $(r_1, \ldots, r_n) \odot (x_1 : X_1, \ldots, x_n : X_n); y_1 : A_1, \ldots, y_m : A_m \vdash_{\mathsf{MS}} l : B$ *there is a morphism in the category* $\mathcal{M}$*:*

$$[\![\Pi]\!]^{\mathsf{MS}} : \mathsf{Mny}(r_1 \odot [\![X_1]\!]^{\mathsf{GS}}) \otimes \ldots \otimes \mathsf{Mny}(r_n \odot [\![X_n]\!]^{\mathsf{GS}}) \otimes [\![A_1]\!]^{\mathsf{MS}} \otimes \ldots \otimes [\![A_m]\!]^{\mathsf{MS}} \longrightarrow [\![B]\!]^{\mathsf{MS}}$$

  (where an empty MS *context is interpreted as* $\emptyset^{\mathsf{MS}} = \mathsf{I}$*).*

Appendix C.4 [40] gives the full definition of the interpretation, including intermediate derivations from the mGL *model.*

Finally, we have our soundness and completeness theorems:

▶ **Theorem 19** (Soundness of Mixed Graded/Linear Logic models). *Suppose a mixed graded/-linear model as above. Then for derivation* Π_1 *of* $\delta \odot \Delta \vdash_{\mathsf{GS}} t_1 : X$ *and derivation* Π_2 *of* $\delta \odot \Delta \vdash_{\mathsf{GS}} t_2 : X$ *then if* $\Pi_1 \equiv \Pi_2$ *then* $[\![\Pi_1]\!] = [\![\Pi_2]\!]$*.*

Similarly for Π_1 *of* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l_1 : A$ *and derivation* Π_2 *of* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l_2 : A$ *then if* $\Pi_1 \equiv \Pi_2$ *then* $[\![\Pi_1]\!] = [\![\Pi_2]\!]$*.*

Proof. This proof holds by mutual induction. For the details see Appendix C.5 [40]. ◀

▶ **Theorem 20** (Completeness of Mixed Graded/Linear Logic models). *For derivations* Π_1*,* Π_2 *(of either* GS *or* MS*) if* $[\![\Pi_1]\!] = [\![\Pi_2]\!]$ *in all mixed graded/linear models, then* $\Pi_1 \equiv \Pi_2$*.*

Proof. This is a standard proof, where we build a generic model based on the syntax and the equational theory. For the details see Appendix C.6 [40]. ◀

4 Natural Deduction

We now develop a natural deduction formulation of mGL. Whilst sequent calculus judgments were denoted $\vdash_{\mathsf{MS}}$ and $\vdash_{\mathsf{GS}}$, natural deduction judgments are correspondingly $\vdash_{\mathsf{MT}}$ and $\vdash_{\mathsf{GT}}$.

The syntax for terms is identical to the sequent calculus, collected in Figure 1. Appendix B [40] gives the introduction and elimination rules and structural rules for mGL's natural deduction formulation. The unit constructors are j and i. Tensor products in both systems are denoted by pairs of terms with corresponding let-expressions for eliminators. The graded modal introduction form $\mathsf{Lin}\ l$ operates on mixed terms, dual to $\mathsf{Grd}\ r\ t$ which operates on graded terms. The mixed syntax includes abstraction $\lambda x.l$ and function application $l_1\ l_2$. The most interesting aspect is the rules for the modal operators:

$$\mathsf{Lin}_I\ \frac{\delta \odot \Delta; \emptyset \vdash_{\mathsf{MT}} l : B}{\delta \odot \Delta \vdash_{\mathsf{GT}} \mathsf{Lin}\, l : \mathsf{Lin}\, B} \qquad \mathsf{Lin}_E\ \frac{\delta \odot \Delta \vdash_{\mathsf{GT}} t : \mathsf{Lin}\, A}{\delta \odot \Delta; \emptyset \vdash_{\mathsf{MT}} \mathsf{Unlin}\, t : A}$$

$$\mathsf{Grd}_I\ \frac{\delta \odot \Delta \vdash_{\mathsf{GT}} t : X}{r * \delta \odot \Delta; \emptyset \vdash_{\mathsf{MT}} \mathsf{Grd}\, r\, t : \mathsf{Grd}_r\, X} \qquad \mathsf{Grd}_E\ \frac{\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MT}} l_1 : \mathsf{Grd}_r\, X \quad (\delta_1, r, \delta_3) \odot (\Delta_1, x : X, \Delta_3); \Gamma_1 \vdash_{\mathsf{MT}} l_2 : B}{(\delta_1, \delta_2, \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); (\Gamma_1, \Gamma_2) \vdash_{\mathsf{MT}} \mathsf{let}\, \mathsf{Grd}\, r\, x = l_1\, \mathsf{in}\, l_2 : B}$$

In the sequent calculus presented in Section 2, the right rule for Lin is in the graded subsystem, but the left rule is in the mixed subsystem. A similar idea arises here, the introduction rule for Lin (rule Lin_I) is in the graded subsystem and the elimination rule (rule Lin_E) is in the mixed subsystem. Introducing Grd_r formulas (rule Grd_I) has the effect of scaling the input grades by r. The elimination rule for Grd_r (rule Grd_E) is a pattern match on the form of l_1. Since Lin and Grd are the decomposition of graded modalities (Section 3), the form of the elimination rule for Grd_r is defined in a way which resembles that of elimination rules for graded modalities in other natural deduction-based type systems [31].

This formulation also has explicit graded structural rules:

$$\text{WEAK}\ \frac{(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2) \vdash_{\mathsf{GT}} t : Y}{(\delta_1, 0, \delta_2) \odot (\Delta_1, x : X, \Delta_2) \vdash_{\mathsf{GT}} t : Y} \qquad \text{CONT}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : X, \Delta_2) \vdash_{\mathsf{GT}} t : Y}{(\delta_1, r_1 + r_2, \delta_2) \odot (\Delta_1, x : X, \Delta_2) \vdash_{\mathsf{GT}} [x/y]t : Y}$$

$$\text{EX}\ \frac{(\delta_1, r_1, r_2, \delta_2) \odot (\Delta_1, x : X, y : Y, \Delta_2) \vdash_{\mathsf{GT}} t : Z}{(\delta_1, r_2, r_1, \delta_2) \odot (\Delta_1, y : Y, x : X, \Delta_2) \vdash_{\mathsf{GT}} t : Z}$$

In the transition from sequent calculus to natural deduction, left rules transform into elimination rules, and as a result the additional graded left rules in the mixed sequent calculus are no longer explicitly part of the system, but can be derived. We go on to prove that the sequent calculus of Section 2.1 is equivalent to the natural deduction system.

We give two main results related to the natural deduction system; the first is substitution for typing. Note that this reuses the row-vector multiplication operation of Section 2.2.

▶ **Lemma 21** (Substitution for $\vdash_{\mathsf{GT}}$ and $\vdash_{\mathsf{MT}}$). *The following hold by mutual induction:*

1. *(Graded) If* $\delta_2 \odot \Delta_2 \vdash_{\mathsf{GT}} t_1 : X$ *and* $(\delta_1, \delta, \delta_3) \odot (\Delta_1, x^n : X^n, \Delta_3) \vdash_{\mathsf{GT}} t_2 : Y$*, then* $(\delta_1, \delta \circledast [{\delta_2}^n], \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3) \vdash_{\mathsf{GT}} [t_1, \ldots, t_1/x_1, \ldots, x_n]t_2 : Y$.
2. *(Graded/Mixed) If* $\delta_2 \odot \Delta_2 \vdash_{\mathsf{GT}} t : X$ *and* $(\delta_1, \delta, \delta_3) \odot (\Delta_1, x^n : X^n, \Delta_3); \Gamma \vdash_{\mathsf{MT}} l : B$*, then* $(\delta_1, \delta \circledast [{\delta_2}^n], \delta_3) \odot (\Delta_1, \Delta_2, \Delta_3); \Gamma \vdash_{\mathsf{MT}} [t, \ldots, t/x_1, \ldots, x_n]l : B$.
3. *(Mixed) If* $\delta_2 \odot \Delta_2; \Gamma_2 \vdash_{\mathsf{MT}} l_1 : A$ *and* $\delta_1 \odot \Delta_1; (\Gamma_1, x : A, \Gamma_3) \vdash_{\mathsf{MT}} l_2 : B$*, then* $(\delta_1, \delta_2) \odot (\Delta_1, \Delta_2); (\Gamma_1, \Gamma_2, \Gamma_3) \vdash_{\mathsf{MT}} [l_1/x]l_2 : B$.

Proof. By mutual induction on the second assumed derivation (see Appendix C.7 [40]). ◀

Since we have an explicit structural rule for contraction (above and listed in Appendix B [40]), the substitution lemma on the graded fragment is formalized as multi-substitution. Otherwise, its proof is a fairly standard substitution proof for graded systems (e.g., as in [31]). Lastly, the natural deduction system is interderivable with the sequent calculus, which we establish such that the term witnessing the derivations does not change between systems:

▶ **Theorem 22** (Sequent calculus and natural deduction interderivability). $\delta \odot \Delta \vdash_{\mathsf{GS}} t : X \Leftrightarrow \delta \odot \Delta \vdash_{\mathsf{GT}} t : X$ *and* $\delta \odot \Delta; \Gamma \vdash_{\mathsf{MS}} l : A \Leftrightarrow \delta \odot \Delta; \Gamma \vdash_{\mathsf{MT}} l : A$.

Proof. By mutual induction on the assumed derivations (Appendix C.8 and C.9 [40]). The sequent calculus to natural deduction direction requires the substitution lemma above. ◀

The implication of the previous result is that we only need a semantic model of one of the two systems, and the other can be modelled using the same interpretation of terms. We chose to model the sequent calculus form directly.

5 Discussion

5.1 Relating linear base vs. graded base calculi

A major thread of graded type systems in the literature starts with a linear logic base and then generalises the ! modality to a semiring-graded modality atop a linear logic, e.g., the systems of Brunel et al. [6], Gaboardi et al. [12], Orchard et al. [31], and others [11, 18]. Often these systems are presented with a single context containing both linear and graded propositions [12, 31]. Overall, these approaches have a common core which is isomorphic to the natural deduction MT fragment shown here with the (natural deduction analogue of the) derived $\Box_r$ graded modality of Lemma 3 as part of their definition (i.e., not derived). We refer to this style of graded type system as the *linear base* style.

A contrasting approach has no base notion of linearity, but instead has pervasive grading tracking substructurality, i.e., no linear assumptions, every assumption has a grade, and function arrows come equipped with a grade describing the usage of their input in the function (e.g., written $A \xrightarrow{r} B$). Such systems include the coeffect calculi of Petricek et al. [33, 34], the general graded modal system of Bernardy et al. [1], and several others [2, 4, 7, 28, 30]. The GT fragment of our system here corresponds to a common subset of these approaches: a subset without function arrows and without a graded modality, since there is no graded modality that lives in the GT side ($\Box_i$ is derived into MT). Hughes et al. also develop a program synthesis technique for graded base systems, where grades are used to prune the search space [19]; its synthesis calculus formulation resembles closely GS.

Our work thus shows the relationship between the linear base and graded base style, namely that there is a mutual embedding between these two approaches which generates the graded modality in the linear base (Lemma 3). Exploring this in more depth is further work. For instance, it is unclear what is needed to realise a graded comonadic modality in GT that arises from the embedding (or a different embedding), and how this could interact with a graded function arrow in GS or GT. Pursuing this line of work would help to explain the relationship between the two dominant styles of graded system in the literature, which seem strongly related, and their relative expressive power. Nonetheless, by following Benton's programme and giving it a graded rendering here, we can already see here the close connection between these two styles of graded system.

5.2 Related work on adjoint logics

Pruiksma et al. formalized a general way to add and remove structural rules from a logic through adjunctions [37]. Their work is similar to ours as it relates logics through adjoint decompositions based on modal operators to control structural rules. Their formulation with "modes of truth" resembles our work with grades; however, modes of truth lack the algebraic properties graded formulations depend on and instead have a very relational flavor. Building on this work of Pruiksma et al., Jang et al. develop a natural deduction formulation

of adjoint logic [21]. They leverage this to give a functional language able to reason about resource properties like strictness and erasure. Similar reasoning can be developed ontop of our natural deduction formulation here, though this is left as further work.

A question is whether grading can be unified with the adjoint logic approach. Eades and Orchard sketched a unification based on generalising semiring operations to relations rather than functions, with predicates classifying unit values [20]. Hanukaev et al. develop this idea further, introducing a dependent type system based on a similar structure as the logics here but using a generalised notion of grading that combines the modes of adjoint logic [16]. They prove that their system is well-formed syntactically, but do not introduce any semantic model. Our logic mGL can be seen as an instantiation of their system, but the categorical model given here could potentially be generalised into a model of their system.

5.3 Further work

Practical implementation to leverage linear/grading separation

The separation of the mixed system (MS/MT) from the purely graded fragment (GS/GT) (which acts more as a standard intuitionistic system) can provide a basis for a programming language design. In such a language, the restrictions of linearity could be used only for handling data that needs to be linear, such as file handles or channels. However, for data types which need not be linear, e.g., primitive types like integers, characters, or structures over them, the graded fragment could be used without having to confront linearity constraints. The mutual embedding would allow the programmer to move smoothly between these two subcalculi. Similar ideas are discussed for the polarized extension of SILL for concurrent programming [35]. The implementation could borrow ideas from the Granule programming language, which already provides a mature and feature rich implementation of a linear-base style graded type system [31]. Instead, an mGL-inspired implementation could be based on the natural deduction term calculus with the modalities mediating between the two judgments. Exploring this application, perhaps as an extension to Granule, is future work.

Other generalisations

In LNL, the adjunction can be followed in the opposite direction to derive a monad $?A = \mathsf{Lin}(\mathsf{Mny}A)$. However, in mGL we do not get a graded monad by composing $\mathsf{Lin}\,(\mathsf{Grd}_r\,X)$ since the adjoint resolution of a graded monad has a strict action on the other side (on $\mathcal{M}$ in the model). Exploring a calculus with a pair of actions to allow both graded monads and graded comonads is further work.

Uniqueness typing

Recent work has demonstrated that *uniqueness* is a closely related but distinct concept to linearity [27]; uniqueness logic [17] is substructural in much the same way as linear logic, but provides a monadic modality for enabling the structural rules in contrast to linear logic's comonadic ! modality. Building an adjoint model for uniqueness or a calculus which unifies uniqueness and linearity [27] would be interesting further work, and this could potentially be extended to more recent systems which develop graded notions of uniqueness [26].

References

1 Andreas Abel and Jean-Philippe Bernardy. A unified view of modalities in type systems. *Proc. ACM Program. Lang.*, 4(ICFP):90:1–90:28, 2020. doi:10.1145/3408972.

2 Robert Atkey. Syntax and Semantics of Quantitative Type Theory. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 56–65. ACM, 2018. doi:10.1145/3209108.3209189.

3 P. N. Benton. A mixed linear and non-linear logic: Proofs, terms and models (extended abstract). In Leszek Pacholski and Jerzy Tiuryn, editors, *Computer Science Logic, 8th International Workshop, CSL '94, Kazimierz, Poland, September 25-30, 1994, Selected Papers*, volume 933 of *Lecture Notes in Computer Science*, pages 121–135. Springer, 1994. doi:10.1007/BFB0022251.

4 Jean-Philippe Bernardy, Mathieu Boespflug, Ryan R. Newton, Simon Peyton Jones, and Arnaud Spiwack. Linear Haskell: practical linearity in a higher-order polymorphic language. *Proc. ACM Program. Lang.*, 2(POPL):5:1–5:29, 2018. doi:10.1145/3158093.

5 Flavien Breuvart and Michele Pagani. Modelling coeffects in the relational semantics of linear logic. In *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, pages 567–581, 2015. doi:10.4230/LIPIcs.CSL.2015.567.

6 Aloïs Brunel, Marco Gaboardi, Damiano Mazza, and Steve Zdancewic. A core quantitative coeffect calculus. In Zhong Shao, editor, *Programming Languages and Systems - 23rd European Symposium on Programming, ESOP 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8410 of *Lecture Notes in Computer Science*, pages 351–370. Springer, 2014. doi:10.1007/978-3-642-54833-8_19.

7 Pritam Choudhury, Harley Eades III, Richard A. Eisenberg, and Stephanie Weirich. A graded dependent type system with a usage-aware semantics. *Proc. ACM Program. Lang.*, 5(POPL):1–32, 2021. doi:10.1145/3434331.

8 Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. The Structure of Exponentials: Uncovering the Dynamics of Linear Logic Proofs. In Georg Gottlob, Alexander Leitsch, and Daniele Mundici, editors, *Computational Logic and Proof Theory, Third Kurt Gödel Colloquium, KGC'93, Brno, Czech Republic, August 24-27, 1993, Proceedings*, volume 713 of *Lecture Notes in Computer Science*, pages 159–171. Springer, 1993. doi:10.1007/BFB0022564.

9 Loris D'Antoni, Marco Gaboardi, Emilio Jesús Gallego Arias, Andreas Haeberlen, and Benjamin C. Pierce. Sensitivity analysis using type-based constraints. In *Proceedings of the 1st Annual Workshop on Functional Programming Concepts in Domain-Specific Languages, FPCDSL@ICFP 2013, Boston, Massachusetts, USA, September 22, 2013*, pages 43–50, 2013. doi:10.1145/2505351.2505353.

10 Soichiro Fujii, Shin-ya Katsumata, and Paul-André Melliès. Towards a Formal Theory of Graded Monads. In Bart Jacobs and Christof Löding, editors, *Foundations of Software Science and Computation Structures - 19th International Conference, FOSSACS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9634 of *Lecture Notes in Computer Science*, pages 513–530. Springer, 2016. doi:10.1007/978-3-662-49630-5_30.

11 Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce. Linear dependent types for differential privacy. In *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013*, pages 357–370, 2013. doi:10.1145/2429069.2429113.

12 Marco Gaboardi, Shin-ya Katsumata, Dominic A. Orchard, Flavien Breuvart, and Tarmo Uustalu. Combining effects and coeffects via grading. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming, ICFP 2016, Nara, Japan, September 18-22, 2016*, pages 476–489, 2016. doi:10.1145/2951913.2951939.

13 Dan R. Ghica and Alex I. Smith. Bounded linear types in a resource semiring. In Zhong Shao, editor, *Programming Languages and Systems - 23rd European Symposium on Programming, ESOP 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8410 of *Lecture Notes in Computer Science*, pages 331–350. Springer, 2014. doi:10.1007/978-3-642-54833-8_18.

14 Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987. doi:10.1016/0304-3975(87)90045-4.

15 Jean-Yves Girard, Andre Scedrov, and Philip J. Scott. Bounded linear logic: A modular approach to polynomial-time computability. *Theor. Comput. Sci.*, 97(1):1–66, 1992. doi:10.1016/0304-3975(92)90386-T.

16 Peter Hanukaev and Harley Eades III. Combining dependency, grades, and adjoint logic. In *Proceedings of the 8th ACM SIGPLAN International Workshop on Type-Driven Development*, TyDe 2023, pages 58–70, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3609027.3609408.

17 Dana Harrington. Uniqueness logic. *Theor. Comput. Sci.*, 354(1):24–41, 2006. doi:10.1016/j.tcs.2005.11.006.

18 Jack Hughes, Danielle Marshall, James Wood, and Dominic Orchard. Linear Exponentials as Graded Modal Types. In *5th International Workshop on Trends in Linear Logic and Applications (TLLA 2021)*, Rome (virtual), Italy, June 2021. URL: https://hal-lirmm.ccsd.cnrs.fr/lirmm-03271465.

19 Jack Hughes and Dominic Orchard. Program synthesis from graded types. In Stephanie Weirich, editor, *Programming Languages and Systems - 33rd European Symposium on Programming, ESOP 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part I*, volume 14576 of *Lecture Notes in Computer Science*, pages 83–112. Springer, 2024. doi:10.1007/978-3-031-57262-3_4.

20 Harley Eades III and Dominic Orchard. Grading adjoint logic. *CoRR*, abs/2006.08854, 2020. arXiv:2006.08854.

21 Junyoung Jang, Sophia Roshal, Frank Pfenning, and Brigitte Pientka. Adjoint Natural Deduction. In Jakob Rehof, editor, *9th International Conference on Formal Structures for Computation and Deduction, FSCD 2024, July 10-13, 2024, Tallinn, Estonia*, volume 299 of *LIPIcs*, pages 15:1–15:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.FSCD.2024.15.

22 Max I. Kanovich, Stepan L. Kuznetsov, Vivek Nigam, and Andre Scedrov. Soft subexponentials and multiplexing. In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part I*, volume 12166 of *Lecture Notes in Computer Science*, pages 500–517. Springer, 2020. doi:10.1007/978-3-030-51074-9_29.

23 Shin-ya Katsumata. Parametric effect monads and semantics of effect systems. In Suresh Jagannathan and Peter Sewell, editors, *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pages 633–646. ACM, 2014. doi:10.1145/2535838.2535846.

24 Shin-ya Katsumata. A double category theoretic analysis of graded linear exponential comonads. In Christel Baier and Ugo Dal Lago, editors, *Foundations of Software Science and Computation Structures - 21st International Conference, FOSSACS 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings*, volume 10803 of *Lecture Notes in Computer Science*, pages 110–127. Springer, 2018. doi:10.1007/978-3-319-89366-2_6.

25 Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65(3):154–170, 1958.

26 Danielle Marshall and Dominic Orchard. Functional Ownership through Fractional Uniqueness. *Proc. ACM Program. Lang.*, 8(OOPSLA1):1040–1070, 2024. doi:10.1145/3649848.

27 Danielle Marshall, Michael Vollmer, and Dominic Orchard. Linearity and Uniqueness: An Entente Cordiale. In Ilya Sergey, editor, *Programming Languages and Systems - 31st European Symposium on Programming, ESOP 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*, volume 13240 of *Lecture Notes in Computer Science*, pages 346–375. Springer, 2022. doi:10.1007/978-3-030-99336-8_13.

28 Conor McBride. I Got Plenty o' Nuttin'. In Sam Lindley, Conor McBride, Philip W. Trinder, and Donald Sannella, editors, *A List of Successes That Can Change the World - Essays Dedicated to Philip Wadler on the Occasion of His 60th Birthday*, volume 9600 of *Lecture Notes in Computer Science*, pages 207–233. Springer, 2016. doi:10.1007/978-3-319-30936-1_12.

29 Paul-André Melliès. Categorical semantics of linear logic. In Pierre-Louis Curien, Hugo Herbelin, Jean-Louis Krivine, and Paul-André Melliès, editors, *Interactive Models of Computation and Program Behaviour*. Panoramas et Synthèses 27, Société Mathématique de France, 2009.

30 Benjamin Moon, Harley Eades III, and Dominic Orchard. Graded modal dependent type theory. In *Programming Languages and Systems - 30th European Symposium on Programming, ESOP 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings*, pages 462–490, 2021. doi:10.1007/978-3-030-72019-3_17.

31 Dominic Orchard, Vilem-Benjamin Liepelt, and Harley Eades III. Quantitative program reasoning with graded modal types. *Proc. ACM Program. Lang.*, 3(ICFP):110:1–110:30, 2019. doi:10.1145/3341714.

32 Dominic A. Orchard, Tomas Petricek, and Alan Mycroft. The semantic marriage of monads and effects. *CoRR*, abs/1401.5391, 2014. arXiv:1401.5391.

33 Tomas Petricek, Dominic A. Orchard, and Alan Mycroft. Coeffects: Unified static analysis of context-dependence. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, pages 385–397, 2013. doi:10.1007/978-3-642-39212-2_35.

34 Tomas Petricek, Dominic A. Orchard, and Alan Mycroft. Coeffects: a calculus of context-dependent computation. In *Proceedings of the 19th ACM SIGPLAN international conference on Functional programming, Gothenburg, Sweden, September 1-3, 2014*, pages 123–135, 2014. doi:10.1145/2628136.2628160.

35 Frank Pfenning and Dennis Griffith. Polarized substructural session types. In Andrew M. Pitts, editor, *Foundations of Software Science and Computation Structures - 18th International Conference, FoSSaCS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9034 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2015. doi:10.1007/978-3-662-46678-0_1.

36 Elaine Pimentel, Carlos Olarte, and Vivek Nigam. Process-as-formula interpretation: A substructural multimodal view (invited talk). In Naoki Kobayashi, editor, *6th International Conference on Formal Structures for Computation and Deduction, FSCD 2021, July 17-24, 2021, Buenos Aires, Argentina (Virtual Conference)*, volume 195 of *LIPIcs*, pages 3:1–3:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.FSCD.2021.3.

37 Klaas Pruiksma, William Chargin, Frank Pfenning, and Jason Reed. Adjoint logic. Unpublished Draft: http://www.cs.cmu.edu/~fp/papers/adjoint18b.pdf, 2018.

38 Klaas Pruiksma and Frank Pfenning. A message-passing interpretation of adjoint logic. *Journal of Logical and Algebraic Methods in Programming*, page 100637, 2020.

39 Greg Restall. *An introduction to substructural logics*. Routledge, 2002.

40 Victoria Vollmer, Danielle Marshall, Harley Eades III, and Dominic Orchard. A Mixed Linear and Graded Logic: Proofs, Terms, and Models. *CoRR*, abs/2401.17199, 2024. doi:10.48550/arXiv.2401.17199.

41 Jan von Plato. A proof of Gentzen's Hauptsatz without multicut. *Arch. Math. Log.*, 40(1):9–18, 2001. doi:10.1007/S001530050170.

42 Philip Wadler. Linear types can change the world! In Manfred Broy and Cliff B. Jones, editors, *Programming concepts and methods: Proceedings of the IFIP Working Group 2.2, 2.3 Working Conference on Programming Concepts and Methods, Sea of Galilee, Israel, 2-5 April, 1990*, page 561. North-Holland, 1990.

43 David Walker. Substructural type systems. *Advanced topics in types and programming languages*, pages 3–44, 2005.

Quantitative Graded Semantics and Spectra of Behavioural Metrics

Jonas Forster
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Lutz Schröder
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Paul Wild
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Harsh Beohar
University of Sheffield, UK

Sebastian Gurke
Universität Duisburg-Essen, Germany

Barbara König
Universität Duisburg-Essen, Germany

Karla Messing
Universität Duisburg-Essen, Germany

Abstract

Behavioural metrics provide a quantitative refinement of classical two-valued behavioural equivalences on systems with quantitative data, such as metric or probabilistic transition systems. In analogy to the linear-time/ branching-time spectrum of two-valued behavioural equivalences on transition systems, behavioural metrics vary in granularity, and are often characterized by fragments of suitable modal logics. In the latter respect, the quantitative case is, however, more involved than the two-valued one; in fact, we show that probabilistic metric trace distance cannot be characterized by any compositionally defined modal logic with unary modalities. We go on to provide a unifying treatment of spectra of behavioural metrics in the emerging framework of graded monads, working in coalgebraic generality, that is, parametrically in the system type. In the ensuing development of *quantitative graded semantics*, we introduce algebraic presentations of graded monads on the category of metric spaces. Moreover, we provide a general criterion for a given real-valued modal logic to characterize a given behavioural distance. As a case study, we apply this criterion to obtain a new characteristic modal logic for trace distance in fuzzy metric transition systems.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics

Keywords and phrases transition systems, modal logics, coalgebras, behavioural metrics

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.33

Related Version *Full Version*: `https://arxiv.org/abs/2306.01487` [23]

Funding The fourth author was supported by the EPSRC NIA Grant EP/X019373/1, while the remaining authors were supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 434050016 (SpeQt).

1 Introduction

While qualitative models of concurrent systems are traditionally analysed using various notions of two-valued process equivalence, it has long been recognized that for systems involving quantitative data, notions of behavioural *distance* play a useful role as a more fine-grained measure of process similarity. Well-known examples include behavioural distances

on probabilistic transition systems [25, 13, 46], on systems combining nondeterminism and probability [8], and on metric transition systems [11, 16]. Like in the two-valued case, where process equivalences of varying granularity are arranged on the *linear-time/branching-time spectrum* [47], one has a spectrum of behavioural metrics on a given system type that vary in granularity (with greater distances thought of as having finer granularity) [15].

An important point of interest in this context are *characteristic modal logics*. In the two-valued setting, a logic is *characteristic* for a given behavioural equivalence if the latter coincides with the respective induced logical indistinguishability relation, so that behavioural inequivalence can be certified by distinguishing formulae (as in the recent proof of the failure of unlinkability in the ICAO 9303 e-passport standard [17])). For instance, Hennessy-Milner logic is characteristic for bisimilarity [28], and most equivalences on the classical spectrum are characterized by fragments of Hennessy-Milner logic [47] that are compositionally defined, i.e. given by a choice of modalities and propositional operators equipped with a recursively defined semantics (e.g. trace equivalence is characterized by the logic built from diamonds, truth, and – optionally – disjunction). In the quantitative setting, a logic is *characteristic* if the induced logical distance coincides with the respective behavioural distance, so that high behavioural distance may be *certified* by means of distinguishing modal formulae [40]. A prototypical example is quantitative probabilistic modal logic, which is characteristic for branching-time behavioural distance on probabilistic transition systems [46]. However, it turns out that in general, the quantitative setting behaves less smoothly in this respect than the two-valued setting. Indeed, we show as our first main result that for probabilistic metric trace distance (on generative probabilistic transition systems in which the set of labels is equipped with a metric, i.e. on the probabilistic variant of metric transition systems), there does not exist any characteristic quantitative modal logic at all. Here, the term *modal logic* is understood in a fairly broad sense; essentially, we stipulate no more than that, in analogy to the two-valued case as discussed above, the logic should be a compositionally defined fragment of a bisimulation-invariant next-step logic with unary modalities.

We subsequently work towards positive results, using the framework of *graded semantics* [37, 14] to achieve an appropriate level of generality. Graded semantics is parametric both in the *type* of systems (e.g. probabilistic, metric, fuzzy) and in the quantitative *semantics* of systems, i.e. the choice of behavioural distance. The system type is abstracted as an endofunctor on a suitable base category following the paradigm of *universal coalgebra* [43]. Parametricity in the system semantics, on the other hand, is based on the choice of a *graded monad*, which handles additional semantic identifications (beyond branching-time equivalence) by algebraic means, using grades to control the depth of look-ahead. Both Kleisli-style coalgebraic trace semantics [27] and the smoother, but less widely applicable Eilenberg-Moore-style coalgebraic trace semantics [29] are subsumed by this framework [37].

Graded semantics has recently been extended to cover behavioural distances in the Eilenberg-Moore-style setting [5, 24], and, generalizing the two-valued case [37, 14], a canonical notion of *quantitative graded logic* has been identified. Quantitative graded logics are always *invariant* under the underlying behavioural distance in the sense that formula evaluation is nonexpansive, so that logical distance is below behavioural distance. In some cases, the reverse inequality, i.e. *expressivity* of quantitative graded logics, can be established by a straightforward generalization of corresponding criteria for the two-valued case. Notably, one can show that in the Eilenberg-Moore setting, one essentially always has a characteristic modal logic [24], in sharp contrast to our present negative result. The flip side of the coin is that Eilenberg-Moore style trace semantics applies to only rather few system types (essentially automata with effects), and in particular does not support a metric on the labels as found, for instance, in standard metric transition systems.

Our second, now positive, contribution in the present work is to extend the framework to unrestricted graded semantics, notably including Kleisli-style coalgebraic trace semantics and, hence, trace semantics on systems with labels taken from a metric space. For the syntactic treatment of spectra of behavioural distances in this sense, we introduce a graded extension of *quantitative algebra* [36] that allows describing graded monads on the category of metric spaces by operations and approximate equations. As suggested by our negative result, establishing expressivity of graded logics in the general case presents additional challenges compared to the two-valued variant and the Eilenberg-Moore case. In particular, it turns out that the expressivity criterion needs to be parametric in a strengthening of the inductive hypothesis in the induction on depth of look-ahead that it encapsulates; indeed, this happens already in strikingly simple cases such as metric streams. We develop a number of example applications: We recover results on expressivity of quantitative modal logics for (finite-depth) branching-time distances [33, 48, 19, 32], as well as a recent result on expressivity of a quantitative modal logic for trace distance in metric transition systems [4], which in fact we generalize to systems with metric state space and closed branching. In a concluding case study, we moreover identify a new characteristic modal logic for trace distance on fuzzy metric transition systems, which turns out to require next-step modalities incorporating a constant shift on label distances.

Omitted proofs and additional details can be found in the full version [23].

Related Work. We have mentioned previous work on coalgebraic branching-time behavioural distances [2, 33, 22, 49, 50, 4, 32] and on graded semantics for two-valued behavioural equivalences and preorders [37, 14, 19]. Kupke and Rot [34] study logics for *coinductive predicates*, which generalize branching-time behavioural distances. Generally, our overall setup differs from the one used in [34] and elsewhere by working with coalgebras that already live on metric spaces (e.g. [42, 53, 46, 22, 26]); this allows covering functors on metric spaces that are not liftings of set functors, such as the full Hausdorff functor (which takes closed subsets). Recent work on Galois connections for logical distances [4, 5] is highly general (and in fact not even tied to state-based systems) but leaves more work to the instantiation than the framework of graded monads. Moreover, it is aimed primarily at fixpoint characterizations of logical distance, and in fact induces behavioural distance from the logic, while we aim to provide logical characterizations of *given* behavioural distances. Alternative coalgebraic approaches to process equivalences coarser than branching time include coalgebraic trace semantics in Kleisli [27] and Eilenberg-Moore categories [29], which are both subsumed by the paradigm of graded monads [37], as well as an approach in which behavioural equivalences are *defined* via characteristic logics [31]. The Eilenberg-Moore and Kleisli setups can be unified using corecursive algebras, which also support, under certain assumptions, a logical characterization for these cases [41]. The Eilenberg-Moore approach has been applied to linear-time behavioural distances [2]. Recently, some of the present authors used the graded-semantics approach to Eilenberg-Moore semantics to extract characteristic logics that factor through the determinization of a coalgebra [24]. We make use of their notion of *graded logic* and complement their work by considering unrestricted graded semantics, in particular covering the more broadly applicable Kleisli-style semantics.

De Alfaro et al. [11] introduce a linear-time logic for (state-labelled) metric transition systems. The semantics of this logic is defined by first computing the set of paths of a system, so that propositional operators and modalities have a different meaning than in corresponding branching-time logics, while our graded logics are fragments of branching-time logics. Fahrenberg et al. [16] present a game-based approach to a spectrum of behavioural

distances on metric transition systems. A two-valued logic for probabilistic trace semantics (for a discrete set of labels) has been considered in the context of differential privacy [7]. A notion of logical distance is then obtained via a real-valued semantics defined using a syntactic distance on formulae; this semantics is not compositional (truth values are defined by taking infima over the whole logical syntax), so subsequent results relating this logical distance to notions of weak anonymity do not contradict our impossibility result on (compositional) characteristic logics for probabilistic trace semantics.

2 Preliminaries

Basic familiarity with category theory is assumed (e.g. [1]). We write **Set** for the category of sets and maps. Below, we recall some background on (bounded) metric spaces and universal coalgebra.

Metric spaces. The real unit interval $[0,1]$ will serve as the domain of distances and truth values. Under the usual ordering $\leq$, $[0,1]$ forms a complete lattice; we write $\bigvee, \bigwedge$ for joins and meets in $[0,1]$ (e.g. $\bigvee_i x_i = \sup_i x_i$), and $\vee, \wedge$ for binary join and meet, respectively. We denote truncated addition and subtraction by $\oplus$ and $\ominus$, respectively; that is, $x \oplus y = \min(x+y, 1)$ and $x \ominus y = \max(x-y, 0)$. These operations form part of a structure of $[0,1]$ as a (co-)quantale; for readability, we refrain from working with more general quantales [49, 22].

▶ **Definition 2.1.** A (bounded) *pseudometric space* is a pair (X,d) consisting of a set X and a function $d\colon X \times X \to [0,1]$ satisfying the standard conditions of *reflexivity* ($d(x,x) = 0$ for all $x \in X$), *symmetry* ($d(x,y) = d(y,x)$ for all $x, y \in X$), and the *triangle inequality* ($d(x,z) \leq d(x,y) + d(y,z)$ for all $x, y, z \in X$); if additionally *separation* holds (for $x, y \in X$, if $d(x,y) = 0$ then $x = y$), then (X,d) is a *metric space*. A function $f\colon X \to Y$ between pseudometric spaces (X, d_X) and (Y, d_Y) is *nonexpansive* if $d_Y(f(x), f(y)) \leq d_X(x,y)$ for all $x, y \in X$. Metric spaces and nonexpansive maps form a category **Met**.

We often do not distinguish notationally between a (pseudo-)metric space (X,d) and its underlying set X. Occasionally we use subscripts to make explicit the carrier to which a (pseudo-)metric is associated, i.e. d_X is the (pseudo-)metric of the space with carrier X. The categorical product $(X, d_X) \times (Y, d_Y)$ of (pseudo-)metric spaces equips the Cartesian product $X \times Y$ with the supremum (pseudo-)metric $d_{X\times Y}((a,b),(a',b')) = d_X(a,a') \vee d_Y(b,b')$. Similarly, the *Manhattan tensor* $\boxplus$ equips $X \times Y$ with the *Manhattan (pseudo-)metric* $d_{X \boxplus Y}((a,b),(a',b')) = d_X(a,a') \oplus d_Y(b,b')$. We occasionally write elements of the product X^{n+m} as vw if $v \in X^n$ and $w \in X^m$. Given (pseudo-)metric spaces X, Y, the nonexpansive functions $X \to Y$ form a (pseudo-)metric space under the standard supremum distance.

▶ **Example 2.2.** We recall some key examples of functors on **Set** and **Met**.

1. We write $\mathcal{P}_\omega$ for the finite powerset functor on **Set**, and $\overline{\mathcal{P}}_\omega$ for the lifting of $\mathcal{P}_\omega$ to **Met** given by the Hausdorff metric. Explicitly, for a metric space (X,d) and $A, B \in \mathcal{P}_\omega X$,
$$d_{\overline{\mathcal{P}}_\omega X}(A,B) = \left(\bigvee_{a\in A} \bigwedge_{b\in B} d(a,b)\right) \vee \left(\bigvee_{b\in B} \bigwedge_{a\in A} d(a,b)\right). \tag{2.1}$$
Both $\mathcal{P}_\omega$ and $\overline{\mathcal{P}}_\omega$ are monads, with multiplication taking big unions.
2. Related to the above, the closed Hausdorff monad $\mathcal{P}_c$ on **Met** sends a metric space X to the set of closed subsets of X, again equipped with the Hausdorff metric. For a nonexpansive function $f\colon X \to Y$, $\mathcal{P}_c f$ sends $A \in \mathcal{P}_c X$ to the closure of $f[A]$. Monad multiplication takes the closure of the big union.

3. Similarly, $\mathcal{D}_\omega$ denotes the functor on **Set** that maps a set X to the set of finitely supported probability distributions on X, and $\overline{\mathcal{D}}_\omega$ denotes the lifting of $\mathcal{D}_\omega$ to **Met** that equips $\mathcal{D}_\omega X$ with the Kantorovich metric. Explicitly, for a metric space (X,d) and $\mu,\nu\in\mathcal{D}_\omega X$,

$$d_{\overline{\mathcal{D}}_\omega X}(\mu,\nu)=\bigvee_f \sum_{x\in X} f(x)(\mu(x)\ominus\nu(x))$$

where f ranges over all nonexpansive functions $X\to[0,1]$. We often write elements of $\overline{\mathcal{D}}_\omega X$ as finite formal sums $\sum p_i\cdot x_i$, with $x_i\in X$ and $\sum p_i=1$.

4. The *finite fuzzy powerset* functor $\mathcal{F}_\omega$ is given on sets X by $\mathcal{F}_\omega X=\{A\colon X\to[0,1]\mid A(x)=0 \text{ for almost all } x\in X\}$, and on maps $f\colon X\to Y$ by $\mathcal{F}_\omega f(A)(y)=\bigvee\{A(x)\mid f(x)=y\}$ for $A\in\mathcal{F}_\omega X$. That is, $\mathcal{F}_\omega X$ consists of the finite fuzzy subsets of X, given by assigning membership degrees in $[0,1]$ to elements of X, and $\mathcal{F}_\omega f$ acts by taking fuzzy direct images. We lift $\mathcal{F}_\omega$ to a functor $\overline{\mathcal{F}}_\omega$ on metric spaces that equips $\mathcal{F}_\omega X$ with the fuzzy Hausdorff distance [49, Example 5.3.1]. Explicitly, $d_{\overline{\mathcal{F}}_\omega X}(A,B)=d_0(A,B)\vee d_0(B,A)$ for a metric space (X,d) and $A,B\in\mathcal{F}_\omega X$, where

$$d_0(A,B)=\bigvee_x\bigwedge_y(A(x)\ominus B(y))\vee(A(x)\wedge d(x,y)).$$

Thus, $d_0(A,B)$ is analogous to the left-hand term in the binary join defining the Hausdorff metric (2.1): Both terms can be read intuitively as "B is far from A if there is x such that $x\in A$ and for all y, if $y\in B$ then y is far from x", where $d_0(A,B)$ takes into account that the sets A,B are fuzzy (in particular, the "if $y\in B$" is reflected in the contribution of $B(y)$ being negative).

Coalgebra. *Universal coalgebra* [43] has established itself as a way to reason about state-based systems at an appropriate level of abstraction. It is based on encapsulating the transition type of systems as an endofunctor $G\colon\mathcal{C}\to\mathcal{C}$ on a base category $\mathcal{C}$. Then, a G-coalgebra (X,γ) consists of a $\mathcal{C}$-object X, thought of as an object of *states*, and a morphism $\gamma\colon X\to GX$, thought of as assigning to each state a collection of successors, structured according to G. A $\mathcal{C}$-morphism $h\colon X\to Y$ is a morphism of G-coalgebras $(X,\gamma)\to(Y,\delta)$ if $Gh\cdot\gamma=\delta\cdot h$.

For a functor $G\colon\mathbf{Met}\to\mathbf{Met}$, one has a canonical notion of *branching-time behavioural distance* d_γ^G on a G-coalgebra (X,γ) [22]. In case G is a lifting of a set functor (which means roughly that the underlying set of GX is independent of the metric on X), the general definition simplifies as follows: d_γ^G is the least fixpoint of the map $d\mapsto d_{G(X,d)}\circ(\gamma\times\gamma)$ [2, 22].

▶ **Example 2.3.** Throughout the paper, we *fix a metric space* $\mathcal{A}$ of *labels*. Finitely branching metric transition systems with transition labels in $\mathcal{A}$ are coalgebras for the functor $\overline{\mathcal{P}}_\omega(\mathcal{A}\times-)$. (More precisely, a metric transition system is usually assumed to have a set as its state space, while $\overline{\mathcal{P}}_\omega(\mathcal{A}\times-)$-coalgebras more generally have a metric space of states, subsuming mere sets of states as discrete metric spaces). Similarly, coalgebras for the functor $\mathcal{P}_c(\mathcal{A}\times-)$ are *closed-branching* metric transition systems, where sets of successors can be infinite but are required to be closed. With few exceptions (e.g. [22]), most coalgebraic approaches to behavioural metrics (e.g. [2, 33, 50, 34]) rely on the functor being a lifting of a **Set**-functor. We work with unrestricted functors on **Met**, thus, e.g., covering the above-mentioned functor $\mathcal{P}_c(\mathcal{A}\times-)$, which is not a lifting of a set functor. We use trace semantics on metric labelled transition systems (both finitely branching and closed-branching) as a running example of concepts as they appear throughout the text.

Quantitative Coalgebraic Modal Logic. We proceed to introduce the requisite notion of quantitative coalgebraic modal logic [45, 33, 50, 24], in a formulation geared towards easing the extraction of invariant fragments for various semantics [24], and instantiated to the category of metric spaces. The notion of quantitative coalgebraic modal logic will also serve as the yardstick for our negative result on characteristic modal logics for probabilistic metric trace semantics (Section 3).

Syntactically, a *modal logic* is a triple $\mathcal{L} = (\Theta, \mathcal{O}, \Lambda)$ where Θ is a set of truth constants, $\mathcal{O}$ is a set of propositional operators, each with associated finite arity, and Λ is a set of modal operators, also each with an associated finite arity. For readability, we restrict to unary modal operators; extending our positive results to modal operators of higher arity is simply a matter of adding indices. The set of *formulae* of $\mathcal{L}$ is then given by the grammar

$$\phi ::= c \mid p(\phi_1, \dots, \phi_n) \mid L\phi \qquad (c \in \Theta,\ p \in \mathcal{O}\ n\text{-ary},\ L \in \Lambda).$$

Formulae are interpreted in G-coalgebras for a given functor $G\colon \mathbf{Met} \to \mathbf{Met}$, and take values in the truth value object $\Omega = [0,1]$, which we equip with the standard metric $d_\Omega(x,y) = |x-y|$. Moreover, the semantics is parametric in the following components:

- For every $c \in \Theta$, a nonexpansive map $\hat{c}\colon 1 \to \Omega$.
- For $p \in \mathcal{O}$ with arity n, a nonexpansive map $[\![p]\!]\colon \Omega^n \to \Omega$
- For $L \in \Lambda$, a nonexpansive map $[\![L]\!]\colon G\Omega \to \Omega$

The evaluation of a formula ϕ on a G-coalgebra (X, γ) is then a nonexpansive map $[\![\phi]\!]_\gamma\colon X \to \Omega$, inductively defined by

$$[\![c]\!]_\gamma = (X \xrightarrow{!} 1 \xrightarrow{\hat{c}} \Omega) \qquad [\![p(\phi_1, \dots, \phi_n)]\!]_\gamma = (X \xrightarrow{\langle [\![\phi_1]\!]_\gamma, \dots, [\![\phi_n]\!]_\gamma \rangle} \Omega^n \xrightarrow{[\![p]\!]} \Omega)$$

$$[\![L\phi]\!]_\gamma = (X \xrightarrow{\gamma} GX \xrightarrow{G[\![\phi]\!]_\gamma} G\Omega \xrightarrow{[\![L]\!]} \Omega)$$

▶ **Example 2.4.** We briefly exemplify the semantics of modalities: Take the functor $G = \overline{\mathcal{P}}_\omega(\mathcal{A} \times (-))$ modelling metric transition systems (Example 2.3), and define the interpretation $[\![\Diamond_a]\!]\colon \overline{\mathcal{P}}_\omega(\mathcal{A} \times \Omega) \to \Omega$ of modalities $\Diamond_a$, for $a \in \mathcal{A}$, by $[\![\Diamond_a]\!](U) = \bigvee_{(b,v)\in U}(1 - d(a,b)) \wedge v$. Then, roughly speaking, the degree to which a state in a metric transition system satisfies a formula $\Diamond_a\phi$ is the degree to which it has a b-successor that satisfies ϕ, for some b that is close to a. (The use of $1 - d(a,b)$ is owed to the usual discrepancy between 1 representing "true" but also "far apart".)

In the framework defined so far, truth constants are interchangeable with nullary propositional operators, but in the setting of graded logics (Section 6), the two concepts will play syntactically and semantically distinct roles. In particular, invariance w.r.t. a target semantics (Theorem 6.6) will in general hold only for formulae of *uniform depth*, that is, formulae in which all occurrences of truth constants are nested under the same number of modal operators. In cases where there are no truth constants, all formulae are uniform. We write $\mathcal{L}_{\mathsf{unif}}$ for the set of uniform-depth $\mathcal{L}$-formulae.

▶ **Definition 2.5.** *Logical distance* under the logic $\mathcal{L}$ on a G-coalgebra (X, γ) is the pseudometric $d^{\mathcal{L}}$ given by $d^{\mathcal{L}}(x,y) = \bigvee\{d_\Omega([\![\phi]\!]_\gamma(x), [\![\phi]\!]_\gamma(y)) \mid \phi \in \mathcal{L}_{\mathsf{unif}}\}$.

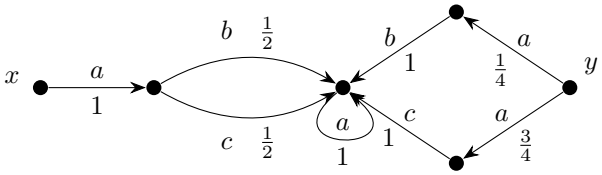
Logical distance is always a lower bound for branching-time behavioural distance [33, 50, 22]; we discuss details in Remark 7.10.

3 Probabilistic Metric Trace Semantics

Finitely branching *probabilistic metric transition systems* over a metric space of transition labels $\mathcal{A}$ are coalgebras for the functor $G^{\mathsf{prob}} = \overline{\mathcal{D}}_\omega(\mathcal{A} \boxplus (-))$ (cf. Examples 2.2 and 2.3). The *probabilistic (metric) trace semantics* [9] of a probabilistic transition system calculates, at each depth n, a distribution over length-n traces. One then obtains a notion of *depth-n probabilistic trace distance* d_n^{ptrace}, which takes Kantorovich distances of depth-n trace distributions under the Manhattan distance on traces. Formal definitions are as follows.

▶ **Definition 3.1.** We write $\mathcal{A}^{\boxplus n}$ for the n-fold Manhattan tensor $\mathcal{A} \boxplus \cdots \boxplus \mathcal{A}$. Let (X, γ) be a G^{prob}-coalgebra. For each $x \in X$, the *depth-n trace distribution* $\mu_x^n \in \overline{\mathcal{D}}_\omega(\mathcal{A}^{\boxplus n})$ is inductively defined as $\mu_x^{n+1}(aw) = \sum_{y \in X} \gamma(x)(a, y)\mu_y^n(w)$ for $a \in \mathcal{A}$ and $w \in \mathcal{A}^n$, with $\mu_x^0 \in \overline{\mathcal{D}}_\omega(\mathcal{A}^{\boxplus 0}) \cong \overline{\mathcal{D}}_\omega(1)$ being the unique distribution on the singleton set 1. The *probabilistic trace distance* on X is $d^{\mathsf{ptrace}} = \bigvee_{n<\omega} d_n^{\mathsf{ptrace}}$, where $d_n^{\mathsf{ptrace}}(x, y) = d_{\overline{\mathcal{D}}_\omega(\mathcal{A}^{\boxplus n})}(\mu_x^n, \mu_y^n)$.

Consider the following concrete example, where we assume that $d(b, c) = 0.5$.



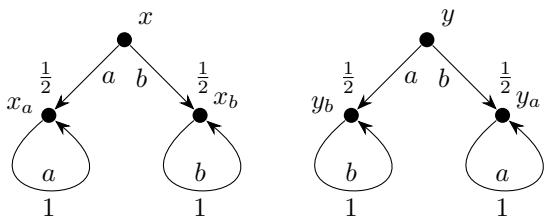
For $n \geq 2$ we then have by the above definition that $\mu_x^n = \frac{1}{2}aba^{n-2} + \frac{1}{2}aca^{n-2}$ while $\mu_y^n = \frac{1}{4}aba^{n-2} + \frac{3}{4}aca^{n-2}$. The distance of the two relevant traces is given by $d(aba^{n-2}, aca^{n-2}) = d(a, a) \oplus d(b, c) \oplus d(a, a) \oplus \ldots \oplus d(a, a) = 0.5$. Calculating the Kantorovich distance of trace distributions then gives us that $d(\mu_x^n, \mu_y^n) = \frac{1}{4}d(aba^{n-2}, aca^{n-2}) = 0.125$, and by extension $d^{\mathsf{ptrace}}(x, y) = 0.125$.

One would now like to have a logic that characterizes the trace distance d^{ptrace}. However, we establish the following impossibility result instead:

▶ **Theorem 3.2.** *Let $\mathcal{L} = (\Theta, \mathcal{O}, \Lambda)$ be a coalgebraic modal logic with unary modalities for the functor G^{prob}, over a non-discrete metric space $\mathcal{A}$ of labels. Then $d^{\mathcal{L}} \neq d^{\mathsf{ptrace}}$.*

In other words, no quantitative coalgebraic modal logic with unary modalities has a compositionally defined fragment that characterizes probabilistic metric trace distance. The restriction to coalgebraic modal logics effectively means only that modal logics should be invariant under the standard branching-time semantics and have only next-step modalities [39, 44]. Theorem 3.2 implies in particular that the logic featuring modalities $\Diamond_a$ for $a \in \mathcal{A}$, with $\Diamond_a\phi$ being the expected truth value of ϕ restricted to a-successors, fails to characterize probabilistic metric trace distance (even though it characterizes two-valued probabilistic trace *equivalence* [6, 14]). In fact, it can even be shown that giving up the requirement of interpretations of modalities being nonexpansive does not help.

Proof sketch (Theorem 3.2). Suppose that $\mathcal{L}$ is invariant under probabilistic metric trace semantics ($d^{\mathcal{L}} \leq d^{\mathsf{ptrace}}$); we show that $\mathcal{L}$ fails to be expressive ($d^{\mathcal{L}} \not\geq d^{\mathsf{ptrace}}$). As an intermediate step, we show that invariance under probabilistic metric trace semantics implies that modal operators are affine maps. Then calculation shows that affine modalities are unable to distinguish the states x and y in the following system, where $d(a, b) = v < 1$, to a degree greater than v^2, even though the behavioural distance of x and y under probabilistic trace semantics is v.



◀

We leave the question of whether a characteristic logic with higher-arity modalities exists as an open problem.

While expressive quantitative coalgebraic logics for branching-time semantics exist for a wide variety of systems [33, 50, 22, 26], this is thus apparently not always the case for linear-time semantics. The no-go result above emphasizes the challenges of the quantitative setting and the need for a theory of quantitative coalgebraic logics beyond branching time. In the following, we will address precisely this problem, by adopting techniques from the theory of graded semantics and highlighting issues unique to the metric setting.

4 Graded Monads and Graded Algebras

The framework of *graded semantics* [14, 37] is based on the central notion of *graded monads*, which algebraically describe the structure of observable behaviours, in particular identifications beyond branching time, at each finite depth. Here, *depth* is understood as look-ahead, measured in terms of the number of transition steps.

▶ **Definition 4.1.** A *graded monad* $\mathbb{M} = ((M_n)_{n\in\mathbb{N}}, \eta, (\mu^{n,k})_{n,k\in\mathbb{N}})$ on a category $\mathcal{C}$ consists of a family of functors $M_n\colon \mathcal{C} \to \mathcal{C}$ for $n \in \mathbb{N}$ and natural transformations $\eta\colon Id \to M_0$ (the *unit*) and $\mu^{n,k}\colon M_nM_k \to M_{n+k}$ for all $n, k \in \mathbb{N}$ (the *multiplications*), subject to essentially the same laws as ordinary monads up to the insertion of grades; specifically, one has *unit laws* $\mu^{0,n} \cdot \eta M_n = id_{M_n} = \mu^{n,0} \cdot M_n\eta$ and an *associative law* $\mu^{n+k,m} \cdot \mu^{n,k}M_m = \mu^{n,k+m} \cdot M_n\mu^{k,m}$.

In particular, (M_0, η, μ^{00}) is an ordinary (non-graded) monad.

The understanding of the data constituting a graded monad is similar as for plain monads: Roughly speaking (this will be made more precise in Section 5), M_nX may be thought of as a space of terms of depth n, modulo given identities, over variables from X; μ^{nk} substitutes depth-k terms into a depth-n term, obtaining a depth-$(n + k)$ term; and η converts variables into terms of depth 0.

▶ **Example 4.2.** We discuss graded monads modelling the linear-time end of the spectrum, noting that graded monads cover also branching-time (Remark 7.10) and intermediate semantics, involving simulation, readiness, failures etc. [14]. A *Kleisli distributive law* is a natural transformation $\lambda\colon FT \to TF$ where F is a functor and T a monad, subject to coherence with the monad structure [27]. This yields a graded monad with $M_n = TF^n$ [37]; here, T may be understood as defining the branching type of the system, and F as defining a type of accepted structure. We will use the following instance of this construction as a running example: Take $F = \mathcal{A} \times (-)$ and $T = \overline{\mathcal{P}}_\omega$ or $T = \mathcal{P}_c$ (corresponding to nondeterministic branching). Then $\lambda(a, U) = \{(a, x) \mid x \in U\}$ defines a distributive law $\lambda\colon \mathcal{A} \times T(-) \to T(\mathcal{A} \times (-))$ (in particular, λ is nonexpansive). We obtain the *graded metric trace monads* $M_n = T(\mathcal{A}^n \times (-))$.

Graded monads come with a graded analogue of Eilenberg-Moore algebras, which play a central role in the semantics of graded logics [37, 14].

▶ **Definition 4.3** (Graded Algebra). Let $\mathbb{M}$ be a graded monad in $\mathcal{C}$. A *graded M_n-algebra* $((A_k)_{k\leq n}, (a^{mk})_{m+k\leq n})$ consists of a family of $\mathcal{C}$-objects A_i and morphisms $a^{mk}\colon M_m A_k \to A_{m+k}$ satisfying essentially the same laws as a monad algebra, up to insertion of the grades. Specifically, we have $a^{0m}\cdot \eta_{A_m} = id_{A_m}$ for $m \leq n$, and whenever $m+r+k \leq n$, then $a^{m+r,k}\cdot \mu^{m,r}_{A_k} = a^{m,r+k}\cdot M_m a^{r,k}$. An *$M_n$-homomorphism* of M_n-algebras A and B is a family $(f_k\colon A_k \to B_k)_{k\leq n}$ of maps such that whenever $m+k\leq n$, then $f_{m+k}\cdot a^{m,k} = b^{m,k}\cdot M_m f_k$. Graded M_n-Algebras and their homomorphisms form a category $\mathrm{Alg}_n(\mathbb{M})$.

That is, elements of a graded algebra are stratified by depth, and applying an operation of depth m to elements of depth k yields elements of depth $m+k$, For $n=1$, this definition instantiates as follows: An M_1-algebra is a tuple $(A_0, A_1, a^{00}, a^{01}, a^{10})$, such that 1) (A_0, a^{00}) and (A_1, a^{01}) are M_0-algebras. 2) (Homomorphy) $a^{10}\colon M_1A_0 \to A_1$ is an M_0-homomorphism $(M_1A_0, \mu^{01}) \to (A_1, a^{01})$. 3) (Coequalization) $a^{10}\cdot M_1 a^{00} = a^{10}\cdot \mu^{10}$, i.e. the following diagram commutes (without necessarily being a coequalizer):

$$M_1M_0A_0 \underset{\mu^{10}}{\overset{M_1a^{00}}{\rightrightarrows}} M_1A_0 \xrightarrow{a^{10}} A_1 \tag{4.1}$$

It is easy to see that $((M_kX)_{k\leq n}, (\mu^{m,k})_{m+k\leq n})$ is an M_n-algebra for every $\mathcal{C}$-object X. Again, M_0-algebras are just (non-graded) algebras for the monad (M_0, η, μ^{00}).
The semantics of modalities will later need the following property:

▶ **Definition 4.4** (Canonical algebras). Let $(-)_0\colon \mathrm{Alg}_1(\mathbb{M}) \to \mathrm{Alg}_0(\mathbb{M})$ be the functor taking an M_1-algebra $A = ((A_k)_{k\leq 1}, (a^{mk})_{m+k\leq 1})$ to the M_0-algebra (A_0, a^{00}). An M_1-algebra A is *canonical* if it is free over $(A)_0$, i.e. if for every M_1-algebra B and M_0-homomorphism $f : (A)_0 \to (B)_0$, there is a unique M_1-homomorphism $g : A \to B$ such that $(g)_0 = f$.

▶ **Lemma 4.5** ([14, Lemma 5.3]). *An M_1-algebra A is canonical iff* (4.1) *is a coequalizer diagram in the category of M_0-algebras.*

5 Graded Quantitative Theories

Monads on **Set** are induced by equational theories [35]. By equipping each operation with an assigned depth and requiring each axiom to be of uniform depth, one obtains a notion of *graded equational theory* which, modulo size issues, can be brought into bijective correspondence with graded monads [37]. On the other hand, Mardare et al. [36] introduce a system of quantitative equational reasoning, with formulae of the form $s =_\epsilon t$ understood as "s differs from t by at most ϵ". These quantitative equational theories induce monads on the category of metric spaces. We introduce a graded version of this system to present graded monads in **Met**, keeping to finitary operations (and hence finite branching) for ease of presentation.

▶ **Definition 5.1** (Graded signatures, uniform terms). A *graded signature* consists of an algebraic signature Σ and a function $\delta\colon \Sigma \to \mathbb{N}$ assigning a *depth* to each algebraic operation. *Uniform depth* of terms is then defined inductively: Variables have uniform depth 0, and for m-ary $f \in \Sigma$, $f(t_1, \dots, t_m)$ has uniform depth $n+k$ if $\delta(f) = n$ and all t_i have uniform depth k. In particular, constants $c \in \Sigma$, as terms, have uniform depth n for all $n \geq \delta(c)$. We write $\mathbb{T}^\Sigma_n X$, or just $\mathbb{T}_n X$, for the set of terms of uniform depth n over X. A *substitution of uniform depth n* is a function $\sigma\colon X \to \mathbb{T}_n Y$. Such a substitution extends to a map $\sigma : \mathbb{T}_k X \to \mathbb{T}_{k+n} Y$ on terms for all $k \in \mathbb{N}$, where as usual one defines $\sigma(f(t_1, \dots, t_m)) = f(\sigma(t_1), \dots, \sigma(t_m))$. A substitution is *uniform-depth* if it is of uniform depth n for some n.

▶ **Definition 5.2** (Graded quantitative theory)**.** For a set Z, we let $\mathcal{E}(Z)$ denote the set of quantitative equalities $z_1 =_\epsilon z_2$ where $z_1, z_2 \in Z$ and $\epsilon \in [0,1]$. Given a set X of variables, we then write $\mathcal{E}(\mathbb{T}(X)) = \bigcup_{n\in\mathbb{N}} \mathcal{E}(\mathbb{T}_n(X))$; that is, $\mathcal{E}(\mathbb{T}(X))$ is the set of uniform-depth quantitative equalities among Σ-terms over X. A *quantitative theory* $\mathcal{T} = (\Sigma, \delta, E)$ consists of a graded signature (Σ, δ) and a set $E \subseteq \mathcal{P}(\mathcal{E}(X)) \times \mathcal{E}(\mathbb{T}X)$ of *axioms*. Axioms $(\Gamma, s =_\epsilon t)$ are written in the form $\Gamma \vdash s =_\epsilon t$; we refer to Γ as the *context* of the axiom. The *depth* of $\Gamma \vdash s =_\epsilon t$ is that of $s =_\epsilon t$. We say that $\mathcal{T}$ is *depth-1* if all its operations and axioms have depth at most 1.

The context Γ of an axiom $\Gamma \vdash s =_\epsilon t$ forms a constraint on the variables that is required in order for $s =_\epsilon t$ to hold. Correspondingly, *derivability* of quantitative equalities in $\mathcal{E}(\mathbb{T}(X))$ over a graded quantitative theory $\mathcal{T} = (\Sigma, \delta, E)$ in a *context* $\Gamma_0 \in \mathcal{P}(\mathcal{E}(X))$ is defined inductively by the following rules:

$$(\mathbf{triang})\ \frac{t =_\epsilon s \quad s =_{\epsilon'} u}{t =_{\epsilon+\epsilon'} u} \qquad (\mathbf{refl})\ \frac{}{s =_0 s} \qquad (\mathbf{sym})\ \frac{t =_\epsilon s}{s =_\epsilon t}$$

$$(\mathbf{wk})\ \frac{t =_\epsilon s}{t =_{\epsilon'} s}\ (\epsilon' \geq \epsilon) \qquad (\mathbf{arch})\ \frac{\{t =_{\epsilon'} s \mid \epsilon' > \epsilon\}}{t =_\epsilon s} \qquad (\mathbf{assn})\ \frac{}{\phi}\ (\phi \in \Gamma_0)$$

$$(\mathbf{ax})\ \frac{\{\sigma(u) \mid u \in \Gamma\}}{\sigma(t) =_\epsilon \sigma(s)}\ ((\Gamma, t =_\epsilon s) \in E) \qquad (\mathbf{nexp})\ \frac{t_1 =_\epsilon s_1 \quad \dots \quad t_n =_\epsilon s_n}{f(t_1, \dots, t_n) =_\epsilon f(s_1, \dots, s_n)}$$

where σ is a uniform-depth substitution. Note the difference between rules (**ax**) and (**assn**): Quantitative equalities from the theory can be substituted into, while this is not sound for quantitative equalities from the context. A graded quantitative equational theory *presents* a graded monad $\mathbb{M}$ on **Met** where M_nX is the set of terms of uniform depth n over variables in X, quotiented by the equivalence relation that identifies terms s, t if $s =_0 t$ is derivable in context X, with the distance $d_{M_n}([s],[t]) = \epsilon$ of equivalence classes $[s], [t] \in M_nX$ being the least ϵ such that $s =_\epsilon t$ is derivable (which exists by (**arch**)). Multiplication collapses terms-over-terms, and the unit maps an element of $x \in X$ to $[x] \in M_0X$.

▶ Remark 5.3. The above system for quantitative reasoning follows Ford et al. [20] in slight modifications to the original (ungraded) system [36]. In particular, we make do without a cut rule, and allow substitution only into axioms (substitution into derived equalities is then admissible [20]). We include the rule **(nexp)** ensuring that all operations are nonexpansive, i.e. the induced graded monad is *enriched* (acts nonexpansively on functions).

We recall that a graded monad is *depth*-1 [37, 14] if μ^{nk} and $M_0\mu^{1k}$ are epi-transformations and the diagram below is a coequalizer of M_0-algebras for all X and $n < \omega$:

$$M_1M_0M_nX \underset{\mu^{10}M_n}{\overset{M_1\mu^{0n}}{\rightrightarrows}} M_1M_nX \xrightarrow{\mu^{1n}} M_{1+n}X. \tag{5.1}$$

By Lemma 4.5 the following is then immediate:

▶ **Proposition 5.4** ([14, Corollary 5.4])**.** *If $\mathbb{M}$ is a depth-1 graded monad, then for every $n \in \mathbb{N}$ and every object X, the M_1-algebra with carriers M_nX, $M_{n+1}X$ and multiplications as algebra structure is canonical.*

We briefly refer to canonical algebras as per the above proposition as being *of the form* M_nX.

Crucially, we establish a metric variant of a result on depth-1 graded monads on **Set** [37]:

▶ **Theorem 5.5.** *Graded monads on* **Met** *presented by depth-*1 *graded quantitative theories are depth-*1*.*

▶ Remark 5.6. A depth-1 graded monad $\mathbb{M}$ can be reconstructed from its constituents of depth at most one, i.e. from M_0, M_1, η, and the μ^{nk} for $n + k \leq 1$ [14]. Graded semantics (Section 6) does however make use of the full structure of $\mathbb{M}$ also at higher depths.

Presentations of graded trace monads. We proceed to investigate the quantitative-algebraic presentation of graded trace monads that are given by a Kleisli distributive law of the functor $\mathcal{A} \times (-)$ (with $\mathcal{A}$ being the space of action labels) over a monad (Example 4.2). Given a function $k\colon [0,1]^2 \to [0,1]$ with suitable properties, we write $\otimes$ for the tensor that equips the Cartesian product of two sets with the metric $d_{A\otimes B}((a,b),(a',b')) = k(d(a,a'),d(b,b'))$ generated by k. This induces trace distances on $\mathcal{A}^n$, $n \geq 0$, by viewing $\mathcal{A}^n$ as the n-fold tensor of $\mathcal{A}$. Examples include the Euclidean ($k(x,y) = \sqrt{x^2+y^2}$), supremum ($k(x,y) = \max(x,y)$), and Manhattan ($k(x,y) = x \oplus y$) distances. The fact that k computes distances of traces recursively "one symbol at a time" translates into uniform depth-1 equations:

▶ **Definition 5.7.** Let $\mathcal{T} = (\Sigma, \mathcal{E})$ be a quantitative algebraic presentation of a (plain) monad T on **Met**. We define a graded quantitative theory $\mathcal{T}[\mathcal{A}]$ by including the operations and equations of $\mathcal{T}$ at depth 0, along with unary depth-1 operations a for all labels $a \in \mathcal{A}$, and as depth-1 axioms the distributive laws $\vdash a(f(x_1,\dots,x_n)) =_0 f(a(x_1),\dots,a(x_n))$ for all $a \in \mathcal{A}$ and $f \in \Sigma$, as well as the distance axioms $x =_\epsilon y \vdash a(x) =_{k(d(a,b),\epsilon)} b(y)$.

The obvious candidate for a Kleisli distributive law inducing the graded monad presented by the theory $\mathcal{T}[\mathcal{A}]$ is the family of maps $\lambda_X\colon \mathcal{A} \otimes TX \to T(\mathcal{A} \otimes X)$ given by

$$\lambda_X(a,t) = T\langle a, id_X\rangle_\otimes(t) \tag{5.2}$$

where $\langle a, id_X\rangle_\otimes$ takes $x \in X$ to $(a,x) \in \mathcal{A} \otimes X$. However, these maps λ_X may fail to be nonexpansive, depending on T and $\otimes$; for instance, this happens for $T = \overline{\mathcal{D}}_\omega$ and $\otimes$ being Cartesian product $\times$ (which carries the supremum distance):

▶ **Example 5.8.** Put $X = \{x,y\}$ where $d(x,y) = 1$, and $s = 0.5 \cdot x + 0.5 \cdot y$, $t = 1 \cdot x \in \overline{\mathcal{D}}_\omega X$. Clearly $d(s,t) = 0.5$. Given $a, b \in \mathcal{A}$ with $d(a,b) = 0.5$, we have $d((a,s),(b,t)) = 0.5$ in $\mathcal{A} \times \overline{\mathcal{D}}_\omega X$ while $d(\lambda_X(a,s),\lambda_X(b,t)) = d(0.5\cdot(a,x)+0.5\cdot(a,y), 1\cdot(b,x)) = 0.75$ in $\overline{\mathcal{D}}_\omega(\mathcal{A}\times X)$.

Nonexpansiveness is, of course, needed to obtain a graded monad on **Met**, and as we show later (Remark 6.3), its failure may cause undesirable effects. In the case of Manhattan distance, nonexpansiveness always holds:

▶ **Lemma 5.9.** *The maps λ_X as per* (5.2) *are nonexpansive as maps $\mathcal{A} \boxplus TX \to T(\mathcal{A} \boxplus X)$.*

In case the λ_X as per (5.2) *are* nonexpansive, we do in fact have that the distributive law λ and the algebraic theory $\mathcal{T}[\mathcal{A}]$ induce the same graded monad:

▶ **Lemma 5.10.** *Let λ_X be defined by* (5.2)*. If $\lambda_X\colon \mathcal{A}\otimes T \to T(\mathcal{A}\otimes(-))$ is nonexpansive for all X, then the λ_X form a Kleisli distributive law $\lambda\colon \mathcal{A}\otimes T \to T(\mathcal{A}\otimes(-))$, and the graded monad induced by λ according to Example 4.2 is presented by the quantitative equational theory $\mathcal{T}[\mathcal{A}]$ as per Definition 5.7.*

▶ **Example 5.11.** In our running example of finitely branching metric trace semantics, it is easy to check that the distributive law claimed in Example 4.2 is indeed nonexpansive, so the induced graded monad is, by Lemma 5.10, presented by the corresponding theory as per

Definition 5.7, and in particular is depth-1. Explicitly, recall [36, Corollary 9.4] that $\overline{\mathcal{P}}_\omega$ is a monad, presented in quantitative algebra by the usual axioms of join semilattices for a binary join operation $+$ and a constant 0 (nonexpansiveness of $+$ is enforced by the deduction rules). The quantitative graded theory presenting the graded metric trace monad $\overline{\mathcal{P}}_\omega(\mathcal{A}^n \times -)$ according to Lemma 5.10 has depth-0 operators $+$ and 0 as above and adds unary depth-1 operations a for all $a \in \mathcal{A}$, subject to axioms (for $a, b \in \mathcal{A}$, $\epsilon \in [0,1]$)

$$\vdash a(0) =_0 0 \qquad \vdash a(x+y) =_0 a(x) + a(y) \qquad x =_\epsilon y \vdash a(x) =_{\epsilon \vee d_{\mathcal{A}}(a,b)} b(y).$$

The distribution of the operations a over the join semilattice structure effectively implements trace equivalence, and the last axiom determines the metric on traces, which in this case is taken to be the supremum metric.

6 Graded Quantitative Semantics and Graded Logics

We proceed to introduce the framework of *graded quantitative semantics*, to study spectra of behavioural metrics for various system types. By "spectra" we informally refer to collections of process comparisons of varying granularity that arise by observing a specific system type in different ways, as exemplified by the classical linear-time/branching-time spectrum on labelled transition systems [47]. Generally, a *graded semantics* [37] $(\mathbb{M}, \alpha)$ of a functor $G\colon \mathcal{C} \to \mathcal{C}$ consists of a graded monad $\mathbb{M}$ and a natural transformation $\alpha\colon G \to M_1$. Intuitively, $M_n 1$ (where 1 is a terminal object of $\mathcal{C}$) is a domain of behaviours observable after n transition steps, with α determining behaviours after one step. For a G-coalgebra (X, γ), we inductively define *behaviour maps* $\gamma^{(n)}\colon X \to M_n 1$ assigning to a state in X its behaviour after n steps:

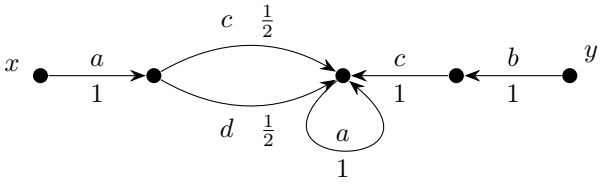
$$\gamma^{(0)}\colon X \xrightarrow{M_0 !\cdot \eta} M_0 1 \qquad \gamma^{(n+1)}\colon X \xrightarrow{\alpha\cdot\gamma} M_1 X \xrightarrow{M_1\gamma^{(n)}} M_1 M_n 1 \xrightarrow{\mu^{1n}} M_{n+1} 1$$

For $\mathcal{C} = \mathbf{Met}$, these maps induce a notion of *graded behavioural distance* (for readability, we refrain from working with more general $\mathcal{C}$, such as categories of relational structures [20]):

▶ **Definition 6.1** (Graded behavioural distance)**.** Given a graded semantics $\alpha\colon G \to M_1$ of a functor G on $\mathbf{Met}$, *(graded) behavioural distance* is the pseudometric on states in G-coalgebras (X, γ) given by $d^\alpha(x, y) = \bigvee_{n\in\mathbb{N}} d_{M_n 1}(\gamma^{(n)}(x), \gamma^{(n)}(y))$ for $x, y \in X$.

▶ **Example 6.2.** The metric trace semantics of finitely branching metric transition systems [11, 15] and closed-branching metric transition systems is captured by the graded metric trace monads $M_n = \overline{\mathcal{P}}_\omega(\mathcal{A}^n \times -)$ and $M_n = \mathcal{P}_c(\mathcal{A}^n \times -)$ (Example 4.2), respectively (with α being identity). The behaviour maps calculate, at each depth n, sets of length-n traces, whose distance is given by the Hausdorff distance induced by the supremum metric on traces.

▶ Remark 6.3. In cases where nonexpansiveness of α or the natural transformations of $\mathbb{M}$ does not hold (e.g. if one attempts to construct $\mathbb{M}$ using a family of maps (5.2) that fails to be nonexpansive, cf. Example 5.8), other expected properties can fail. For instance, it can happen that trace distance exceeds branching time distance (while for trace semantics induced by nonexpansive graded semantics, general properties of graded semantics imply that trace distance is below branching-time distance, in tune with the two-valued setting where trace equivalence is coarser than bisimilarity). Example 5.8 manifests in the $\overline{\mathcal{D}}_\omega(\mathcal{A} \times -)$-coalgebra (i.e. generative probabilistic metric transition system) shown below, where $\mathcal{A} = \{a, b, c, d\}$ with relevant distances $d(a, b) = 0.5$ and $d(c, d) = 1$:



Here, we have length-n trace distributions $\mu_x^n = \frac{1}{2}\cdot(aca^{n-2}) + \frac{1}{2}\cdot(ada^{n-2})$ and $\mu_y^n = 1\cdot(bca^{n-2})$ for $n \geq 2$. When the metric on traces is defined via supremum distance, instead of Manhattan distance as in Section 3, the trace distance of the states x and y is $\bigvee_{n\in\mathbb{N}} d(\mu_x^n, \mu_y^n) = 0.75$, while their branching-time distance (cf. Section 2) is 0.5.

We have the following criterion for invariance of a logic under a graded semantics $(\alpha, \mathbb{M})$, with $\mathbb{M}$ depth-1, for a functor $G\colon \mathbf{Met} \to \mathbf{Met}$ *that we fix from now on*; recall from Section 2 that we use Ω to denote the unit interval $[0,1]$ equipped with Euclidean distance.

▶ **Definition 6.4** (Graded logic)**.** Let $o\colon M_0\Omega \to \Omega$ be an M_0-algebra structure on Ω. A logic $\mathcal{L}$ is a *graded logic* (for $(\alpha, \mathbb{M})$) if the following hold:

1. For n-ary $p \in \mathcal{O}$, the semantics $[\![p]\!]$ is an M_0-algebra homomorphism $(\Omega, o)^n \to (\Omega, o)$.
2. For each $L \in \Lambda$, there is an associated nonexpansive map $(\!|L|\!)\colon M_1\Omega \to \Omega$ such that the semantics $[\![L]\!]\colon G\Omega \to \Omega$ factors as $[\![L]\!] = (G\Omega \xrightarrow{\alpha_\Omega} M_1\Omega \xrightarrow{(\!|L|\!)} \Omega)$, and such that the tuple $(\Omega, \Omega, o, o, (\!|L|\!))$ constitutes an M_1-algebra (that is, $(\!|L|\!)$ satisfies homomorphy and coequalization, cf. Section 2). We abuse notation and write $(\!|L|\!)$ to denote the M_1-algebra $(\Omega, \Omega, o, o, (\!|L|\!))$.

Notice the different treatment of nullary propositional operators and truth constants: The former are required to be interpreted as homomorphisms $1 \to (\Omega, o)$ in a graded logic, while no such condition is imposed on truth constants. In many examples, $\alpha = id$, in which case condition 2 just states that $(\Omega, \Omega, o, o, [\![L]\!])$ is an M_1-algebra (non-identity α are associated, for instance, with readiness and failure semantics [14]).

▶ **Definition 6.5.** We say that $\mathcal{L}$ is *invariant* with respect to a graded semantics $(\alpha, \mathbb{M})$ if $d^{\mathcal{L}} \leq d^\alpha$ holds in all G-coalgebras; *expressive* if $d^{\mathcal{L}} \geq d^\alpha$; and *characteristic* if $d^{\mathcal{L}} = d^\alpha$.

▶ **Theorem 6.6** ([24, Proposition 21])**.** *Let $\mathcal{L}$ be a graded logic for $(\alpha, \mathbb{M})$. Then the evaluation maps $[\![\phi]\!]_\gamma$ of uniform-depth $\mathcal{L}$-formulae ϕ on G-coalgebras (X, γ) are nonexpansive w.r.t. behavioural distance d^α, and hence $\mathcal{L}$ is invariant.*

The assumption of uniform depth cannot be removed in general [24].

▶ **Example 6.7.** We have a graded logic $\mathcal{L}^{\mathsf{mtrace}}$ for metric trace semantics (Example 6.2) featuring modalities $\Diamond_a$ for all $a \in \mathcal{A}$ as in Example 2.4, a single truth constant 1, and no propositional operators. We equip the set $\Omega = [0,1]$ of truth values with the usual $\overline{\mathcal{P}}_\omega$-algebra (i.e. join semilattice) structure $([0,1], \vee, 0)$, and let $\hat{1}\colon 1 \to [0,1]$ take the value 1. The logic $\mathcal{L}^{\mathsf{mtrace}}$ remains invariant under metric trace semantics when extended with propositional operators that are nonexpansive join-semilattice morphisms, such as $\vee$. Analogously we define the logic $\mathcal{L}^{\mathsf{cmtrace}}$ for trace semantics of closed-branching metric transition systems. Notice that the interpretation of 1 fails to be homomorphic, so 1 needs to be a truth constant.

7 Expressivity Criteria

We proceed to adapt expressivity criteria appearing in previous work on two-valued behavioural equivalences [14, 19] to the quantitative setting, which poses quite specific challenges. A key role in the treatment of expressivity of logics will be played by the notion of initiality [1].

▶ **Definition 7.1.** A family of maps $(f_i\colon A \to B)_{i\in I}$ between metric spaces A and B is *initial* if A carries the smallest (pseudo-)metric making all maps f_i nonexpansive, explicitly: $d(x,y) = \bigvee_i d(f_i(x), f_i(y))$.

Using this notion, the definition of expressivity can be rephrased as follows: An invariant logic $\mathcal{L}$ is expressive if for every G-coalgebra (X,γ), the family of all evaluation maps $[\![\phi]\!]_\gamma$ of uniform-depth formulae ϕ is initial on (X, d^α).

▶ Remark 7.2. In the branching-time case, a stronger notion of expressivity, roughly phrased as *density* of the set of depth-n formulae in the set of nonexpansive properties at depth n, follows from expressivity under certain additional conditions [22, 48, 49, 51, 33], using lattice-theoretic variants of the Stone-Weierstraß theorem. The analogue of the Stone-Weierstraß theorem in general fails for coarser semantics. Also, for semantics coarser than branching time, expressivity in the sense of Definition 6.5 can often be established using more economic sets of propositional operators (e.g. no propositional operators at all), for which density will clearly fail.

Our expressivity result is based on propagating initiality through an induction on depth. Unlike in the Eilenberg-Moore case [24], this requires, in many examples, to strengthen the inductive invariant; we treat this systematically as follows:

▶ **Definition 7.3.** An *initiality invariant* is a property Φ of sets $\mathfrak{A} \subseteq \mathbf{Met}(X,\Omega)$ of nonexpansive functions such that (i) every family of maps satisfying Φ is initial, and (ii) Φ is upwards closed w.r.t. subset inclusion.

▶ **Example 7.4.**

1. Initiality itself is an initiality invariant. If Φ is initiality, then we say "initial-type" for "Φ-type".
2. We say that $\mathfrak{A} \subseteq \mathbf{Met}(X,\Omega)$ is *normed isometric* if whenever $d(x,y) > \epsilon$ for $x, y \in X$ and $\epsilon > 0$, then there is some $f \in \mathfrak{A}$ such that $|f(x) - f(y)| > \epsilon$ and $f(x) \vee f(y) = 1$. Normed isometry is an initiality invariant.

Our expressivity criterion then takes the following shape:

▶ **Definition 7.5.** Let Φ be an initiality invariant. A graded logic $\mathcal{L} = (\Theta, \mathcal{O}, \Lambda)$ with truth value object (Ω, o) is Φ-*type depth-0 separating* if the family of maps $\{o \cdot M_0\hat{c}\colon M_0 1 \to \Omega \mid c \in \Theta\}$ has property Φ. Moreover, $\mathcal{L}$ is Φ-*type depth-1 separating* if whenever A is a canonical M_1-algebra of the form $M_n 1$ (Proposition 5.4) and $\mathfrak{A}$ is a set of M_0-homomorphisms $A_0 \to \Omega$ that has property Φ and is closed under the propositional operators in $\mathcal{O}$, then the set

$$\Lambda(\mathfrak{A}) := \{(\!|L|\!)^\bullet(g) : A_1 \to \Omega \mid L \in \Lambda, g \in \mathfrak{A}\}$$

has property Φ, where $(\!|L|\!)^\bullet(g)\colon A_1 \to \Omega$ is the (by canonicity, unique) morphism extending the M_0-algebra morphism g to an M_1-algebra morphism $A \to (\!|L|\!)$ (Definition 6.4).

▶ **Theorem 7.6** (Expressivity)**.** *Let Φ be an initiality invariant, and suppose that a graded logic $\mathcal{L}$ is both Φ-type depth-0 separating and Φ-type depth-1 separating. Then $\mathcal{L}$ is expressive.*

▶ Remark 7.7. Our definition of separation differs from notions used for two-valued logics [14, 19] and for quantitative graded semantics induced by Eilenberg-Moore distributive laws [24], which overall have turned out to be much more well-behaved than the more general setting of the present work. The most obvious novelty is the use of an initiality invariant Φ strengthening the induction hypothesis in the inductive proof of Theorem 7.6. We will see that this is needed even in very simple examples in our more general setting. Moreover, we have phrased

separation in terms of the specific canonical algebras M_n1 on which it is needed, rather than on unrestricted canonical algebras. This allows exploiting additional properties of M_n1, e.g. that for graded monads $M_n = TF^n$ arising from Kleisli distributive laws (Example 4.2), M_n1 is free as an M_0-algebra.

▶ **Example 7.8.**

1. *Metric Streams:* A simple example for failure of initial-type separation (Example 7.4) are metric streams, i.e. streams over a metric space of labels $(\mathcal{A}, d_\mathcal{A})$; these are coalgebras for the functor $G = \mathcal{A} \times -$. Behavioural distance on streams is captured by the graded monad $G^n = \mathcal{A}^n \times \{-\}$. The logic $\mathcal{L}$ consisting of the truth constant 1 and modalities $\Diamond_a$ for all $a \in \mathcal{A}$, with interpretation $[\![\Diamond_a]\!] \colon \mathcal{A} \times [0,1] \to [0,1]$ given by $(b, v) \mapsto (1 - d_\mathcal{A}(a, b)) \wedge v$, is Φ-type depth-0 separating and Φ-type depth-1 separating for Φ being normed isometry, and hence expressive by Theorem 7.6. (The modality $\Diamond_a$ restricts the corresponding modality for metric transition systems as per Examples 2.4 and 6.7 to metric streams: a state satisfies $\Diamond_a\phi$ to the degree that its output is close to a and its successor satisfies ϕ). On the other hand, $\mathcal{L}$ fails to be initial-type depth-1 separating, illustrating the necessity of the general form of Theorem 7.6.
2. *Metric transition systems:* The graded logics $\mathcal{L}^{\mathsf{mtrace}}$ and $\mathcal{L}^{\mathsf{cmtrace}}$ for metric trace semantics (Example 6.7), in the version with no propositional operators, are Φ-type depth-0 separating and Φ-type depth-1 separating for Φ being normed isometry, and hence are expressive by Theorem 7.6. We thus improve on an example from recent work based on Galois connections [4], where application of the general framework required the inclusion of propositional shift operators (which were subsequently eliminated in an ad-hoc manner), and we generalize to systems with closed branching on a metric state space.
3. Probabilistic metric trace semantics is modelled straightforwardly as a graded semantics using a graded trace monad (Example 4.2). By Theorem 3.2, however, there is no graded logic for probabilistic metric trace semantics that satisfies the conditions of Theorem 7.6.

▶ Remark 7.9. In a recent approach based on Galois connections [4, 5], logics are related to fixpoints of behaviour functions induced by the logic itself (similar to approaches that define trace semantics via intended characteristic logics [31]), while our present interest is in providing logical characterizations of *given* behavioural distances. The Galois framework is highly general, and in fact not even tied to coalgebraic modelling, or in fact to state-based systems of any kind [4], but correspondingly offers less concrete recipes. Instantiated to our current setup, the key condition of *compatibility* appearing in *op. cit.* roughly speaking amounts to initial-type depth-1 separation of the logic w.r.t. its own Kantorovich lifting [2, 5].

▶ Remark 7.10 (Branching-time semantics). Any functor G yields a graded monad given by iterated application of G, that is $M_n = G^n$, and by unit and multiplication being identity [37]. In general, the finite-depth branching-time semantics of a G-coalgebra (X, γ) is defined via its *canonical cone* $(p_i \colon X \to G^i1)_{i<\omega}$ into the *final sequence* $1 \xleftarrow{!} G1 \xleftarrow{G!} G^21 \leftarrow \dots$ of G. The p_i are defined inductively by $p_0 = ! : X \to 1$ and $p_{i+1} = Gp_i \cdot \gamma$. This semantics is captured by the graded monad $M_n = G^n$ and $\alpha = id$ [37]. More specifically, the *finite-depth branching-time behavioural distance* of states $x, y \in X$ is $\bigvee_{i<\omega} d(p_i(x), p_i(y))$, and thus agrees with the graded behavioural distance obtained via the graded semantics in the graded monad $M_n = G^n$. This monad has $M_0 = id$, so that the corresponding graded logics are just branching-time logics without further restriction [37, 14]. Coalgebraic quantitative logics of this kind have received some recent attention [22, 48, 49, 51, 11, 30, 33]. Suppose Λ is a finite *separating* set of modalities, i.e. the maps $[\![L]\!] \cdot Gf \colon GX \to \Omega$, with L ranging over modalities and f over nonexpansive maps $X \to \Omega$, form an initial family. Moreover, let $\mathcal{O}$ contain

truth 1, meet $\wedge$, fuzzy negation $\neg$ (i.e. $\neg x = 1 - x$), and truncated addition of constants $(-) \oplus c$. Then one shows using a variant of the Stone-Weierstraß theorem [51] that the graded logic $\mathcal{L}$ given by Λ, $\mathcal{O}$, and $\Theta = \emptyset$ is initial-type depth-0 separating and initial-type depth-1 separating. By Theorem 7.6, we obtain that $\mathcal{L}$ is expressive. Previous work on quantitative branching-time logics [51, 33, 48, 49, 22] discusses, amongst other things, conditions on G that allow concluding expressivity even for infinite-depth behavioural distance.

8 Case Study: Fuzzy Metric Trace Semantics

We apply the recipe outlined above to obtain a characteristic logic for trace distance on *fuzzy metric transition systems*. That is, we proceed as follows: We cast fuzzy metric trace distance as a graded semantics using a suitable depth-1 graded monad $\mathbb{M}$, and check that $\mathbb{M}$ is depth-1 using the techniques outlined in Section 5. We then identify a corresponding graded logic $\mathcal{L}$, verifying the requirements of Definition 6.4. Invariance of $\mathcal{L}$ then follows automatically (Theorem 6.6). Finally, we show expressivity using Theorem 7.6.

A *fuzzy $\mathcal{A}$-labelled metric transition system (fuzzy metric LTS)* [12, 54, 55, 30]) consists of a set (or metric space) X of states and a fuzzy transition relation $R\colon X \times \mathcal{A} \times X \to [0,1]$, with $\mathcal{A}$ a metric space. A fuzzy LTS (X, R) is *finitely branching* if $\{(a, y) \mid R(x, a, y) > 0\}$ is finite for every $x \in X$. Equivalently, a finitely branching fuzzy LTS is a coalgebra for the functor $\overline{\mathcal{F}}_\omega(\mathcal{A} \times (-))$ (cf. Example 2.2.4).

A natural fuzzy trace semantics of fuzzy transition systems assigns to each state x of a fuzzy LTS (X, R) a fuzzy trace set $Tr(x) \in \mathcal{F}_\omega(\mathcal{A}^*)$ where

$$Tr(x)(a_1 \dots a_n) = \bigvee\{\textstyle\bigwedge_{i=1}^n R(x_{i-1}, a_i, x_i) \mid x = x_0, x_1, \dots, x_n \in X\}.$$

This notion of trace relates, for instance, to a notion of fuzzy path that is implicit in the semantics of fuzzy computation tree logic [38] and to notions of fuzzy language accepted by fuzzy automata (e.g. [3]). We obtain a notion of *fuzzy trace distance* d^T of states x, y, given by the distance of $Tr(x)$, $Tr(y)$ in $\overline{\mathcal{F}}_\omega(\mathcal{A}^*)$, i.e. under fuzzy Hausdorff distance (Example 2.2.4) w.r.t. the metric on $\mathcal{A}^*$ that is the supremum metric on each $\mathcal{A}^n$, and assigns distance 1 to traces of different lengths. To capture this distance in a graded semantics, consider the distributive law $\lambda\colon \mathcal{A} \times \overline{\mathcal{F}}_\omega(-) \to \overline{\mathcal{F}}_\omega(\mathcal{A} \times -)$ given by $\lambda(a, U)(a, x) = U(x)$ and $\lambda(a, U)(b, x) = 0$ for $b \neq a$. By Example 4.2 we thus obtain the *graded fuzzy metric trace monad* $M_n = \overline{\mathcal{F}}_\omega(\mathcal{A}^n \times (-))$. The monad $\overline{\mathcal{F}}_\omega$ can be presented by the following quantitative equational theory: Take a binary operation $+$, a constant 0, and unary operations r for every $r \in [0,1]$. Impose strict equations $(=_0)$ saying that $+$, 0 form a join semilattice structure and that the operations r define an action of the monoid $([0,1], \wedge)$ (i.e. $1(x) = x$, $r(s(x)) =_0 (r \wedge s)(x)$). Finally, impose axioms $x =_\epsilon y \vdash r(x) =_\epsilon s(y)$ for $r, s \in [0,1]$ such that $|r - s| \leq \epsilon$. By Lemma 5.10, the graded fuzzy trace monad $M_n = \overline{\mathcal{F}}_\omega(\mathcal{A}^n \times X)$ is presented by the above algebraic description of $\overline{\mathcal{F}}_\omega$ at depth 0, with additional depth-1 unary operations a for $a \in \mathcal{A}$ and depth-1 equations $a(x + y) =_0 a(x) + a(y)$, $a(0) =_0 0$, $a(r(x)) =_0 r(a(x))$, and $x =_\epsilon y \vdash a(x) =_{\epsilon \vee d(a,b)} b(y)$.

Fuzzy metric trace logic interprets the additional operations $r \in [0,1]$ on the truth value object $[0,1]$ by $r(x) = r \wedge x$, and otherwise uses the same quantitative join semilattice structure as for metric trace semantics (Example 6.7). We include the truth constant 1 and modal operators $\Diamond_a^c$ for $a \in \mathcal{A}$ and $c \in [0,1] \cap \mathbb{Q}$, with interpretation $[\![\Diamond_a^c]\!]\colon M_1[0,1] \to [0,1]$ given by $[\![\Diamond_a^c]\!](A) = \bigvee_{b \in \mathcal{A}, v \in [0,1]} A(b, v) \wedge v \wedge (c \ominus d(a, b))$. (When $\mathcal{A}$ is discrete, then $\Diamond_a^1$ is the usual fuzzy diamond modality, e.g. [18]). Thus, a state x in a fuzzy metric transition system satisfies $\Diamond_a^c \phi$ to the degree that x has a b-successor y with b close to a and y satisfying ϕ;

crucially, "closeness" of b to a needs to be shifted down as governed by the parameter c. This logic is initial-type depth-0 separating and initial-type depth-1 separating, and hence expressive for fuzzy trace distance by Theorem 7.6; both this result and the logic itself appear to be new (the case with $\mathcal{A}$ discrete is partially covered in work on Galois connections [5]). Indeed for non-discrete $\mathcal{A}$, the logic with only $\Diamond_a^1$ instead of all $\Diamond_a^c$ fails to be expressive. The logic remains invariant when extended with additional nonexpansive propositional operators that are $\overline{\mathcal{F}}_\omega$-homomorphic, such as $\vee$.

9 Conclusions

We have shown that there is no unary quantitative coalgebraic modal logic characterizing a natural notion of quantitative trace distance on probabilistic metric transition systems. Moving onwards from this observation, we have developed a generic framework for linear-time/branching-time spectra of behavioural distances on state-based systems in coalgebraic generality, covering, for instance, metric, probabilistic, and fuzzy transition systems. Unlike previous work on Eilenberg-Moore-style coalgebraic trace distances [5, 24], the framework covers also systems with labels from a metric space. The key abstractions in the framework are based on the notion of a graded monad on the category of metric spaces and an arising notion of quantitative graded semantics. We have provided a graded quantitative algebraic system for the description of such graded monads (extending and modifying the existing non-graded system [36]). Moreover, we have established sufficient conditions for canonical invariant *quantitative graded logics* [24] to be *expressive* for given quantitative graded semantics, and we have exploited this result to obtain expressive logics for some instances of Kleisli-type trace semantics [27], notably including a new result for fuzzy metric trace semantics.

One important next step in the development will be to identify a generic game-based characterization of behavioural distances in the framework of graded semantics, generalizing work specific to metric transition systems [15] and building on game-based concepts for two-valued graded semantics [21]. Also, there is interest in computing distinguishing quantitative formulae (cf. [10, 52] for the two-valued branching-time setting), generalizing recent results for the branching-time case [40] to spectra of coarser semantics.

References

1 Jiří Adámek, Horst Herrlich, and George E. Strecker. *Abstract and Concrete Categories*. John Wiley and Sons, 1990. Reprint: `http://www.tac.mta.ca/tac/reprints/articles/17/tr17abs.html`.

2 Paolo Baldan, Filippo Bonchi, Henning Kerstan, and Barbara König. Coalgebraic behavioral metrics. *Log. Methods Comput. Sci.*, 14(3), 2018. `doi:10.23638/LMCS-14(3:20)2018`.

3 Radim Belohlávek. Determinism and fuzzy automata. *Inf. Sci.*, 143(1-4):205–209, 2002. `doi:10.1016/S0020-0255(02)00192-5`.

4 Harsh Beohar, Sebastian Gurke, Barbara König, and Karla Messing. Hennessy-Milner theorems via Galois connections. In Bartek Klin and Elaine Pimentel, editors, *Computer Science Logic, CSL 2023*, volume 252 of *LIPIcs*, pages 12:1–12:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. `doi:10.4230/LIPIcs.CSL.2023.12`.

5 Harsh Beohar, Sebastian Gurke, Barbara König, Karla Messing, Jonas Forster, Lutz Schröder, and Paul Wild. Expressive quantale-valued logics for coalgebras: An adjunction-based approach. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *Theoretical Aspects of Computer Science, STACS 2024*, volume 289 of *LIPIcs*, pages 10:1–10:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. `doi:10.4230/LIPICS.STACS.2024.10`.

6 Marco Bernardo and Stefania Botta. A survey of modal logics characterising behavioural equivalences for non-deterministic and stochastic systems. *Math. Struct. Comput. Sci.*, 18(1):29–55, 2008. doi:10.1017/S0960129507006408.

7 Valentina Castiglioni, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. A logical characterization of differential privacy via behavioral metrics. In Kyungmin Bae and Peter Ölveczky, editors, *Formal Aspects of Component Software, FACS 2018*, volume 11222 of *LNCS*, pages 75–96. Springer, 2018. doi:10.1007/978-3-030-02146-7.

8 Valentina Castiglioni, Daniel Gebler, and Simone Tini. Logical characterization of bisimulation metrics. In Mirco Tribastone and Herbert Wiklicky, editors, *Quantitative Aspects of Programming Languages and Systems, QAPL 2016*, volume 227 of *EPTCS*, pages 44–62, 2016. doi:10.4204/EPTCS.227.

9 Ivan Christoff. Testing equivalences and fully abstract models for probabilistic processes. In Jos C. M. Baeten and Jan Willem Klop, editors, *Theories of Concurrency: Unification and Extension, CONCUR 1990*, volume 458 of *LNCS*, pages 126–140. Springer, 1990. doi:10.1007/BFb0039056.

10 Rance Cleaveland. On automatically explaining bisimulation inequivalence. In Edmund M. Clarke and Robert P. Kurshan, editors, *Computer Aided Verification, CAV 1990*, volume 531 of *LNCS*, pages 364–372. Springer, 1990. doi:10.1007/BFB0023750.

11 Luca de Alfaro, Marco Faella, and Mariëlle Stoelinga. Linear and branching system metrics. *IEEE Trans. Software Eng.*, 35(2):258–273, 2009. doi:10.1109/TSE.2008.106.

12 Liliana D'Errico and Michele Loreti. Modeling fuzzy behaviours in concurrent systems. In Giuseppe F. Italiano, Eugenio Moggi, and Luigi Laura, editors, *Theoretical Computer Science, 10th Italian Conference, ICTCS 2007*, pages 94–105. World Scientific, 2007. doi:10.1142/9789812770998_0012.

13 Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled Markov processes. *Theor. Comput. Sci.*, 318(3):323–354, 2004. doi:10.1016/j.tcs.2003.09.013.

14 Ulrich Dorsch, Stefan Milius, and Lutz Schröder. Graded monads and graded logics for the linear time - branching time spectrum. In Wan J. Fokkink and Rob van Glabbeek, editors, *Concurrency Theory, CONCUR 2019*, volume 140 of *LIPIcs*, pages 36:1–36:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CONCUR.2019.36.

15 Uli Fahrenberg and Axel Legay. The quantitative linear-time-branching-time spectrum. *Theor. Comput. Sci.*, 538:54–69, 2014. doi:10.1016/j.tcs.2013.07.030.

16 Uli Fahrenberg, Axel Legay, and Claus Thrane. The quantitative linear-time–branching-time spectrum. In Supratik Chakraborty and Amit Kumar, editors, *Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011*, volume 13 of *LIPIcs*, pages 103–114. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2011. doi:10.4230/LIPIcs.FSTTCS.2011.103.

17 Ihor Filimonov, Ross Horne, Sjouke Mauw, and Zach Smith. Breaking unlinkability of the ICAO 9303 standard for e-passports using bisimilarity. In Kazue Sako, Steve A. Schneider, and Peter Y. A. Ryan, editors, *Computer Security, ESORICS 2019*, volume 11735 of *LNCS*, pages 577–594. Springer, 2019. doi:10.1007/978-3-030-29959-0_28.

18 Melvin Fitting. Many-valued modal logics. *Fund. Inform.*, 15:235–254, 1991. doi:10.3233/FI-1991-153-404.

19 Chase Ford, Stefan Milius, and Lutz Schröder. Behavioural preorders via graded monads. In *Logic in Computer Science, LICS 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470517.

20 Chase Ford, Stefan Milius, and Lutz Schröder. Monads on categories of relational structures. In Fabio Gadducci and Alexandra Silva, editors, *Algebra and Coalgebra in Computer Science, CALCO 2021*, volume 211 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CALCO.2021.14.

21 Chase Ford, Stefan Milius, Lutz Schröder, Harsh Beohar, and Barbara König. Graded monads and behavioural equivalence games. In Christel Baier and Dana Fisman, editors, *Logic in Computer Science, LICS 2022*, pages 61:1–61:13. ACM, 2022. doi:10.1145/3531130.3533374.

22 Jonas Forster, Sergey Goncharov, Dirk Hofmann, Pedro Nora, Lutz Schröder, and Paul Wild. Quantitative Hennessy-Milner theorems via notions of density. In Bartek Klin and Elaine Pimentel, editors, *Computer Science Logic, CSL 2023*, volume 252 of *LIPIcs*, pages 22:1–22:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CSL.2023.22.

23 Jonas Forster, Lutz Schröder, and Paul Wild. Quantitative graded semantics and spectra of behavioural metrics. *CoRR*, abs/2306.01487, 2023. doi:10.48550/arXiv.2306.01487.

24 Jonas Forster, Lutz Schröder, Paul Wild, Harsh Beohar, Sebastian Gurke, and Karla Messing. Graded semantics and graded logics for Eilenberg-Moore coalgebras. In Barbara König and Henning Urbat, editors, *Coalgebraic Methods in Computer Science, CMCS 2024*, volume 14617 of *LNCS*, pages 114–134. Springer, 2024. doi:10.1007/978-3-031-66438-0_6.

25 Alessandro Giacalone, Chi-Chang Jou, and Scott Smolka. Algebraic reasoning for probabilistic concurrent systems. In Manfred Broy, editor, *Programming concepts and methods, PCM 1990*, pages 443–458. North-Holland, 1990.

26 Sergey Goncharov, Dirk Hofmann, Pedro Nora, Lutz Schröder, and Paul Wild. Kantorovich functors and characteristic logics for behavioural distances. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures, FoSSaCS 2023*, volume 13992 of *LNCS*, pages 46–67. Springer, 2023. doi:10.1007/978-3-031-30829-1_3.

27 Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Log. Methods Comput. Sci.*, 3(4), 2007. doi:10.2168/LMCS-3(4:11)2007.

28 Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985. doi:10.1145/2455.2460.

29 Bart Jacobs, Alexandra Silva, and Ana Sokolova. Trace semantics via determinization. *J. Comput. Syst. Sci.*, 81(5):859–879, 2015. doi:10.1016/j.jcss.2014.12.005.

30 Manisha Jain, Alexandre Madeira, and Manuel A. Martins. A fuzzy modal logic for fuzzy transition systems. In Amy P. Felty and João Marcos, editors, *Logical and Semantic Frameworks with Applications, LSFA 2019*, volume 348 of *ENTCS*, pages 85–103. Elsevier, 2019. doi:10.1016/j.entcs.2020.02.006.

31 Bartek Klin and Jurriaan Rot. Coalgebraic trace semantics via forgetful logics. *Log. Methods Comput. Sci.*, 12(4), 2016. doi:10.2168/LMCS-12(4:10)2016.

32 Yuichi Komorida, Shin-ya Katsumata, Clemens Kupke, Jurriaan Rot, and Ichiro Hasuo. Expressivity of quantitative modal logics : Categorical foundations via codensity and approximation. In *Logic in Computer Science, LICS 2021*, pages 1–14. IEEE, 2021. doi:10.1109/LICS52264.2021.9470656.

33 Barbara König and Christina Mika-Michalski. (metric) bisimulation games and real-valued modal logics for coalgebras. In Sven Schewe and Lijun Zhang, editors, *Concurrency Theory, CONCUR 2018*, volume 118 of *LIPIcs*, pages 37:1–37:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CONCUR.2018.37.

34 Clemens Kupke and Jurriaan Rot. Expressive logics for coinductive predicates. *Log. Methods Comput. Sci.*, 17(4), 2021. doi:10.46298/lmcs-17(4:19)2021.

35 F. E. J. Linton. Some aspects of equational categories. In S. Eilenberg, D. K. Harrison, S. MacLane, and H. Röhrl, editors, *Proceedings of the Conference on Categorical Algebra*, pages 84–94. Springer, 1966. doi:10.1007/978-3-642-99902-4_3.

36 Radu Mardare, Prakash Panangaden, and Gordon D. Plotkin. Quantitative algebraic reasoning. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Logic in Computer Science, LICS 2016*, pages 700–709. ACM, 2016. doi:10.1145/2933575.2934518.

37 Stefan Milius, Dirk Pattinson, and Lutz Schröder. Generic trace semantics and graded monads. In Lawrence S. Moss and Pawel Sobocinski, editors, *Algebra and Coalgebra in Computer Science, CALCO 2015*, volume 35 of *LIPIcs*, pages 253–269. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CALCO.2015.253.

38 Haiyu Pan, Yongming Li, Yongzhi Cao, and Zhanyou Ma. Model checking computation tree logic over finite lattices. *Theor. Comput. Sci.*, 612:45–62, 2016. doi:10.1016/j.tcs.2015.10.014.

39 Dirk Pattinson. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame J. Formal Log.*, 45(1):19–33, 2004. doi:10.1305/NDJFL/1094155277.

40 Amgad Rady and Franck van Breugel. Explainability of probabilistic bisimilarity distances for labelled Markov chains. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures, FoSSaCS 2023*, volume 13992 of *LNCS*, pages 285–307. Springer, 2023. doi:10.1007/978-3-031-30829-1_14.

41 Jurriaan Rot, Bart Jacobs, and Paul Blain Levy. Steps and traces. *J. Log. Comput.*, 31(6):1482–1525, 2021. doi:10.1093/logcom/exab050.

42 Jan J. M. M. Rutten. Relators and metric bisimulations. In Bart Jacobs, Larry Moss, Horst Reichel, and Jan J. M. M. Rutten, editors, *Coalgebraic Methods in Computer Science, CMCS 1998*, volume 11 of *ENTCS*, pages 252–258. Elsevier, 1998. doi:10.1016/S1571-0661(04)00063-5.

43 Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000. doi:10.1016/S0304-3975(00)00056-6.

44 Lutz Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. *Theor. Comput. Sci.*, 390(2-3):230–247, 2008. doi:10.1016/j.tcs.2007.09.023.

45 Lutz Schröder and Dirk Pattinson. Description logics and fuzzy probability. In Toby Walsh, editor, *International Joint Conference on Artificial Intelligence, IJCAI 2011*, pages 1075–1081. IJCAI/AAAI, 2011. doi:10.5591/978-1-57735-516-8/IJCAI11-184.

46 Franck van Breugel and James Worrell. A behavioural pseudometric for probabilistic transition systems. *Theor. Comput. Sci.*, 331(1):115–142, 2005. doi:10.1016/j.tcs.2004.09.035.

47 Rob J. van Glabbeek. The linear time - branching time spectrum I. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, pages 3–99. North-Holland/Elsevier, 2001. doi:10.1016/b978-044482830-9/50019-9.

48 Paul Wild and Lutz Schröder. Characteristic logics for behavioural metrics via fuzzy lax extensions. In Igor Konnov and Laura Kovács, editors, *Concurrency Theory, CONCUR 2020*, volume 171 of *LIPIcs*, pages 27:1–27:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CONCUR.2020.27.

49 Paul Wild and Lutz Schröder. A quantified coalgebraic van Benthem theorem. In Stefan Kiefer and Christine Tasson, editors, *Foundations of Software Science and Computation Structures, FOSSACS 2021*, volume 12650 of *LNCS*, pages 551–571. Springer, 2021. doi:10.1007/978-3-030-71995-1_28.

50 Paul Wild and Lutz Schröder. Characteristic logics for behavioural hemimetrics via fuzzy lax extensions. *Log. Methods Comput. Sci.*, 18(2), 2022. doi:10.46298/lmcs-18(2:19)2022.

51 Paul Wild, Lutz Schröder, Dirk Pattinson, and Barbara König. A van Benthem theorem for fuzzy modal logic. In Anuj Dawar and Erich Grädel, editors, *Logic in Computer Science, LICS 2018*, pages 909–918. ACM, 2018. doi:10.1145/3209108.3209180.

52 Thorsten Wißmann, Stefan Milius, and Lutz Schröder. Explaining behavioural inequivalence generically in quasilinear time. In Serge Haddad and Daniele Varacca, editors, *Concurrency Theory, CONCUR 2021*, volume 203 of *LIPIcs*, pages 32:1–32:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.CONCUR.2021.32.

53 James Worrell. Coinduction for recursive data types: partial orders, metric spaces and omega-categories. In Horst Reichel, editor, *Coalgebraic Methods in Computer Science, CMCS 2000*, volume 33 of *ENTCS*, pages 337–356. Elsevier, 2000. doi:10.1016/S1571-0661(05)80356-1.

54 Hengyang Wu, Taolue Chen, Tingting Han, and Yixiang Chen. Bisimulations for fuzzy transition systems revisited. *Int. J. Approx. Reason.*, 99:1–11, 2018. doi:10.1016/j.ijar.2018.04.010.

55 Hengyang Wu, Yixiang Chen, Tian-Ming Bu, and Yuxin Deng. Algorithmic and logical characterizations of bisimulations for non-deterministic fuzzy transition systems. *Fuzzy Sets Syst.*, 333:106–123, 2018. doi:10.1016/j.fss.2017.02.008.

The Lambda Calculus Is Quantifiable

Valentin Maestracci
I2M, Université d'Aix-Marseille, France

Paolo Pistone
LIP, Université Claude Bernard Lyon 1, France

Abstract

In this paper we introduce several quantitative methods for the lambda-calculus based on partial metrics, a well-studied variant of standard metric spaces that have been used to metrize non-Hausdorff topologies, like those arising from Scott domains. First, we study quantitative variants, based on program distances, of sensible equational theories for the λ-calculus, like those arising from Böhm trees and from the contextual preorder. Then, we introduce applicative distances capturing higher-order Scott topologies, including reflexive objects like the D_∞ model. Finally, we provide a quantitative insight on the well-known connection between the Böhm tree of a λ-term and its Taylor expansion, by showing that the latter can be presented as an isometric transformation.

2012 ACM Subject Classification Theory of computation → Lambda calculus; Theory of computation → Program semantics

Keywords and phrases Lambda-calculus, Scott semantics, Partial metric spaces, Böhm trees, Taylor expansion

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.34

Related Version *Full Version*: https://arxiv.org/abs/2411.11809

Funding *Paolo Pistone*: Research has been funded by the French project ANR-23-CPJ1-0054-01.

1 Introduction

Two notions of program approximation. One of the fundamental goals of program semantics is to understand when two different programs compute *the same* function. This is why, since its origins, the semantics of the λ-calculus, the mathematical foundation for higher-order programming languages, has been focused on the problem of *program equivalence*. Indeed, λ-*theories*, the equational theories of the λ-calculus, constitute one of the pillars of the mathematical theory behind this much studied language, ranging from more operational theories, like β-equivalence, to more observational ones, like contextual equivalence.

Actually, several well-known denotational models of the λ-calculus are not just the source for some λ-theory, but they also provide a *topological* point of view on them: the interpretations of the λ-calculus via Böhm trees, Scott domains or the Taylor expansion, involve spaces whose objects can be seen as limits of "finite" approximants, as well as *continuous* functions between such spaces, that is, functions commuting with such limits. In this way, the λ-theory induced by a topological model is associated with a notion of approximation, in the sense that a program is equivalent to another program whenever the net of finite approximants of the first converges to the second.

However, in general computer science, the approximation of a program is more commonly thought as the fact of computing values which are *close* (possibly *up to* some probability) to those produced by the program itself. By the way, the replacement of computationally expensive algorithms by more efficient, but somehow inaccurate, ones, is pervasive in all domains involving probabilistic or numerical methods. This has motivated, in the last few years, a rise of interest towards semantic approaches to functional languages focused, rather than on program equivalence, on notions of *program similarity* [37, 17, 11, 14, 16]. In these

approaches, each type is endowed with a *pseudo-metric*, measuring the amount to which two programs behave in a similar, although non necessarily equivalent, way, and thus providing ways to estimate the errors produced by approximated optimization methods. At the same time, since any pseudo-metric induces an equational theory over programs, namely the one formed by all the pairs of programs which are at *no* distance the one from the other, this approach can be seen as a way to enrich, or "topologize", well-established notions of program equivalence.

Quantifying λ-theories via partial metrics. In a sense, the overall goal of this paper is to reconcile these two, apparently different, ways to look at program approximations, by developing metric counterparts to well-established methods for the λ-calculus, thus providing ways to enrich λ-theories with notions of program similarity.

One reason why one could wish to approximate λ-theories by metrics is computational: while equational theories are generally undecidable, equivalences and, as we'll see, distances of finite approximants can often be computed effectively. Could one thus express the equivalence between two terms as the fact that the distance between their respective approximants gets closer and closer to zero? This amounts to requiring that the limits in the topology T_1 generating the λ-theory are *also* limits in the topology T_2 generated by some program pseudo-metric. In other words, that T_1 is *finer* than T_2.

At the same time, since program metrics are generally undecidable as well, could the distances between two programs be themselves approximated by looking at the (computable) distances between their approximants? This amounts to requiring, conversely, that the metric limits, that is, the limits in T_2, are *also* limits in the topology T_1 inducing the λ-theory. In other words, that T_2 is *finer* than T_1.

All this sums up to the following question: can we make the topology arising from the semantics and the topology arising from the metric *coincide*? At first, one would tend to answer no: for instance, while the topology of a metric space is *always* Hausdorff, the topologies arising from the semantics of the λ-calculus (e.g. Scott domains) are not. Nevertheless, there is a well-known reply to this answer, namely *partial metrics* [8, 9, 28, 42, 40, 38], a well-studied variant of standard metrics developed in connection with ideas from program semantics. A partial metric differs from a standard metric in that the self-distances $p(x, x)$ need *not* be zero; correspondingly, one has a *stronger* triangular law of the form $p(x, y) \leq p(x, z) + p(z, y) - p(z, z)$, taking into account the self-distance of the middle point z. As a consequence, distinct points will *not* have disjoint neighborhoods, as soon as the self-distance of one makes it "too thick", so to say, to separate it from the other.

In fact, *any* continuous domain with a countable basis is *quantifiable* (a term we borrow from [40]) by a partial metric. This means that its Scott topology does coincide with the topology induced by the metric [9, 35, 42, 40, 41], so that the limits in the Scott topology agree with the metric limits and viceversa.

While the quantification of domains via partial metrics has been well-known for a while, the application of such results to the study of higher-order languages has not been much explored so far. We do it in this paper: we introduce quantitative variants for well-known methods like Böhm trees, Scott domains and the Taylor expansion, based on partial metrics, at the same time providing ways to approximate their associated λ-theories.

Contributions. In this paper we show that several well-known approaches to the study of the λ-calculus can be *quantified*, that is, enriched with metric reasoning on program similarity. Our contributions can be summarized as follows:

- We introduce a partial metric variant of the notion of *sensible* λ-theory [5] and we explore quantitative versions of well-known theories like those arising from Böhm trees and the contextual preorder.
- We introduce *applicative* partial metrics, and we illustrate their use to quantify higher-order Scott domains as well as reflexive objects, like Scott's model D_∞. This opens the way to apply metric techniques to typed or non-typed higher-order languages.
- Finally, we study the *Taylor expansion* of λ-terms [18, 19, 4], a powerful technique inspired by ideas from linear logic, and show that it can be presented as an isometric transformation from Böhm trees to sets of resource λ-terms, thus refining the well-known *commutation theorem* [20], that relates the corresponding λ-theories.

Outline. In Section 2 we recall basic notions about partial metric spaces. In Section 3 we introduce quantitative variants of sensible λ-theories. In Section 4 we investigate the quantification of higher-order Scott domains via applicative distances, and in Section 5 we apply these ideas to the quantification of reflexive objects. In Section 6 we discuss the Taylor expansion. Finally, in Section 7 we indicate related work as well as a few future directions.

2 Partial Metric Spaces

In this section we introduce partial metric spaces and we illustrate a few examples.

▶ **Definition 1.** *A function $p : X \times X \to [0, +\infty]$ is called a* partial metric *(PM) when it satisfies the following axioms:*
(P1) $p(x, x) \leq p(x, y)$,
(P2) *If* $p(x, x) = p(x, y) = p(y, y)$ *then* $x = y$,
(P3) $p(x, y) = p(y, x)$,
(P4) $p(x, y) \leq p(x, z) + p(z, y) - p(z, z)$.
p is called a partial pseudo-metric *(PPM) when it satisfies P1, P3 and P4, and a* partial ultra-metric *(PUM) when it satisfies P1, P3 and*
(P4U) $p(x, y) \leq \max\{p(x, z), p(z, y)\}$.

While in a standard (pseudo-)metric space each point is at distance 0 from itself, condition P1 states that the distance of a point from itself is only required to be *smaller* than its distance from any other point. Condition P2 adapts the usual separation condition $d(x, y) = 0 \Rightarrow x = y$ to non-zero self-distances, and distinguishes PMs from PPMs. Condition P3 is the usual symmetry, while P4 is a strengthening of the triangular law of metric spaces, that also takes into account the possibly non-zero self-distance of the middle point z. P4U is as for standard ultra-metric spaces. Notice that P4U implies P4, so PUMs are indeed PPMs. Notice that a PPM (resp. a PUM, a PM) p always induces a pseudo-metric (resp. a ultra-metric, a metric) by the formula $d_p(x, y) := 2p(x, y) - p(x, x) - p(y, y)$.

A PPM p induces a preorder on X defined by $x \leq_p y$ iff $p(x, y) \leq p(x, x)$. Notice that this implies by P1 that $p(x, y) = p(x, x)$. When p is a PM the preorder $\leq_p$ is indeed an order. With respect to this preorder, p is *antimonotonic* in the sense that $x \leq_p x'$ implies $p(x', y) \leq p(x, y)$. Intuitively, the higher points are those with smaller self-distance.

The symmetrization of the preorder $\leq_p$ yields an equivalence relation $\simeq_p$. In the next section we will indeed explore the use of partial metrics as ways of approximating preorders or equivalence relations on λ-terms. We will say that a PPM p *quantifies* an order (resp. an equivalence) relation over X when this relation coincides with $\leq_p$ (resp. $\simeq_p$).

Let us now talk about the topology induced by a PPM.

▶ **Definition 2** (open balls, topology). *Let p be a PPM on X. For any $x \in X$ and $\epsilon \in (0, +\infty)$, the* open ball of center x and radius ϵ *is the set $B^p_\epsilon(x) = \{y \in X \mid p(y,x) < p(x,x) + \epsilon\}$. The* topology of p*, noted $\mathcal{O}_p(X)$, is formed by all subsets $U \subseteq X$ which are unions of open balls.*

Recall that, by P1, the distance between two points x, y is always greater or equal than the self-distances $p(x,x), p(y,y)$. We could equivalently define open balls as for standard metric spaces, i.e. $B^p_\epsilon(x) = \{y \in X \mid p(y,x) < \epsilon\}$, but this would make $B^p_\epsilon(x)$ *empty* whenever $\epsilon \leq p(x,x)$. Open balls are upper: if $y \in B^p_\epsilon(x)$ and $y \leq_p y'$, by antimonotonicity we deduce $p(y',x) \leq p(y,x) < p(x,x) + \epsilon$, whence $y' \in B^p_\epsilon(x)$. As a consequence, all open sets $U \in \mathcal{O}_p(X)$ are upper.

Contrarily to standard metric spaces, the topology $\mathcal{O}_p(X)$ is not in general Hausdorff: suppose x, y are distinct points such that $x \leq_p y$; since any open set containing x must also contain y, there can be no *disjoint* open sets U, V such that $x \in U$ and $y \in V$. In some cases, as we'll see, $\mathcal{O}_p(X)$ may coincide with the Scott topology induced by the order $\leq_p$.

In Sections 4 and 5 we will explore the use of partial metrics as ways of approximating (Scott) topologies on λ-terms. We will say that a PPM p *quantifies* a topology $\mathcal{O}(X)$ over X when $\mathcal{O}(X) = \mathcal{O}_p(X)$.

Continuous functions between PPMs can be defined in the usual topological sense: given PPMs p, p', respectively on X and X', a function $f : X \to X'$ is p, p'*-continuous* when f^{-1} sends open sets in $\mathcal{O}_{p'}(X')$ onto open sets in $\mathcal{O}_p(X)$. There is an equivalent ϵ/δ-definition: f is p, p'-continuous if for all $x \in X$ and $\epsilon > 0$, there exists $\delta > 0$ such that $f(B^p_\delta(x)) \subseteq B^{p'}_\epsilon(f(x))$.

We compare different PPMs on a set X by relating the associated topologies:

▶ **Definition 3.** *Given two PPMs p, p' on X, we say that p is* finer than p' *(noted $p \sqsubseteq p'$) when $\mathcal{O}_{p'}(X) \subseteq \mathcal{O}_p(X)$.*

Equivalently, $p \sqsubseteq p'$ when the identity map $\mathrm{id}_X : X \to X$ is p, p'-continuous, i.e. every open p'-ball contains an open p-ball around any of its points.

We conclude this short presentation with a few examples.

▶ **Example 4** (Sierpinski space). The simplest example of a non-Hausdorff topology that is quantified by a partial metric is the *Sierpinski space* $S = \{0, 1\}$, with the Scott topology $\mathcal{O}_\sigma(S) = \{\emptyset, \{1\}, \{0,1\}\}$ induced by the order $0 \leq 1$. Define the PM s on S by $s(0,0) = s(0,1) = 1$ and $s(1,1) = 0$. Notice how this implies $0 \leq_s 1$. Since 0 has self-distance 1, the unique open balls are indeed $\emptyset$, $\{1\}$ and $\{0,1\}$, that is, $\mathcal{O}_\sigma(S) = \mathcal{O}_s(S)$.

▶ **Example 5** (Intervals). The closed intervals of $\mathbb{R}$, noted $\mathbf{I}(\mathbb{R})$, admit the PM $p_{\mathrm{int}}(I_1, I_2) := \inf\{|b - a| \mid I_1 \cup I_2 \subseteq [a,b]\}$, which is the size of the smallest interval containing I_1 and I_2. The order defined by the metric here is intuitive, it is reverse inclusion/the Scott information order: $I \leq_{p_{int}} J$ iff $p_{\mathrm{int}}(I,J) \leq p_{\mathrm{int}}(I,I)$ iff $J \subseteq I$. The more information one has, the higher. This example explains the choice of the word "partial": an interval, in term of Scott topology, represents an information on a partial execution: we have yet to discover the precise real number that we are computing. By contrast, the total elements will be those with self distance 0 (the ones where p behaves like a regular metric), i.e. of the form $\{r\}$, a complete information, of a terminated execution.

▶ **Example 6** (Labeled trees). Let $\Sigma\mathrm{Tree}_{\leq\infty}$ be the set of (non necessarily finite) finitely branching Σ-labeled trees, where Σ is a countable set of labels. For any $\alpha \in \Sigma\mathrm{Tree}_{\leq\infty}$, let $|\alpha| \in \mathbb{N} \cup \{\infty\}$ indicate the *height* of α. For any $n \in \mathbb{N}$, let α_n be the finite tree obtained by truncating all paths of α at length n, if $|\alpha| \geq n$, and be undefined otherwise. We write $\alpha_n \triangleq \beta_n$ to indicate that α_n and β_n are both definite and equal, and $\alpha_n \not\triangleq \beta_n$ for its negation. For any $\alpha, \beta \in \Sigma\mathrm{Tree}_{\leq\infty}$, define $\mathrm{div}(\alpha,\beta) := \inf\{n \mid \alpha_n \triangleq \beta_n \text{ and } \alpha_{n+1} \not\triangleq \beta_{n+1}\}$.

The standard tree (ultra-)metric d_{tree} is defined by $d(\alpha,\beta)=0$ if $\alpha=\beta$ and $2^{-\text{div}(\alpha,\beta)}$ otherwise. We obtain instead a PUM by simply letting $p_{\text{tree}}(\alpha,\beta) := 2^{-\text{div}(\alpha,\beta)}$ (where it is intended that $2^{-\infty}=0$). For a finite tree α, its self-distance is $p_{\text{tree}}(\alpha,\alpha)=2^{-|\alpha|}$, while $p_{\text{tree}}(\alpha,\alpha)=0$ holds iff α has infinite height. Also this case suggests that finite trees are seen as "partial" objects, while the infinite trees are the "total" ones. Indeed, p_{tree}, unlike d_{tree}, quantifies the Scott topology on $\Sigma\text{Tree}_{\leq\infty}$ (see Section 4).

3 Quantifying λ-Theories

In this section we introduce quantitative variants, based on partial metrics, of sensible λ-theories that arise from well-studied models of the untyped lambda-calculus, that is, the theory of Böhm trees and the theory of contextual equivalence. Moreover, we lift several properties of such equational theories to the corresponding notion of program similarity.

λ-PPMs. Let us first recall the standard notion of λ-theory [5].

▶ **Definition 7.** *A* λ-theory T *is an equivalence relation* $\simeq_T$ *on the set* Λ *of all* λ*-terms satisfying the rules below:*
(congr1) $M \simeq_T N \Rightarrow MP \simeq_T NP$,
(congr2) $M \simeq_T N \Rightarrow PM \simeq_T PN$,
(ξ) $M \simeq_T N \Rightarrow \lambda x.M \simeq_T \lambda x.N$,
(β) $(\lambda x.M)N \simeq_T M[N/x]$.
A λ*-theory* T *is said* extensional *when it satisfies the rule* (η)*:*
(η) $M \simeq_T \lambda x.Mx$.
and sensible *when it equates all unsolvable terms and does not equate a solvable and an unsolvable term.*

Notice that a sensible theory T must be consistent: it cannot equate *all* terms.

A λ-theory may either arise from an operational theory (e.g. β- and $\beta\eta$-reduction) or be induced by a model (as the theory formed by all equations between terms that are interpreted by the same entity in the model). While there exists a continuum of different λ-theories, beyond the theories of β and $\beta\eta$-equivalence (respectively, the smallest λ-theory and the smallest extensional λ-theory), very few theories have been studied in depth. Indeed, all most common denotational models of the untyped λ-calculus induce one of the two sensible theories $\mathcal{B}$, and $\mathcal{H}^*$, that we consider below.

We now introduce a quantitative variant of λ-theories. Let us first recall that a point x in a topological space X is said *generic* when its closure is X or, equivalently, all its neighborhoods are dense in X. For instance, 0 is generic in the Sierpinski space S. In the case of PPM we have the following:

▶ **Lemma 8.** *x is generic in the topology $\mathcal{O}_p(X)$ iff $x \leq_p y$ holds for all $y \in X$.*

Proof. Call x generic for p if $x \leq_p y$ (that is, $p(y,x)=p(x,x)$) holds for all $y \in X$. x is generic for p iff the only open ball centered at x is X: from $p(y,x)=p(x,x)$ it follows that for any $\epsilon>0$, $y \in B_\epsilon(x)$, that is, $B_\epsilon(x)=X$; conversely, if any open ball centered at x is equal to X, then, for all $\epsilon>0$, $p(y,x)<p(x,x)+\epsilon$, which implies $p(y,x)\leq p(x,x)$ and thus $p(y,x)=p(x,x)$ by P1.

Now, if x is generic for p, then any open set U containing x must contain some open ball $B_\epsilon(x)$, which forces $U=X$ since $B_\epsilon(x)=X$, so x is generic in $\mathcal{O}_p(X)$. Conversely, if x is generic in $\mathcal{O}_p(X)$, then for any $\epsilon>0$, the closure of $B_\epsilon(x)$ is X. This implies that for all $\epsilon>0$, $p(y,x)\leq p(x,x)+\epsilon$, and thus that $p(y,x)=p(x,x)$, so x is generic for p. ◀

▶ Remark 9. Generic points are indistinguishable: if x and y are both generic for p, then from $p(y,y) = p(x,y) = p(x,x)$ it follows that $x \simeq_p y$. Conversely, if x is generic and y is not, then, $x \not\simeq_p y$: if $x \simeq_p y$, then, since $p(y,x) = p(y,y)$, for all z, $p(y,z) \leq p(y,x) + p(x,z) - p(x,x) = p(y,x) + p(x,x) - p(x,x) = p(y,x) = p(y,y)$, so y would be generic as well.

▶ **Definition 10** (λ-PPM). *A pseudo-partial metric p over Λ is called a* λ-PPM *(resp. an* extensional λ-PPM*) if the following hold:*

- $\simeq_p$ *is a λ-theory (resp. an extensional λ-theory);*
- *all contexts $\mathsf{C}[-]$ correspond to p-continuous maps $\Lambda \to \Lambda$.*

p is called sensible *if all unsolvable terms are generic while no solvable term is.*

Observe that we do not require contexts to be *non-expansive* (or 1-Lipschitz), as in other standard metric approaches [37, 17, 15], but just continuous. Also notice that, by Remark 9, a sensible PPM p must satisfy $M \simeq_p N$ for all unsolvable terms M, N, and $M \not\simeq_p N$ for M unsolvable and N solvable: the associated λ-theory $\simeq_p$ is thus sensible.

In the rest of this section we introduce λ-PPMs quantifying the λ-theories $\mathcal{B}$ and $\mathcal{H}^*$.

Böhm Trees. The interpretation of λ-terms as Böhm trees is one of the fundamental tools in the λ-calculus. The Böhm tree $\mathcal{B}(M)$ of a λ-term M is a $(\Lambda \cup \{\bot\})$-labeled tree defined *co-inductively* as follows:

- if M reduces to $\lambda x_1.\ldots.\lambda x_m.xM_1\ldots M_n$, then $\mathcal{B}(M)$ has a root with label $\lambda x_1.\ldots.\lambda x_m.x$ and n subtrees $\mathcal{B}(M_1), \ldots, \mathcal{B}(M_n)$;
- otherwise, $\mathcal{B}(M)$ only consists of the root, with label $\bot$.

An alternative presentation of $\mathcal{B}(M)$ is via *partial terms*, which are λ-terms in normal form, enriched with the constant $\bot$ and rules $\lambda x.\bot \to \bot$, $\bot M \to \bot$. We note these partial terms $A, B, \ldots$. The set $\mathcal{A}$ of partial terms is ordered by the contextual closure $\preceq$ of the relation generated by $\bot \preceq A$, for all $A \in \mathcal{A}$. Partial terms correspond straightforwardly to *finite* Böhm trees.

For any λ-term M, let the partial term $M_{\mathcal{A}}$ be defined *inductively* as follows: $M_{\mathcal{A}} = \lambda\vec{x}.y(M_1)_{\mathcal{A}}\ldots(M_n)_{\mathcal{A}}$ if $M = \lambda\vec{x}.yM_1\ldots M_n$, and $M_{\mathcal{A}} = \bot$ if $M = \lambda\vec{x}.(\lambda y.P)M_1\ldots M_{n+1}$. Let $A \leq M$ whenever M β-reduces to M' with $A \preceq M'_{\mathcal{A}}$. We then let $\mathcal{B}(M) = \{A \mid A \leq M\}$. Observe that $\mathcal{B}(M)$ can be seen at the same time as a tree under the relation $\leq$, and the standard tree ordering $\mathcal{B}(M) \preceq \mathcal{B}(N)$ holds precisely when $\mathcal{B}(M)$ is included in $\mathcal{B}(N)$.

The λ-theory $\mathcal{B}$ contains all equations $M \simeq_{\mathcal{B}} N$, where $\mathcal{B}(M) = \mathcal{B}(N)$. $\mathcal{B}$ is sensible but non-extensional (as e.g. $\mathcal{B}(\lambda x.x) \neq \mathcal{B}(\lambda x.\lambda y.xy)$).

We now introduce the corresponding λ-PPM: we measure the distance between λ-terms by comparing their Böhm trees via the tree partial metric.

▶ **Definition 11** (Böhm partial metric). *For any two λ-terms M, N, let*

$$p_{\text{Böhm}}(M,N) := p_{\text{tree}}(\mathcal{B}(M), \mathcal{B}(N)).$$

Observe that $p_{\text{Böhm}}(M,M) = 0$ iff $\mathcal{B}(M)$ is infinite. It is not difficult to check that $p_{\text{Böhm}}$ captures the theory $\mathcal{B}$:

▶ **Proposition 12.** *$M \leq_{p_{\text{Böhm}}} N$ iff $\mathcal{B}(M) \leq \mathcal{B}(N)$, and thus $M \simeq_{p_{\text{Böhm}}} N$ iff $M \simeq_{\mathcal{B}} N$.*

As discussed in Section 4, $p_{\text{Böhm}}$ captures the Scott topology of Böhm trees. This proves that contexts are continuous, and thus that $p_{\text{Böhm}}$ is a λ-PPM. Moreover, since $p_{\text{tree}}(\bot, \alpha) = 1$, the unsolvable terms are generic, while, for any solvable term M, $p_{\text{Böhm}}(M,M) < 1$ and thus, for any $\epsilon < 1 - p_{\text{Böhm}}(M,M)$, the open ball $B_\epsilon^{p_{\text{Böhm}}}(M)$ does not contain the term $\lambda x.M$ (since $p_{\text{Böhm}}(M, \lambda x.M) = 1 > p_{\text{Böhm}}(M,M) + \epsilon$).

▶ Remark 13. While the theory $\mathcal{B}$ is Π^0_2-complete, the distances $p_{\text{tree}}(A, B)$ are effectively computable whenever A, B are *finite* trees (equivalently, partial terms). Moreover, to check that $p_{\text{Böhm}}(M, N) < \epsilon$, it is necessary and sufficient to find approximants $A \leq M$ and $B \leq N$ such that $p_{\text{tree}}(A, B) < \epsilon$.

Contextual equivalence. We now consider the theory arising from *contextual equivalence*. Let $M \sqsubseteq_{\text{ctx}} N$ if for all context $\mathtt{C}[-]$, if $\mathtt{C}[M]$ is solvable, then $\mathtt{C}[N]$ is solvable. The theory $\mathcal{H}^*$ contains all equations $M \simeq_{H^*} N$ where $M \sqsubseteq_{\text{ctx}} N$ and $N \sqsubseteq_{\text{ctx}} M$ both hold. It is extensional and sensible, and is indeed the *maximum* sensible theory.

To quantify $\mathcal{H}^*$ we define the following distance:

▶ **Definition 14** (contextual partial metric). *For all terms M, N, we define*

$$p_{\text{ctx}}(M, N) = \sum_{n=0}^{\infty} \left\{ \frac{1}{2^n} \;\middle|\; \mathtt{C}_n[M] \text{ is unsolvable or } \mathtt{C}_n[N] \text{ is unsolvable} \right\},$$

where $(\mathtt{C}_n[-])_{n\in\mathbb{N}}$ is an enumeration of all contexts.

The distance $p_{\text{ctx}}(M, N)$ intuitively counts all contexts $\mathtt{C}_n[-]$ that fail on either M or N. In particular, the self-distance $p_{\text{ctx}}(M, M)$ counts the contexts that fail on M.

The following result shows that p_{ctx} captures the contextual preorder:

▶ **Proposition 15.** *$M \leq_{p_{\text{ctx}}} N$ iff $M \sqsubseteq_{\text{ctx}} N$, and thus $M \simeq_{p_{\text{ctx}}} N$ iff $M \simeq_{\mathcal{H}^*} N$.*

For the result above, the choice of the enumeration is irrelevant, as is the choice of the weights $\frac{1}{2^n}$, which could be replaced by arbitrary weights θ_n summing up to 1.

▶ Remark 16. Contrarily to contextual equivalence, which is Π^0_2-complete as well, to check that $N \in B_\epsilon^{p_{\text{ctx}}}(M)$ one does not need to look at the behavior of M and N under *all* contexts. Intuitively, $B_\epsilon^{p_{\text{ctx}}}(M)$ contains all those programs that behave like M on certain *finitely many* contexts. Indeed, $p_{\text{ctx}}(M, N) < p_{\text{ctx}}(M, M) + \epsilon$ means that the contexts on which M does converge and N does not sum up to some value strictly smaller than ϵ. This is true iff N converges on those finitely many contexts $\mathtt{C}_i[-]$, where $2^{-(i+1)} \leq \epsilon$, on which M converges.

▶ **Proposition 17.** *p_{ctx} is a sensible extensional λ-PPM.*

Proof. Let us show that contexts yield continuous maps. Take a term M, $\epsilon > 0$ and a context $\mathtt{C}$. We need to find some $\delta > 0$ such that for all $P \in B_\delta^{p_{\text{ctx}}}(M)$, $\mathtt{C}[P] \in B_\epsilon^{p_{\text{ctx}}}(\mathtt{C}[M])$. By Remark 16 there exists a *finite* number of contexts $\mathtt{C}_1, \ldots, \mathtt{C}_k$ such that $\mathtt{C}_i[\mathtt{C}[M]]$ is solvable and $N \in B_\epsilon^{p_{\text{ctx}}}(\mathtt{C}[M])$ iff $\mathtt{C}_i[N]$ is solvable for $i = 1, \ldots, k$. Take m such that for all $i = 1, \ldots, k$, the context $\mathtt{C}_i[\mathtt{C}[-]]$ has an index smaller than m, and let $\delta = 2^{-m}$. Notice that $\mathtt{C}_i[\mathtt{C}[M]]$ is solvable. Moreover, for any term P, if $P \in B_\delta^{p_{\text{ctx}}}(M)$, then $\mathtt{C}_i[\mathtt{C}[P]]$ must be solvable for all $i = 1, \ldots, m$. This implies then that $\mathtt{C}[P] \in B_\epsilon^{p_{\text{ctx}}}(\mathtt{C}[M])$, as desired.

The sensibility of p_{ctx} essentially follows from the well-known *genericity lemma* [5, 2]: if $\mathtt{C}[M]$ is solvable, where M is unsolvable, then $\mathtt{C}[N]$ must be solvable *for all* N; this implies that for any unsolvable M, and for any term N, $p_{\text{ctx}}(M, N) = p_{\text{ctx}}(M, M)$, so M is generic in p_{ctx}. Conversely, if M is solvable, then, for any unsolvable term N, one can easily construct a context $\mathtt{C}$ such that $\mathtt{C}[M]$ reduces to $\lambda x.x$ and $\mathtt{C}[N]$ diverges. This allows us to conclude that $p_{\text{ctx}}(M, N) > p_{\text{ctx}}(M, M)$, and thus that M is not generic in p. ◀

Similarly to the λ-theory $\mathcal{H}^*$, the λ-PPM p_{ctx} is *maximum* among sensible λ-PPMs.

▶ **Proposition 18.** *For any sensible λ-PPM p, $p \sqsubseteq p_{\text{ctx}}$.*

Proof. Let p be a sensible λ-ppm. Consider a term M and $\epsilon > 0$. We must find $\delta > 0$ such that $B^p_\delta(M) \subseteq B^{p_{\mathrm{ctx}}}_\epsilon(M)$. By Remark 16 there exists a finite number of contexts $\mathtt{C}_1, \ldots, \mathtt{C}_k$ such that $\mathtt{C}_i[M]$ is solvable and $N \in B^{p_{\mathrm{ctx}}}_\epsilon(M)$ iff $\mathtt{C}_i[N]$ is solvable for $i = 1, \ldots, k$.

Fix an $i \leq k$ and let $Q_i = \mathtt{C}_i[M]$. Since Q_i is solvable and p is sensible, we can find an open set U_i containing Q_i and *not* containing any unsolvable term. Since p is a λ-PPM, $\mathtt{C}_i$ corresponds to a continuous function, and thus $\mathtt{C}_i^{-1}(U_i)$ contains some open ball $B^p_{\delta_i}(M)$. Let $\delta = \min_i \delta_i$: if $P \in B^p_\delta(M)$, then for all $i = 1, \ldots, k$, $\mathtt{C}_i[P] \in U_i$, so it must be solvable. We conclude then that $P \in B^{p_{\mathrm{ctx}}}_\epsilon(M)$. ◀

▶ Remark 19. That $p_{\mathrm{Böhm}} \sqsubset p_{\mathrm{ctx}}$ can be easily seen directly: the elements of $B^{p_{\mathrm{ctx}}}_\epsilon(M)$ are those which converge on a finite number of contexts $\mathtt{C}_1, \ldots, \mathtt{C}_k$ on which M converges too (cf. Remark 16). For any such context $\mathtt{C}_i$, the convergence of $\mathtt{C}_i[M]$ to a head normal form only depends on the exploration of a *finite* portion of $\mathcal{B}(M)$, say up to height m_i. Letting $m = \max_i\{m_i\}$ and $\delta = 2^{-m}$, we have then that $B^{p_{\mathrm{Böhm}}}_\delta(M) \subseteq B^{p_{\mathrm{ctx}}}_\epsilon(M)$. The converse inclusion $p_{\mathrm{ctx}} \sqsubseteq p_{\mathrm{Böhm}}$ cannot hold: any open ball $B^{p_{\mathrm{Böhm}}}_\epsilon(I)$ around $I = \lambda x.x$ that does not contain its η-expansion $\lambda x.\lambda y.xy$ contains *no* open p_{ctx}-ball around I.

Other well-known characterizations of $\mathcal{H}^*$ exist, which suggest different ways to quantify this theory. One is in terms of the so-called *Nakajima trees* (cf. [5], Ex. 19.4.4, p. 511): these are a variant of Böhm trees that are invariant under the η-rule. By adapting the tree partial metric one could then obtain another partial metric p_{Nakajima} that quantifies $\mathcal{H}^*$.

Moreover, the theory $\mathcal{H}^*$ is induced by a large class of denotational models of the λ-calculus (cf. [31]), including in particular the models based on Scott domains, that we discuss in Sections 4 and 5, or the relational model from [6], to which the techniques illustrated in those sections can be easily adapted.

4 Quantifying Scott Domains

As discussed in the introduction, the λ-theories like $\mathcal{B}$ or $\mathcal{H}^*$ are induced by topological models, based on Scott domains, which provide notions of approximant for λ-terms. In this section, after discussing the connection between partial metrics and Scott domains, we introduce applicative PPMs as a means to capture domains of Scott-continuous functions, and we illustrate how this leads to quantify topological models of typed λ-calculi.

Scott Domains via Partial Metrics. Let us recall some basic terminology about dcpos and Scott domains.

A partially ordered set $(X, \leq)$ is a *dcpo* (directed complete partial order) if all directed subsets of X admit a least upper bound. The *way below* relation $\ll$ on a dcpo is defined by $x \ll y$ iff for all directed subset $\Delta \subseteq X$, $y \leq \bigvee \Delta$ implies $x \leq d$, for some $d \in \Delta$. A point $x \in X$ is said *compact* if $x \ll x$. A *basis* for a dcpo X is a subset $B \subseteq X$ such that for any $x \in X$, the set $\Delta = \{y \in B \mid y \ll x\}$ is directed and $x = \bigvee \Delta$. A dcpo is said *continuous* if it has a basis and *algebraic* if it has a basis formed of compact elements. A *domain* is a continuous dcpo with a countable basis. A domain X is *bounded complete* if for any finite set $Y \subseteq_{\mathrm{fin}} X$, if an upper bound of Y exists in X, then $\bigvee Y$ exists in X. A bounded complete and algebraic domain is called a *Scott domain*.

The *Scott topology* $\mathcal{O}_\sigma(X)$ on a partially ordered set $(X, \leq)$ has open sets being upper subsets $U \subseteq X$ which are *finitely accessible*: $x \in U$ implies $y \in U$ for some $y \ll x$. A function $f : X \to Y$ between dcpos is said *continuous* iff f is monotone and commutes with the lubs of directed subsets, that is, for all directed $\Delta \subseteq X$, $f(\bigvee \Delta) = \bigvee f(\Delta)$. This is equivalent to asking f to be continuous, in the usual sense, with respect to the Scott topology. The category of bounded complete domains and continuous functions is cartesian closed (cf. [1]).

Let us specify what it means for a dcpo to be *quantified* by a partial metric.

▶ **Definition 20.** *A dcpo $(X, \leq)$ is* quantified by a PM p *when its associated Scott topology is quantified by p, that is, when $\mathcal{O}_p(X) = \mathcal{O}_\sigma(X)$.*

Beyond the Sierpinski space S, also the other two dcpos from Section 2 are quantified by the associated PMs (proofs are in the long version):

▶ **Proposition 21.** *The interval dcpo $\mathbf{I}(\mathbb{R})$ is quantified by p_{int} (cf. Example 5). The domain $\Sigma\text{Tree}_{\leq\infty}$ of Σ-trees is quantified by p_{tree} (cf. Example 6).*

When p quantifies a dcpo $(X, \leq)$, the order $\leq_p$ coincides with the order $\leq$ of the dcpo.

▶ **Lemma 22.** *Suppose the dcpo $(X, \leq)$ is quantified by p. Then $\leq$ coincides with $\leq_p$.*

Proof. $\leq$ coincides with the *specialization order* $x \leq^{\mathcal{O}_\sigma(X)} y \Leftrightarrow \forall U \in \mathcal{O}_\sigma(X)(x \in U \Rightarrow y \in U)$; similarly, $\leq_p$ coincides with the specialization order $x \leq^{\mathcal{O}_p(X)} y \Leftrightarrow \forall U \in \mathcal{O}_p(X)(x \in U \Rightarrow y \in U)$. From $\mathcal{O}_\sigma(X) = \mathcal{O}_p(X)$ we deduce that the two specialization orders coincide, and thus $\leq$ and $\leq_p$ as well. ◀

However, checking that a partial metric p captures the order of the dcpo is *not* in general enough to deduce that p quantifies the dcpo, as shown by Example 24 below. The following proposition provides necessary (but not sufficient) conditions.

▶ **Proposition 23.** *Let $(X, \leq)$ be a continuous dcpo and p a partial metric on X such that $\leq$ coincides with $\leq_p$. Then the following conditions are equivalent:*

1. *$\mathcal{O}_p(X) \subseteq \mathcal{O}_\sigma(X)$;*
2. *open p-balls are finitely accessible;*
3. *p is Scott-continuous (as a map towards the dcpo $([0, +\infty], \geq)$).*

Proof.

(1 $\Leftrightarrow$ 2) Since the open balls are upper sets, these are Scott open iff they are finitely accessible.

(3 $\Rightarrow$ 2) p is Scott continuous when for all $x \in X$ and directed subset $\Delta \subseteq X$ one has $p(x, \bigvee \delta) = \inf_{d \in \delta} p(x, d)$. Suppose p is continuous and let $y \in B_\epsilon(x)$. We need to show that there exists $y' \ll y$ such that $y' \in B_\epsilon(x)$. This implies that for some $\epsilon' < \epsilon$, $p(y, x) < p(x, x) + \epsilon'$. Since p is continuous and $y = \bigvee\{z \mid z \ll y\}$ we have then $\inf\{p(z, x) \mid z \ll y\} = p(y, x) < p(x, x) + \epsilon'$. This implies in turn that for some $y' \ll y$, $p(y', x) \leq p(x, x) + \epsilon' < p(x, x) + \epsilon$, that is, $y' \in B_\epsilon(x)$.

(2 $\Rightarrow$ 3) Suppose that open p-balls are finitely accessible, hence Scott open. Let $\Delta \subseteq X$ be a directed set and $x \in X$. We need to prove that $p(x, \bigvee \Delta) = \inf_{d \in \Delta} p(x, d)$. Observe that the "$\leq$" direction directly follows from $d \leq \bigvee \Delta$. To prove the "$\geq$" direction we argue as follows: let $p(x, \bigvee \Delta) = p(x, x) + \delta$, with $\delta \in \mathbb{R}_{\geq 0}$. Let $\delta' > \delta$, so that we have $\bigvee \Delta \in B_{\delta'}(x)$. Since $B_{\delta'}(x)$ is Scott-open, there exists $w \ll \bigvee \Delta$ such that $w \in B_{\delta'}(x)$. From $w \ll \bigvee \Delta$ it follows that, for some $d \in D$, $w \leq d$ holds, whence $p(d, x) \leq p(w, x) < p(x, x) + \delta'$. We have thus proved that for all $\delta' > \delta$ there exists $d \in \Delta$ such that $p(d, x) < p(x, x) + \delta'$, which implies then $\inf_{d \in \Delta} p(d, x) \leq p(x, x) + \delta = p(x, \bigvee \Delta)$. ◀

To check the converse condition $\mathcal{O}_\sigma(X) \subseteq \mathcal{O}_p(X)$, one must show that, given $x \ll y$, one can form open balls around y whose elements all lie way above x. This corresponds to showing that the basic open sets $\Uparrow x = \{y \mid x \ll y\}$ for the Scott topology are metric open.

▶ **Example 24.** We construct a PM on $\Sigma\mathrm{Tree}_{\leq\infty}$ that captures the tree order but fails to quantify its Scott topology. Define a variant q of the tree partial metric as $q(\alpha,\beta) = \frac{1}{2}p_{\mathrm{tree}}(\alpha,\beta) + \frac{1}{4}$ if $\alpha \neq \beta$ or $\alpha = \beta$ is finite, and as $p_{\mathrm{tree}}(\alpha,\beta)$ if $\alpha = \beta$ is infinite. q is still a partial metric and furthermore the order $\leq_q$ coincides with the tree order (and thus with $\leq_p$ as well); now, letting α_n be a directed sequence of finite trees converging to an infinite tree α, we have $\lim_n q(\alpha_n,\alpha) = \frac{1}{4} > 0 = q(\bigvee_n \alpha_n,\alpha)$. Hence q is not Scott-continuous, and by Proposition 23 we have that $\mathcal{O}_q(\Sigma\mathrm{Tree}_{\leq\infty}) \not\subseteq \mathcal{O}_\sigma(\Sigma\mathrm{Tree}_{\leq\infty})$.

▶ Remark 25 (computability of $p(x,y) < \epsilon$). An immediate and useful consequence of the fact that open balls are Scott open is that $p(x,y) < \epsilon$ holds precisely when $p(x',y') < \epsilon$ holds for some approximants $x' \ll x$ and $y' \ll y$. In other words, to verify that y is *close enough* to x it is enough to check that the approximants of y get close enough to the approximants of x. When distances between approximants, as well as the relation $b \ll x$ between a point and an approximant, are computable, the property $p(x,y) < \epsilon$ may be (semi-)decidable, even though the exact values $p(x,y)$ are as hard as computing the λ-theory (usually, Π^0_2 or worse). For instance, in the case of Böhm trees, to check that $p_{\mathrm{Böhm}}(M,N) < 2^{-n}$, it is enough to check that $\mathcal{B}(M)$ and $\mathcal{B}(N)$ coincide up to height n, a property which can be semi-decided.

▶ **Example 26** (ϵ/δ-continuity of contexts)**.** As p_{tree} quantifies the Scott topology of trees (cf. Proposition 21), it quantifies the Scott topology of Böhm trees. From the continuity theorem for Böhm trees (cf. [5]) we deduce then the following: for all context $\mathtt{C}[-]$ and λ-term M and for all $\epsilon > 0$, there exists $\delta > 0$ such that, for all terms P, $p_{\mathrm{Böhm}}(P,M) \leq \delta$ implies $p_{\mathrm{Böhm}}(\mathtt{C}[P],\mathtt{C}[M]) \leq \epsilon$. Another way of stating this is that for all $\mathtt{C}[-]$ and M, for all $n \in \mathbb{N}$ there exists $m \in \mathbb{N}$ such that, if $\mathcal{B}(P)$ and $\mathcal{B}(M)$ are the same up to depth m, then $\mathcal{B}(\mathtt{C}[P])$ and $\mathcal{B}(\mathtt{C}[M])$ are the same up to depth n.

Let us conclude this paragraph by recalling a very general result on the quantifiability of domains, already mentioned in the Introduction:

▶ **Theorem 27** (cf. [40])**.** *Let $(X,\leq)$ be a domain with a countable basis $(b_n)_{n\in\mathbb{N}}$, and let $\theta_n \in (0,1]$ be a sequence of weights such that $\sum_n^\infty \theta_n \leq 1$. Then X is quantified by the partial metric $p^X_{b_n,\theta_n}(x,y) = \sum_{n\in N}\theta_n$, where $N := \{n \mid b_n \not\ll x \text{ or } b_n \not\ll y\}$.*

While Theorem 27 provides a general positive answer to the *quantifiability* problem for domains, the practical usability of metrics like $p^X_{b_n,\theta_n}$ depends on whether the relation $b_n \ll x$ between a point and an approximant, and its negation, are computable.

Applicative distances and the function space. The categories of Scott domains (resp. bounded complete domains) and continuous functions are sub-categories of Top that are, as is well-known, cartesian closed. Using Theorem 27 it is possible to define, on each object of such categories, a partial metric that quantifies its topology. However, in common approaches to higher-order languages (e.g. [17, 25, 16]), one requires distances to be defined in a *compositional* way.

For example, given metric spaces (X,d_X) and (Y,d_Y), a standard way to define a metric on their cartesian product is by letting $d_{X\times Y}(\langle x,y\rangle,\langle x',y'\rangle) = d_X(x,x') + d_Y(y,y')$. Indeed, a similar construction also works for PMs:

▶ **Proposition 28.** *Let X,Y be two Scott domains, quantified, respectively, by the partial metrics p_X,p_Y. Their cartesian product $X\times Y$ is then quantified by the partial metric $p_{X\times Y} := \frac{1}{2}(p_X + p_Y)$.*

We omit the proof of Proposition 28 as it is similar to that of Proposition 30 below (still, the proof can be found in the extended version).

▶ Remark 29. In the following discussion we restrict attention to partial metrics valued in $[0,1]$, rather than on $[0,+\infty]$. This is not a limitation, since for any partial metric p with values in $[0,+\infty]$, the partial metric $p^{\leq 1} : X \times X \to [0,1]$ defined by $p^{\leq 1}(x,y) := \frac{p(x,y)}{1+p(x,y)}$ induces the same topology (cf. [34]).

Let us now consider the function space. Given metric spaces (X,d) and (X',d'), a standard compositional way to define a metric on the space $\mathcal{C}(X,X')$ of continuous functions from X to X' is via the sup-condition $d_{\sup}(f,g) = \sup_{x\in X} d'(f(x),f(x'))$. Notably, when X is compact, $d_{\sup}$ metrizes the *compact-open* topology on $\mathcal{C}(X,X')$. Other compositional metrics on the space of *non-expansive* functions $\mathrm{NExp}(X,X')$, depending on *both* d and d', can be found in the literature [10, 15]. Similar compositional definitions are also found in more operational approaches like e.g. [37, 25].

A common intuition in all these definitions is that two functions are close when their application to close (or even identical) points produces points that are still close. We will call functional distances respecting this idea *applicative distances*.

However, to define an applicative PM on the space of continuous functions, we cannot directly adapt a definition like $d_{\sup}$: unlike for standard (pseudo-)metrics, the sups of a family of PPMs does *not* define a PPMs. This is due to condition P4, which relies in a *contravariant* way on the medium self-distance $p(z,z)$.

Instead, we will rely on the remark that a continuous function $f : X \to Y$ is uniquely determined by its action on the (countably many) elements of a basis of X. This suggests indeed the definition from the Proposition below:

▶ **Proposition 30.** *Let X, Y be two bounded complete domains, quantified, respectively, by the PMs p_X, p_Y, and let $(a_n)_{n\in\mathbb{N}}$ be an enumeration of a basis of X. Then, for all $0 < \theta \leq \frac{1}{2}$, their exponential $X \Rightarrow Y$ is quantified by the PM*

$$p^{\theta}_{X\Rightarrow Y}(f,g) = \sum_{n=1}^{\infty} \theta^n p_Y(f(a_n), g(a_n)). \tag{1}$$

Before proving the result above, let us first discuss the PM $p^{\theta}_{X\Rightarrow Y}$. The distances $p^{\theta}_{X\Rightarrow Y}(f,g)$ are defined by infinite series, which are convergent by our assumption that p_X, p_Y are bounded by 1. However, for any $\epsilon > 0$, the verification that $p^{\theta}_{X\Rightarrow Y}(f,g) < \epsilon$ can be reduced to a *finitary* test:

▶ **Lemma 31.** *For all continuous functions $f, g : X \to Y$, for all $n > 0$, there exists $N \in \mathcal{O}(n)$ such that, if $p_Y(f(a_i), g(a_i)) < 2^{-(n+1)}$ holds for all $i = 1, \ldots, N$, then $p^{\theta}_{X\Rightarrow Y}(f,g) < 2^{-n}$.*

Proof. Let $\epsilon = 2^{-n}$. We must choose N so that $\sum_{i>N}^{\infty} \theta^i < \frac{\epsilon}{2}$. Since $\sum_{n=1}^{\infty} \theta^n \leq 1$, this corresponds to requiring $\sum_{n=1}^{N} \theta^n > 1 - \frac{\epsilon}{2}$, or, equivalently, $\frac{1-\theta^{N+1}}{1-\theta} - 1 > 1 - \frac{\epsilon}{2}$. A few computations yield then the condition $N + 1 > \log\left(3\theta + \theta^2\epsilon + \epsilon\right) \in \mathcal{O}(\log \epsilon)$.

Let us show that under this condition the claim holds. Suppose $p^{\theta}(f(a_i), g(a_i)) \leq \frac{\epsilon}{2}$ holds for $i = 1, \ldots, N$. Then we have

$$p^{\theta}(f,g) = \left(\sum_{k=1}^{N} \theta^n q(f(a_k), g(a_k))\right) + \left(\sum_{k>N}^{\infty} \theta^n q(f(a_k), g(a_k))\right) \leq \left(\sum_{k=1}^{N} \theta^n \frac{\epsilon}{2}\right) + \frac{\epsilon}{2} \leq \epsilon$$

◀

The intuition behind the test above is that for N high enough, the infinite sum $\sum_{n\geq N}^{\infty}\theta^n$ gets too small to actually matter in checking that $p^{\theta}_{X\Rightarrow Y}(f,g)$ is smaller than ϵ, and one is thus reduced to the *finite* sum $\sum_{n=1}^{N}\theta^n p_Y(f(a_n),g(a_n))$. This is indeed a key ingredient in showing that open balls of functions are finitely accessible, and in particular, that if $g\in B_\epsilon(f)$, this only depends on finitely many values of g.

Conversely, from $p^{\theta}_{X\Rightarrow Y}(f,g)<\epsilon$, one can deduce bounds $p_Y(f(a_n),g(a_n))<\theta^{-n}\epsilon$ for *all* $n\in\mathbb{N}$, although such bounds become more and more loose as n increases, due to the exponential scaling factor θ^{-n}.

Let us now turn to the proof of Proposition 30. First, let us recall that, given bounded complete domains X,Y, with countable bases $B(X),B(Y)$, the domain $\mathcal{C}(X,Y)$ admits a countable basis formed by all functions of the form $(\Uparrow a\searrow b)$, where $a\in B(X),b\in B(Y)$, and $(\Uparrow a\searrow b)(x)=b$ in case $a\ll x$, while $(\Uparrow a\searrow b)(x)=\bot$ otherwise.

Importantly, while it is always the case that $f\ll g$ implies $f(x)\ll g(x)$ for all $x\in X$, the converse need *not* hold. Rather, the way below relation can be characterized as follows.

▶ **Lemma 32** (cf. [21]). *For all $f,g\in\mathcal{C}(X,Y)$, $f\ll g$ iff there exists basis elements $a_1,\dots,a_n\in B(X)$ and $b_1,\dots,b_n\in B(Y)$ such that*

- $b_i\ll g(a_i)$,
- $\Uparrow a_i\ll g^{-1}(\Uparrow b_i)$[1], *for all* $i=1,\dots,n$,
- $f\leq\bigvee_{i=1}^{n}(\Uparrow a_i\searrow b_i)$.

We now have all ingredients to prove Proposition 30.

Proof of Proposition 30.

$\mathcal{O}_\sigma(X\Rightarrow Y)\supseteq\mathcal{O}_{p_{X\Rightarrow Y}}(X\Rightarrow Y)$: We have to show that open balls are Scott-open. First observe that open balls are upper sets. We thus only need to show that they are finitely accessible: for all $g\in B_\epsilon(f)$ we must find some $h\ll g$ such that $h\in B_\epsilon(f)$. Let then $g\in B_\epsilon(f)$, so that $p^{\lambda}_{X\Rightarrow Y}(f,g)<p^{\lambda}_{X\Rightarrow Y}(f,f)+\epsilon$. Observe that this implies that we can find positive reals $\theta,\delta>0$ such that $\theta+\delta\leq\epsilon$ and $p^{\lambda}_{X\Rightarrow Y}(f,g)<p^{\lambda}_{X\Rightarrow Y}(f,f)+\delta$. Let N be such that $\sum_{n>N}^{\infty}\lambda^n\leq\frac{\theta}{2}$. For all $n\leq N$, fix some $b_n\in B_{\frac{\theta}{2}}(g(a_n))$ such that $b_n\ll g(a_n)$, and some basis element $c_n\ll a_n$.

Let now $h=\bigvee_{i=1}^{N}(\Uparrow c_i\searrow b_i)$. From $b_i\ll g(a_i)$ it follows that $a_i\in g^{-1}(\Uparrow b_i)$, and thus that $\Uparrow c_i\ll g^{-1}(\Uparrow b_i)$. By Lemma 32 this implies that $h\ll g$. Let us show that $h\in B_\epsilon(f)$. For all $n<N$, we have $p_Y(h(a_n),g(a_n))\leq p_Y(b_n,g(a_n))<p_Y(g(a_n),g(a_n))+\frac{\theta}{2}$, whence, for all $n\leq N$, $p_Y(h(a_n),g(a_n))-p_Y(g(a_n),g(a_n))<\frac{\theta}{2}$. Let's check that $h\in B_\epsilon(f)$:

$$
\begin{aligned}
p^{\lambda}_{X\Rightarrow Y}(h,f)&=\sum_{n=1}^{\infty}\lambda^n p_Y(h(a_n),f(a_n))\\
&\leq\sum_{n=1}^{\infty}\lambda^n\Big(p_Y(h(a_n),g(a_n))+p_Y(g(a_n),f(a_n))-p_Y(g(a_n),g(a_n))\Big)\\
&\leq\sum_{n=1}^{N}\lambda^n\Big(p_Y(h(a_n),g(a_n))-p_Y(g(a_n),g(a_n))\Big)\\
&\qquad+\sum_{n>N}^{\infty}\lambda^n p_Y(h(a_n),g(a_n))+p^{\lambda}_{X\Rightarrow Y}(g,f)
\end{aligned}
$$

[1] Recall that $\mathcal{O}(X)$ is a continuous domain. For two open sets $U,V\in\mathcal{O}(X)$, $U\ll V$ holds when any open cover of V has a finite subset which covers U.

$$< \sum_{n=1}^{N} \lambda^n \frac{\theta}{2} + \sum_{n>N}^{\infty} \lambda^n + p^{\lambda}_{X \Rightarrow Y}(f,f) + \delta$$
$$\leq \frac{\theta}{2} + \frac{\theta}{2} + p^{\lambda}_{X \Rightarrow Y}(f,f) + \delta \leq p^{\lambda}_{X \Rightarrow Y}(f,f) + \epsilon.$$

$\mathcal{O}_{\sigma}(X \Rightarrow Y) \subseteq \mathcal{O}_{p_{X \Rightarrow Y}}(X \Rightarrow Y)$: It suffices to show that the basic Scott open sets $\uparrow\!\!\uparrow f$ contain an open p-ball $B_{\epsilon}(g)$ around any of its points $g \in \uparrow\!\!\uparrow f$. So, suppose $f \ll g$: by Lemma 32 there exists $c_1, \ldots, c_n \in X$, $b_1, \ldots, b_n \in Y$ such that $b_i \ll g(c_i)$, $\uparrow\!\!\uparrow c_i \ll g^{-1}(\uparrow\!\!\uparrow b_i)$ and $f \leq \bigvee_i (\uparrow\!\!\uparrow c_i \searrow b_i)$. From $b_i \ll g(c_i)$ it follows that there exists $\epsilon_i > 0$ such that $B_{\epsilon_i}(g(c_i)) \subseteq \uparrow\!\!\uparrow b_i$. Let N be such that for all $i = 1, \ldots, n$, c_i has an index $\leq N$ in the enumeration a_n of $\mathcal{B}(X)$. Let $\epsilon = \lambda^N \min\{\epsilon_1, \ldots, \epsilon_n\}$.
We claim that $B_{\epsilon}(g) \subseteq \uparrow\!\!\uparrow f$: let $h \in B_{\epsilon}(g)$, then, from $\sum_n \lambda^n \big(p_Y(h(a_n), g(a_n)) - p_Y(g(a_n), g(a_n))\big) < \epsilon$, we deduce, for $i = 1, \ldots, n$, $p_Y(h(c_i), g(c_i)) < p_Y(g(c_i), g(c_i)) + \lambda^{-i}\epsilon \leq p_Y(g(c_i), g(c_i)) + \epsilon_i$, that is, $h(c_i) \in B_{\epsilon_i}(g(c_i))$. We deduce that $b_i \ll h(c_i)$, and thus that $f \leq \bigvee_i (\uparrow\!\!\uparrow c_i \searrow b_i) \ll h$. We can thus conclude that $f \ll h$. ◀

We conclude this section with a few examples.

▶ **Example 33** (RealPCF)**.** The language RealPCF [22] is an extension of PCF with a type $\mathbf{I}$ for *partial real numbers* (i.e. finite approximations of real numbers or, equivalently, computable closed intervals) and primitives for computable analysis, with a canonical Scott semantics in which $\mathbf{I}$ is interpreted via the domain $\mathbf{I}(\mathbb{R})$. This is perfectly in line with the quantification of $\mathbf{I}(\mathbb{R})$ we presented in Example 5, which sees smaller and smaller intervals as providing more and more information. Via the applicative distances just presented, we obtain then a quantification of the Scott semantics of full RealPCF.

▶ **Example 34** (Scott topologies of open and closed sets)**.** Given a topological space X, one can endow the space $\mathcal{O}(X)$ of its open sets with the Scott topology induced by the inclusion order, as well as the (homeomorphic) space $\mathcal{C}(X)$ of its closed sets under the Scott topology induced by the reversed inclusion order.

Whenever X is exponentiable in Top (which is the case, in particular, whenever X is a Scott domain), the bijection $h : \mathcal{O}(X) \simeq \mathrm{Top}(X, S)$, where S is the Sierpinski space and $h(U)$ is the characteristic function of U, is a homeomorphism [24]. Given a countable basis $(x_n)_n$ of X, and weights θ_n with $\sum_n \theta_n \leq 1$, we can then quantify $\mathcal{O}(X)$ and $\mathcal{C}(X)$ via

$$p^{\mathcal{O}}_{x_n,\theta_n}(U,V) = \sum_{n=1}^{\infty} \theta_n \cdot s(h(U)(x_n), h(V)(x_n)) = \sum \{\theta_n \mid x_n \notin U \vee x_n \notin V\},$$
$$p^{\mathcal{C}}_{x_n,\theta_n}(C,D) = p^{\mathcal{O}}(\overline{C}, \overline{D}) = \sum \{\theta_n \mid x_n \in C \vee x_n \in D\}.$$

▶ **Example 35** (Böhm trees as closed sets)**.** Consider the poset $\mathcal{A}$ of partial terms. Let $\mathrm{Ide}(\mathcal{A})$ be the dcpo of *ideals* of $\mathcal{A}$, that is, of lower directed subsets of $\mathcal{A}$. Observe that any Böhm tree $\mathcal{B}(M) \subseteq \mathcal{A}$ is an element of $\mathrm{Ide}(\mathcal{A})$, and the set $\downarrow \mathcal{B}(M) = \{U \mid U \subseteq \mathcal{B}(M)\} \subseteq \mathrm{Ide}(\mathcal{A})$ is a closed set under the Scott topology of $\mathrm{Ide}(\mathcal{A})$. Given an enumeration A_n of partial terms and weights θ_n, we can then define an alternative λ-PPM by letting $p^{\mathcal{B}}_{A_n,\theta_n}(M,N) = p^{\mathcal{C}}_{\downarrow A_n,\theta_n}(\downarrow \mathcal{B}(M), \downarrow \mathcal{B}(N)) = \sum_n \{\theta_n \mid A_n \not\leq M \text{ or } A_n \not\leq N\}$. While they produce different distances, $p^{\mathcal{B}}_{A_n,\theta_n}$ and $p_{\text{Böhm}}$ quantify the same topology, i.e. $p_{\text{Böhm}} \sqsupseteq\sqsubseteq p^{\mathcal{B}}$.

▶ **Example 36** (Scott topology of the power set)**.** A countable set X is (trivially) a domain for the order given by equality, and its Scott topology coincides with the indiscrete topology, i.e. $\mathcal{O}(X) = \mathcal{P}(X)$. Given an enumeration x_n of X, the Scott topology on $\mathcal{P}(X)$ is thus quantified by $p^{\mathcal{P}}_{x_n,\theta_n}(A,B) = \sum \{\theta_n \mid x_n \notin A \vee x_n \notin B\}$, for $A, B \subseteq X$.

5 Quantifying a Reflexive Object

The denotational models for the untyped λ-calculus correspond to the *reflexive objects* within some cartesian closed category, that is, the objects X satisfying the isomorphism $X \simeq X \to X$. Within cpo-enriched categories, reflexive objects can be obtained by a direct limit construction, whose paradigmatic example is Scott's D_∞ model. In this section we show how to quantify this model via applicative distances, at the same time illustrating a technique that could be adapted to other similar constructions, like e.g. the reflexive object within the relational model [6].

Quantifying Scott's D_∞. Let us recall the idea of the direct limit construction of a reflexive object. One starts from some bounded complete domain D, and constructs a sequence of spaces $D_0 := D$, $D_{n+1} : D_n \to D_n$, together with maps $i_n : D_n \to D_{n+1}$ and $j_n : D_{n+1} \to D_n$ forming a pair (i_n, j_n) called an *injection/retraction pair*, that is, satisfying $j_n \circ i_n = \mathrm{id}_{D_{n+1}}$ and $i_n \circ j_n \leq \mathrm{id}_{D_n}$. One obtains then a reflexive object $D_\infty \simeq D_\infty \to D_\infty$ as the direct limit of the sequence $D_n \stackrel{i_n}{\to} D_{n+1}$, as well as injection-retraction pairs $i_{n\infty} : D_n \to D_\infty$, $j_{\infty n} : D_\infty \to D_n$.

Notice that an element x of D_∞ yields, for any n, a function $x_n := j_{\infty n}(x) \in D_n = D_{n-1} \to D_{n-2} \to \cdots \to D_0$; conversely, any compact element $x \in D_n$ yields a compact element $i_{n\infty}(x) \in D_\infty$, and such elements form indeed a basis of D_∞.

Suppose now that the starting space D is quantified by some PM p. Using the applicative metrics from the previous section we can quantify all the D_n by letting $p_0 := p$ and $p_{n+1}(x, y) = \sum_{i=1}^{\infty} \frac{1}{2^i} p_n(x(a_i^n), y(a_i^n))$, where $(a_i^n)_i$ is an enumeration of the basis elements of D_n. We obtain then a PM quantifying D_∞ by letting

$$\begin{aligned} p_\infty(x,y) &= \sum_{n=1}^{\infty} \frac{1}{2^n} p_n(x_n, y_n) \\ &= \sum_{n,k_{n-1},\ldots,k_0=1}^{\infty} \left(\frac{1}{2^{n+k_{n-1}+\cdots+k_0}} \right) \cdot p\left(x_n(a_{k_{n-1}}^{n-1}) \ldots (a_{k_0}^0), y_n(a_{k_{n-1}}^{n-1}) \ldots (a_{k_0}^0) \right). \end{aligned}$$

Intuitively, the distance p_∞ compares x and y by considering all possible ways of evaluating the functions $x_n, y_n \in D_n$ on n basis elements of the corresponding spaces $D_{n-1}, \ldots, D_0$. As for the applicative metrics from the previous section, while the distances $p_\infty(x, y)$ are defined via infinite series, one can check that $x \in B_\epsilon^{p_\infty}(y)$ by a finitary criterion.

▶ **Lemma 37.** *For all $x \in D_\infty$ and $n > 0$, there exists $N \in \mathcal{O}(n)$ such that for all $y \in D_\infty$, if, for all $i, k_0, \ldots, k_{i-1} \leq N$, $p(x_i a_{i-1}^{k_{i-1}} \ldots a_0^{k_0}, y_i a_{i-1}^{k_{i-1}} \ldots a_0^{k_0}) < 2^{-(n+1)}$, then $p_\infty(x, y) < 2^{-n}$.*

Proof. We must find N satisfying $\sum_{n,k_{n-1},\ldots,k_0 > N}^{\infty} \frac{1}{2^{n+k_{n-1}+\cdots+k_0}} < \frac{\epsilon}{2}$. Notice that if N satisfies $\sum_{n>N}^{\infty} \frac{1}{2^n} < \frac{\epsilon}{2}$, then it also satisfies the other condition, so we can argue as for Lemma 31. ◀

The following result is proved in detail in the long version.

▶ **Theorem 38.** *The partial metric p_∞ quantifies the Scott topology of D_∞.*

Proof. We will exploit a few properties of the maps i_{nm}, proved in the extended version:

- **i.** For all $n \in \mathbb{N}$, $x \in X_n$ and $y \in X_\infty$, $x \ll y_n \Rightarrow i_{n\infty}(x) \ll y$.
- **ii.** For all $x, y \in X_\infty$, $x \ll y$ iff there exists $N \in \mathbb{N}$ and $w_1, \ldots, w_k \in X_N$ such that $w_1, \ldots, w_k \ll y_N$ and $x \leq i_{N\infty}(w_1 \vee \cdots \vee w_k)$.
- **iii.** For all $n \in \mathbb{N}$, $x \in X_n$ and $y \in X_\infty$, $i_{n\infty}(x) \ll y \Rightarrow \exists N \forall k \geq N, i_{n(n+k)}(x) \ll y_{n+k}$.

$\mathcal{O}_\sigma(D_\infty) \supseteq \mathcal{O}_{p_\infty}(D_\infty)$: Let $y \in B_\epsilon(x)$. We need to find $y' \in D_\infty$ such that $y' \in B_\epsilon(x)$ and $y' \ll y$. From $p_\infty(y,x) < p_\infty(x,x) + \epsilon$ it follows that we can find $\theta, \delta > 0$ such that $p_\infty(y,x) < p_\infty(x,x) + \delta$ and $\delta + \theta \leq \epsilon$. Let N be such that $\sum_{n>N}^{\infty} \frac{1}{2^n} < \frac{\theta}{2}$. Since the p_n-balls are Scott open, for all $n \leq N$, we can find some $z_n \in B_{\frac{\theta}{2}}(y_n)$ such that $z_n \ll y_n$. Observe that by (i.) we have $i_{n\infty}(z_n) \ll y$. This implies in particular that the join $\bigvee_{n=1}^{N} i_{n\infty}(z_n)$ exists in D_∞. Define $y' := \bigvee_{n=1}^{N} i_{n\infty}(z_n)$. Notice that $y' \ll y$ holds so we just have to check that $y' \in B_\epsilon(x)$.

First recall that, by antimonotonicity of p_n, $p_n(a \vee a', b) \leq \min\{p_n(a,b), p_n(a',b)\}$. Now, for all $n \leq N$, we have that $y'_n = j_{\infty n}(y') = \left(\bigvee_{k<N} i_{k,n}(z_k)\right) \vee z_n \vee \left(\bigvee_{n<k\leq N} j_{k,n}(z_k)\right)$. Then we deduce $p_n(y'_n, y_n) \leq p_n(z_n, y_n) < p_n(y_n, y_n) + \frac{\theta}{2}$. We can now compute

$$
\begin{aligned}
p_\infty(y', x) &= \sum_{n=1}^{\infty} \frac{1}{2^n} p_n(y'_n, x_n) \\
&\leq \sum_{n=1}^{\infty} \frac{1}{2^n} \Big(p_n(y'_n, y_n) + p_n(y_n, x_n) - p(y_n, y_n) \Big) \\
&\leq \left(\sum_{n=1}^{\infty} \frac{1}{2^n} p_n(y'_n, y_n) - p_n(y_n, y_n) \right) + p_\infty(y, x) \\
&= \left(\sum_{n=1}^{N} \frac{1}{2^n} (p_n(y'_n, y_n) - p_n(y_n, y_n)) \right) \\
&\qquad + \left(\sum_{n>N}^{\infty} \frac{1}{2^n} (p_n(y'_n, y_n) - p_n(y_n, y_n)) \right) + p_\infty(y, x) \\
&< \left(\sum_{n=1}^{N} \frac{1}{2^n} \frac{\theta}{2} \right) + \frac{\theta}{2} + p_\infty(x,x) + \delta \leq p_\infty(x,x) + \theta + \delta \leq p_\infty(x,x) + \epsilon.
\end{aligned}
$$

$\mathcal{O}_\sigma(X_\infty) \subseteq \mathcal{O}_{p_\infty}(X_\infty)$: Suppose $x \ll y$. We need to find $\epsilon > 0$ such that $B_\epsilon(y) \subseteq \uparrow\!\!\!\uparrow x$. By (ii.) there exists N and $w_1, \ldots, w_k \in X_N$ such that $x \leq i_{N\infty}(w_1 \vee \cdots \vee w_k) \ll y$. By (iii.) there exists $N' \geq N$ such that $i_{NN'}(w_j) \ll y_{N'}$. Observe that $i_{N'\infty}(i_{NN'}(u)) = i_{N\infty}(u)$, which implies that $x \leq \bigvee_j i_{N'\infty}(i_{NN'}(w_j))$.

For each $j = 1, \ldots, k$ we can find then $\epsilon_j > 0$ such that $B_{\epsilon_j}(y_{N'}) \subseteq \uparrow\!\!\!\uparrow i_{NN'}(w_j)$. Let $\epsilon := 2^{-(N'+1)} \min\{\epsilon_j \mid j = 1, \ldots, k\}$. Suppose $z \in B_\epsilon(y)$: for all $j = 1, \ldots, k$, from $p_\infty(z,y) \leq \epsilon$ we deduce $p_{N'}(z_{N'}, y_{N'}) \leq 2^{N'}\epsilon < \epsilon_j$, whence $z_{N'} \in B_{\epsilon_j}(y_{N'})$, which forces $i_{NN'}(w_j) \ll z_{N'}$. By (i.) the last inequality implies $i_{N'\infty}(w_j) = i_{N'\infty}(i_{NN'}(w_j)) \ll z$, and we thus obtain $x \leq \bigvee_j i_{N'\infty}(i_{NN')}(w_j)) \ll z$, that is, $x \ll z$. ◀

The Scott λ-PPM. The interpretation of closed λ-terms in the Scott model D_∞, for D an arbitrary algebraic domain quantified by a PM p, yields a PPM $p_{\text{Scott}}(M,N) := p_\infty([\![M]\!],[\![N]\!])$, where $[\![M]\!] \in D_\infty$ indicates the interpretation of M inside D_∞. When D is non-trivial (i.e. $D \neq \{\bot\}$), using well-known properties of the Scott model, $p_{\text{Scott}}(M,N)$ yields an extensional and sensible λ-PPM.

The result below relates p_{Scott} to the other λ-PPMs discussed in Section 3.

▶ **Proposition 39.** $p_{\text{Böhm}} \sqsubset p_{\text{Scott}} \sqsubset p_{\text{ctx}}$.

Proof sketch.

($p_{\text{Böhm}} \sqsubset p_{\text{Scott}}$) We exploit the *approximation theorem* for D_∞ [5] which says that, for any closed λ-term M, letting $\Lambda^o_\perp$ be the set of closed partial terms and $[\![-]\!] : \Lambda^o_\perp \to D_\infty$ the interpretation function, $[\![M]\!] = \bigvee\{[\![A]\!] \mid A \leq M\}$. Since D is algebraic, any open ball $B^{p_0}_\epsilon([\![M]\!](\vec{a}))$ contains some compact element $c \ll [\![M]\!](\vec{a})$. By the approximation theorem, then, we deduce that there exists a partial term $A \leq M$ such that $c \ll [\![A]\!](\vec{a})$.

Consider the open ball $B^{p_{\text{Scott}}}_\epsilon(M)$. Thanks to Lemma 37 one can find a *finite* number of sequences of basis elements $\vec{a}_1, \ldots, \vec{a}_n$ and positive reals $\delta_1, \ldots, \delta_n > 0$ such that for all term P, if $[\![P]\!](\vec{a}_i) \in B^{p_0}_{\delta_i}([\![M]\!](\vec{a}_i)$ holds for all $i = 1, \ldots, n$, then $P \in B^{p_{\text{Scott}}}_\epsilon(M)$.

By reasoning as above via the approximation theorem, we obtain partial terms $A_1, \ldots, A_n \leq M$ such that $A_i \in B^{p_0}_{\delta_i}([\![M]\!](\vec{a}_i)$, and we deduce then $A = \bigvee_i A_i \in B^{p_{\text{Scott}}}_\epsilon(M)$. Letting now k be the height A and $\theta = 2^{-k}$, we thus conclude that $B^{\text{Böhm}}_\theta(M) \subseteq B^{p_{\text{Scott}}}_\epsilon(M)$.

The strictness follows from the fact that the associated λ-theories $\mathcal{B}$ and $\mathcal{H}^*$ are strictly included, as argued at the end of Remark 19 for the case of p_{ctx}.

($p_{\text{Scott}} \sqsubset p_{\text{ctx}}$) By Proposition 18, we only need to prove strictness. Let $I := \lambda x.x$ and consider the terms $P_k := \lambda y_1.\ldots.\lambda y_k.I$. It can be easily checked that, for any context $\mathtt{C}$, if $\mathtt{C}[I]$ is solvable, then $\mathtt{C}[P_k]$ must be solvable as well. This implies then that, for any $\epsilon > 0$, the open ball $B^{\text{ctx}}_\epsilon(I)$ contains *all* the terms P_k.

Now, one can construct a compact basis element $c \in D_\infty$ such that, for all $k > 2$, $[\![P_k]\!] \not\ll c \ll [\![I]\!]$ (see the extended version for the details). Since $\mathcal{O}_{p_\infty}(D_\infty)$ coincides with the Scott topology, which is generated by the sets $\uparrow b$, for b a basis element, from $c \ll [\![I]\!]$ we deduce that there exists $\epsilon > 0$ such that $B^{p_\infty}_\epsilon([\![I]\!]) \subseteq \uparrow c$. From $[\![P_2]\!] \not\ll c$ we deduce then $[\![P_2]\!] \notin B^{p_\infty}_\epsilon([\![I]\!])$, we conclude that the open p_∞-ball $B^{p_\infty}_\epsilon([\![I]\!])$ contains *no* open p_{ctx}-ball. ◀

Recalling that D_∞ induces the theory $\mathcal{H}^*$, the relation $p_{\text{Böhm}} \sqsubset p_{\text{Scott}}$ is in accordance with what happens with the corresponding λ-theories. By contrast, while D_∞ and the contextual preorder both induce the λ-theory $\mathcal{H}^*$, the first induces a λ-PPM which is *finer* than the contextual partial metric. As can be seen in the proof in the Appendix, the reason behind this is that, given terms $M \sqsubseteq_{\text{ctx}} P$, there exists open p_{Scott}-balls $B_\epsilon(P)$ whose elements all lie above M, while p_{ctx} cannot define any such ball, since whether $M \leq Q$ cannot be tested by applying only *finitely* many contexts to Q (cf. Remark 16).

6 Quantifying the Taylor Expansion

In this section we discuss the Taylor expansion of λ-terms [18, 19, 20], a well-studied method that refines methods based on Böhm trees and Scott domains, by decomposing the non-linear behavior of a term into the *linear* behavior of a set of simpler terms, called *resource λ-terms*. Notably, several well-known properties of λ-terms (like e.g. continuity and stability), which were originally established by topological and semantic methods, can be proved in a simpler, combinatorial way, via the Taylor expansion [4].

The famous *commutation theorem* [20] says that the Taylor expansion commutes with the construction of the Böhm tree, and shows that the associated λ-theories coincide. By presenting the Taylor expansion as an *isometric* transformation, we add a quantitative flavor to this result, showing that also the corresponding notions of program similarity coincide.

All proofs of the results contained in this section can be found in the extended version.

Resource terms and the Taylor expansion. As we said, the Taylor expansion associates a λ-term with a set of terms, called *resource terms*, with a linear operational semantics. The set Λ^r of resource terms is defined by the grammar $t := x \mid \lambda x.t \mid t\langle t, \dots, t\rangle$, where $\langle t, \dots, t\rangle$ indicates a finite multiset of terms. We define an order $\prec$ over resource λ-terms as the context closure of the relation $\emptyset \prec \langle t_1, \dots, t_n\rangle$. The operational semantics of resource terms replaces the standard β-rule with a linear monadic rule $\to_r$ that relates a redex $(\lambda x.t)\langle u_1, \dots, u_n\rangle$ with the set of terms $t[u_{\sigma(1)}/x_1, \dots, u_{\sigma(n)}/n]$, obtained by replacing each occurrence x_i of x in t by the term $u_{\sigma(i)}$, whenever t contains exactly n occurrences of x and where σ is any permutation in $\mathfrak{S}_n$. For example, the resource term $(\lambda x.x\langle x\rangle)\langle y, z\rangle$ reduces to the set of terms $\{y\langle z\rangle, z\langle y\rangle\}$ corresponding to the two possible ways of distributing y, z across the two occurrences of x in $x\langle x\rangle$. Instead, the resource term $(\lambda x.x\langle x\rangle)\langle y\rangle$ reduces to the empty set: as the single occurrence of y cannot be duplicated, it does not suffice to replace all occurrences of x in $x\langle x\rangle$. More generally, if t contains a number of occurrences of x different from n, then $(\lambda x.t)\langle u_1, \dots, u_n\rangle \to_r \emptyset$. Thanks to the impossibility of duplicating terms, linear reduction $\to_r^*$ is not only confluent, but also strongly normalizing (in linear time).

The *Taylor expansion* of a λ-term M is a set $\mathcal{T}(M) \subseteq \Lambda^r$ defined inductively as $\mathcal{T}(x) = \{x\}$, $\mathcal{T}(\lambda x.M) = \{\lambda x.t \mid t \in \mathcal{T}(M)\}$ and $\mathcal{T}(MN) = \{t\langle t_1, \dots, t_n\rangle \mid t \in \mathcal{T}(M), x_n \in \mathbb{N}, t_1, \dots, t_n \in \mathcal{T}(N)\}$. For example, the Taylor expansion of $\lambda x.\lambda y.yx$ is composed of all resource terms of the form $\lambda x.\lambda y.y\langle x, \dots, x\rangle$. Since reduction is confluent and strongly normalizing, we can define the set $\mathrm{nf}(\mathcal{T}(M))$ containing the normal forms of the resource terms in $\mathcal{T}(M)$.

The Taylor expansion extends to *partial* λ-terms by letting $\mathcal{T}(\bot) = \emptyset$. In this way, we can define the Taylor expansion of a Böhm tree $\alpha \in \mathrm{Ide}(\mathcal{A})$ by $\mathcal{T}(\alpha) = \bigcup\{\mathcal{T}(A) \mid A \in \alpha\}$,. The aforementioned commutation theorem says then that $\mathcal{T}(\mathcal{B}(M)) = \mathrm{nf}(\mathcal{T}(M))$; together with the injectivity of $\mathcal{T}$ over Böhm trees (which is easily proved), this shows the equivalence of the λ-theory $\mathcal{B}$ and the λ-theory generated by equating all closed terms whose Taylor expansions have the same normal form.

We provide an alternative, topological, presentation of the Taylor expansion of Böhm trees. A natural choice would be to take the Scott topology induced by the resource term order $\preceq$. However, under this order, Λ^r is not a dcpo: limits of directed sequences need not exist (as they would correspond, just like Böhm trees, to infinite terms). This leads then to consider, just like for partial terms, the completion $\mathrm{Ide}(\Lambda^r)$ of Λ^r, which forms an algebraic dcpo. The elements of $\mathrm{Ide}(\Lambda^r)$ can be seen as possibly infinite resource terms, and the compact elements correspond to the finite ones, that is, to ordinary resource terms.

Recall that $\mathrm{Ide}(\mathcal{A})$ can be identified with the set of Böhm trees; the Taylor expansion can be presented in this setting as a map $\mathcal{T}^* : \mathrm{Ide}(\mathcal{A}) \to \mathcal{P}(\mathrm{Ide}(\Lambda^r))$ defined by $\mathcal{T}^*(\alpha) = \mathrm{Ide}(\mathcal{T}(\alpha))$. To see that it is well-defined, let us observe that $\mathcal{T}(\alpha) \subseteq \Lambda^r$, so $\mathrm{Ide}(\mathcal{T}(\alpha)) \subseteq \mathrm{Ide}(\Lambda^r)$ is a set of ideals. Notice that the set $\mathcal{T}^*(\alpha)$ is *closed* with respect to the Scott topology of $\mathrm{Ide}(\Lambda^r)$.

Defining a metric on Λ^r. We introduce a PUM on Λ^r quantifying the order $\preceq$, which is essentially an adaptation of the tree partial metric. A normal resource term is of the form $t = \lambda x_1.\dots.\lambda x_n.xb_1 \dots b_m$, where each b_i is a finite multiset $b_i = \langle t_i^1, \dots, t_i^{m_i}\rangle$. The *height* of a resource term $h(t)$ is defined recursively as $h(t) = \max_{ij} h(t_i^j) + 1$, where t is as above. For any variable occurrence z in t, we define its *height in* t $h_t(z)$ as $h_t(z) = 1$ if z is as x above, and as $h_t(z) = h_{t_i^j}(z) + 1$ if the occurrence is in t_i^j.

For any normal resource term t and $n \leq h(t)$, we define the resource term $t|_n$, corresponding to the "truncation" of t at height n: if $h(t) \leq n$, then $t|_n = t$, and if $h(t) > n$, then we replace any subterm of t of the form $xb_1 \dots b_m$, where x is at height n, by $x\emptyset \dots \emptyset$. Observe that $h(t|_n) \leq n$ and $h(t|_n) = n$ holds whenever $h(t) \geq n$.

▶ **Definition 40** (resource partial metric). *For any two resource terms $t, u \in \Lambda^r$, we define*

$$r(t,u) := \inf\{2^{-n} \mid h(t), h(u) \geq n \text{ and } t|_n = u|_n\}.$$

By arguing similarly to the case of trees, it can be shown that r is a PUM, and that the order $\leq_r$ coincides with $\preceq$. Notice that $r(t,t) = 2^{-h(t)}$.

Lifting the metric to $\mathcal{P}(\Lambda^r)$. We now discuss how to lift the metric r to subsets of Λ^r. A standard way to lift a metric d from a set X to its powerset $\mathcal{P}(X)$ is via the *Hausdorff lifting* $H_d(A,B) = \max\{\sup_{a\in A}\inf_{b\in B} d(a,b), \sup_{b\in B}\inf_{a\in A} d(a,b)\}$. Intuitively, $H_d(A,B)$ looks, for each element of one set, for its *closest* element in the other set, and then measures the distance that is obtained by this operation in the worst case. The same construction, when applied to a partial metric p, yields the *partial Hausdorff metric* H_p (see [3, 28]) which, in spite of its name, is in fact *not* a partial metric, as it satisfies a *weaker* triangular law $H_p(A,B) \leq H_p(A,C) + H_p(C,B) - \inf_{c\in C} p(c,c)$.

In any case, the Hausdorff lifting H_r of the resource partial metric is not the right choice for us: suppose α is an infinite Böhm tree, so that its self-distance is 0; then $\mathcal{T}(\alpha)$ is a set of *finite* terms of arbitrary depth, so that $H_r(\mathcal{T}(\alpha),\mathcal{T}(\alpha)) = \sup_{t\in\mathcal{T}(\alpha)} r(t,t) = \sup\{2^{-|t|} \mid t \in \mathcal{T}(\alpha)\} = \frac{1}{2} > 0 = p_{\text{tree}}(\alpha,\alpha)$. Beyond making the Taylor expansion non-isometric, from this we deduce that H_r is constantly $\frac{1}{2}$ over *all* non-empty Taylor expansions!

Instead, we introduce the following variant of the Hausdorff lifting:

▶ **Definition 41.** *For any PM $p : X \times X \to [0,1]$, let $H_p^* : \mathcal{P}(X) \times \mathcal{P}(X) \to [0,1]$ be:*

$$H_p^*(A,B) = \max\left\{\sup_{a\in A}\inf_{a'\geq_p a\in A, b\in B} p(a',b), \sup_{b\in B}\inf_{b'\geq_p b\in B, a\in A} p(a,b')\right\}.$$

Intuitively, on two sets A, B, $H_p^*(A,B)$ measures how close the elements of A get to the elements of B as soon as one is allowed to freely move higher within A and B following the order $\leq_p$. Notice that, for α an infinite Böhm tree, we now have $H_r^*(\mathcal{T}(\alpha),\mathcal{T}(\alpha)) = 0$, as desired. Similarly to the partial Hausdorff metric H_p, for a partial metric p, H_p^* is *not* in general a partial metric. Indeed, it only satisfies the following properties:

▶ **Proposition 42.** *For any partial metric space (X,p), the distance H_p^* satisfies:*
1. $H_p^*(A,A) \leq H_p^*(A,B)$;
2. $H_p^*(A,B) = H_p^*(B,A)$;
3. $H_p^*(A,B) \leq H_p^*(A,C) + H_p^*(C,B) - \inf_{c\in C} p(c,c)$.

However, H_p^* is in fact a PM when restricted to $\mathrm{Ide}_p(X)$, the dcpo of ideals with respect to the order $\leq_p$.

▶ **Proposition 43.** *For any PM p on X, H_p^* is a PM on $\mathrm{Ide}_p(X)$ quantifying the order $\subseteq$.*

When $p = r$, the resource partial metric, H_r^* indeed quantifies the Scott topology:

▶ **Proposition 44.** *The PM H_r^* quantifies the Scott topology on $\mathrm{Ide}_r(\Lambda^r)$.*

Taylor is an isometry. The Taylor expansion can be presented either as a map $\mathcal{T} : \Lambda \to \mathcal{P}(\Lambda^r)$ turning a λ-term into a set of resource terms, or as a map $\mathcal{T}^* : \mathrm{Ide}(\mathcal{A}) \to \mathcal{P}(\mathrm{Ide}(\Lambda^r))$ turning a Böhm tree (i.e. an infinitary normal λ-term) into a set of infinitary resource terms.

We will show that both maps are isometries, when considering Λ with the Böhm PM and $\mathrm{Ide}(\mathcal{A})$ with the tree PM, and measuring sets of (finite/infinite) resource terms via the lifting H_r^* of the resource partial metric.

Let the λ-PPM p_{Taylor} be defined by $p_{\text{Taylor}}(M,N) = H_r^*(\text{nf}(\mathcal{T}(M)), \text{nf}(\mathcal{T}(N)))$. As we observed, the λ-theory generated by equating all terms M, N such that $\text{nf}(\mathcal{T}(M)) = \text{nf}(\mathcal{T}(N))$ coincides the theory $\mathcal{B}$. Our result will extend this to the corresponding quantitative theories.

Let us first consider the Taylor expansion of λ-terms.

▶ **Theorem 45.** *$\mathcal{T} : (\Lambda, p_{\text{Böhm}}) \longrightarrow (\mathcal{P}(\Lambda^r), H_r^*)$ is an isometry. Thus, $p_{\text{Taylor}} = p_{\text{Böhm}}$.*

The results above states that, whenever the Böhm trees of two terms M, N differ at height n, then, by moving higher and higher in their normalized Taylor expansions $\mathcal{T}(M)$ and $\mathcal{T}(N)$, one can find resource terms that differ precisely at height n, and can do no better.

Let us now consider the map $\mathcal{T}^*$. Since $\text{Ide}(\Lambda^r)$ is quantified by H_r^*, we can consider its lifting $H^*_{H_r^*}$ to $\mathcal{P}(\text{Ide}_p(\Lambda^r))$. In fact, the computation of $H^*_{H_r^*}$ leads us back to H_r^*:

▶ **Lemma 46.** *For all λ-terms M, N, $H_r^*(\mathcal{T}(M), \mathcal{T}(N)) = H^*_{H_r^*}(\mathcal{T}^*(M), \mathcal{T}^*(N))$.*

Thanks to Proposition 45, this immediately produces:

▶ **Theorem 47.** *$\mathcal{T}^* : (\text{Ide}(\mathcal{A}), p_{\text{tree}}) \longrightarrow (\mathcal{P}(\text{Ide}(\Lambda_r)), H^*_{H_r^*})$ is an isometry.*

▶ Remark 48. As shown in detail in the long version, we can obtain an isometry also if we choose to measure Böhm trees and Taylor expansions using the PMs from Examples 35 and 36. Indeed, for any enumeration $(A_n)_n$ of partial terms, one can define an enumeration $(r_n)_n$ of resource terms and weights θ_n such that $\mathcal{T} : (\mathcal{B}, p^{\mathcal{B}}_{(A_n)_n, \frac{1}{2^n}}) \longrightarrow (\mathcal{P}(\Lambda_r), p^{\mathcal{P}}_{(r_n)_n, \theta_n})$ is an isometry.

7 Conclusions

Related Work. Since their introduction in [8], the literature on partial metrics has grown vast, and comprises both theoretical investigations [39, 34, 3, 29] and connections with theoretical computer science [38], notably domain theory [9, 35, 40, 41]. Recently, an elegant categorical description of partial metric spaces as quantaloid-enriched categories has been proposed [28], as well as a characterization of the partial metric spaces that are *exponentiable* (in a category whose morphisms are the non-expansive - or 1-Lipschitz - functions and not, as in this paper, all continuous functions). While, as we have said, the metrizability of Scott domains via partial metrics has been well known since [9, 35], not much is found in this vast literature about the specific use of partial metrics for studying the topological semantics of the λ-calculus or, more generally, of higher-order programming languages.

Beyond partial metrics, the literature on higher-order program metrics has been growing vast as well. As the category Met of metric spaces and non-expansive functions is *not* cartesian closed, the literature has focused on two complementary directions: on the one hand, restrict to cartesian closed *sub*-categories of Met, like *ultra*-metric spaces [23], or *injective* metric spaces [10]; [15] adapts Mardare's et al.'s quantitative equational theories [32] to higher-order languages, introducing a notion of *quantitative λ-theory* (which, contrarily to λ-PPMs, require contexts to be non-expansive). On the other hand, restrict attention to *linear* [12, 16] or *graded* [37, 17] λ-calculi, which can be modeled in Met. Notably, [17] introduces *metric CPO*s, that is CPOs endowed with *sub*-continuous metrics (i.e. satisfying $d(\lim_n x_n, \lim_n y_n) \leq \epsilon$ whenever $d(x_n, y_n) \leq \epsilon$ holds for all n). This is a weaker condition than quantifiability, since the limits in the metric need not coincide with the CPO limits.

Differential logical relations [14, 13] have been recently introduced as a generalized approach to program metrics, relaxing usual Lipschitz, and even continuity, conditions. Notably, related models based on *generalized* partial metric spaces are studied in [27, 36]. In such models distances need not be positive reals but are computed on an arbitrary *quantale*.

Finally, several works have investigated infinitary λ-calculi defined via a *metric completion* of ordinary terms [30, 33]. These approaches are based on ultrametrics akin to the tree metric considered in this paper for Böhm trees. Recall that ordinary metric spaces are topologically Hausdorff, contrarily to the spaces considered in this paper. The metric completion of partial metric spaces is discussed in [26, 28].

Future Work. While this paper focuses on metric counterparts for well-known techniques, our results suggest several potential developments.

The metrizability of Scott domains suggests to study models based on Lipschitz-continuous, rather than just continuous, functions, as is standard in the literature on linear λ-calculi. For instance, considering the Böhm metric, a non-expansive context should respect depth: if two terms M, N coincide up to depth n, then $\mathtt{C}[M]$ and $\mathtt{C}[N]$ must also coincide up to depth n. This suggests connections with recent work on *stratified* notions of program equivalence [2].

Sections 4 and 6 introduced several methods to lift a partial metric to the powerset; using such liftings, as we suggest at several places, our results based on Scott domains could be adapted to the relational model, in which λ-terms are interpreted via relations $R \in \mathcal{P}(A \times B)$.

While we here just considered the untyped λ-calculus and basic cartesian closed structure (i.e. finite products and exponentials), the applicative distances introduced in this paper should adapt well also to languages with coproducts and dependent types; moreover, our results on the Hausdorff lifting suggests that other monadic liftings (e.g. the probability monad) could be considered. At the same time, the metric account of RealPCF suggested at in Example 33 could be explored in more depth, for instance considering the behavior of operators like the parallel if or even program derivatives.

Finally, the fact that several partial metrics considered in this paper produce computable distances between finite approximants suggests to explore potential connections with quantitative type systems related to the relational and topological semantics, like those based on non-idempotent intersection types [7].

References

1 Roberto M. Amadio and Pierre-Louis Curien. *Domains and Lambda-Calculi*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1998.

2 Victor Arrial, Giulio Guerrieri, and Delia Kesner. Genericity through stratification. In *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '24, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3661814.3662113.

3 Hassen Aydi, Mujahid Abbas, and Calogero Vetro. Partial Hausdorff metric and Nadler's fixed point theorem on partial metric spaces. *Topology and its Applications*, 159(14):3234–3242, 2012. doi:10.1016/j.topol.2012.06.012.

4 Davide Barbarossa and Giulio Manzonetto. Taylor subsumes Scott, Berry, Kahn and Plotkin. *Proc. ACM Program. Lang.*, 4(POPL), December 2019. doi:10.1145/3371069.

5 Hendrik Pieter Barendregt. *The lambda calculus - its syntax and semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1985.

6 Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. Not enough points is enough. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic*, pages 298–312, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-74915-8_24.

7 Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. Non-idempotent intersection types for the lambda-calculus. *Logic Journal of the IGPL*, 25(4):431–464, 2017. doi:10.1093/jigpal/jzx018.

8 Michael Bukatin, Ralph Kopperman, Steve Matthews, and Homeira Pajoohesh. Partial metric spaces. *American Mathematical Monthly*, 116:708–718, October 2009. URL: `http://www.jstor.org/stable/40391197`, `doi:10.4169/193009709X460831`.

9 Michael A. Bukatin and Joshua S. Scott. Towards computing distances between programs via Scott domains. In Sergei Adian and Anil Nerode, editors, *Logical Foundations of Computer Science*, pages 33–43, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg. `doi:10.1007/3-540-63045-7_4`.

10 Maria Manuel Clementino and Dirk Hofmann. Exponentiation in V-categories. *Topology and its Applications*, 153(16):3113–3128, 2006. Special Issue: Aspects of Contemporary Topology. `doi:10.1016/j.topol.2005.01.038`.

11 Raphaëlle Crubillé and Ugo Dal Lago. Metric Reasoning About Lambda-Terms: The General Case. In Hongseok Yang, editor, *Programming Languages and Systems - 26th European Symposium on Programming, ESOP 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, volume 10201 of *Lecture Notes in Computer Science*, pages 341–367, Berlin, Heidelberg, 2017. Springer. `doi:10.1007/978-3-662-54434-1_13`.

12 Fredrik Dahlqvist and Renato Neves. An Internal Language for Categories Enriched over Generalised Metric Spaces. In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic (CSL 2022)*, volume 216 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:18, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.CSL.2022.16`.

13 Ugo Dal Lago and Francesco Gavazzo. Differential logical relations part II: increments and derivatives. In Gennaro Cordasco, Luisa Gargano, and Adele A. Rescigno, editors, *Proceedings of the 21st Italian Conference on Theoretical Computer Science, Ischia, Italy, September 14-16, 2020*, volume 2756 of *CEUR Workshop Proceedings*, pages 101–114. CEUR-WS.org, 2020. URL: `http://ceur-ws.org/Vol-2756/paper_10.pdf`.

14 Ugo Dal Lago, Francesco Gavazzo, and Akira Yoshimizu. Differential logical relations, part I: the simply-typed case. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 111:1–111:14, 2019. `doi:10.4230/LIPIcs.ICALP.2019.111`.

15 Ugo Dal Lago, Furio Honsell, Marina Lenisa, and Paolo Pistone. On Quantitative Algebraic Higher-Order Theories. In Amy P. Felty, editor, *7th International Conference on Formal Structures for Computation and Deduction (FSCD 2022)*, volume 228 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:18, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.FSCD.2022.4`.

16 Ugo Dal Lago, Naohiko Hoshino, and Paolo Pistone. On the Lattice of Program Metrics. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*, volume 260 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.FSCD.2023.20`.

17 Arthur Azevedo de Amorim, Marco Gaboardi, Justin Hsu, Shin-ya Katsumata, and Ikram Cherigui. A semantic account of metric preservation. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, POPL 2017, pages 545–556, New York, NY, USA, 2017. ACM. `doi:10.1145/3009837.3009890`.

18 Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309:1–41, 2003. `doi:10.1016/S0304-3975(03)00392-X`.

19 Thomas Ehrhard and Laurent Regnier. Böhm Trees, Krivine's Machine and the Taylor Expansion of Lambda-Terms. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *Logical Approaches to Computational Barriers*, pages 186–197, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. `doi:10.1007/11780342_20`.

20 Thomas Ehrhard and Laurent Regnier. Uniformity and the Taylor expansion of ordinary lambda-terms. *Theor. Comput. Sci.*, 403(2-3):347–372, 2008. doi:10.1016/j.tcs.2008.06.001.

21 Thomas Erker, Martín Hötzel Escardó, and Klaus Keimel. The way-below relation of function spaces over semantic domains. *Topology and its Applications*, 89(1):61–74, 1998. Domain Theory. doi:10.1016/S0166-8641(97)00226-5.

22 Martín Hötzel Escardó. PCF extended with real numbers. *Theor. Comput. Sci.*, 162(1):79–115, 1996. doi:10.1016/0304-3975(95)00250-2.

23 Martín Hötzen Escardó. A metric model of PCF. Unpublished note presented at the Workshop on Realizability Semantics and Applications, June 1999. Available at the author's webpage., 1999.

24 M.H. Escardo and R. Heckmann. Topologies on spaces of continuous functions. *Topology Proceedings*, 26(2):545–564, 2001-2002.

25 Francesco Gavazzo. Quantitative behavioural reasoning for higher-order effectful programs: Applicative distances. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 452–461, New York, NY, USA, 2018. doi:10.1145/3209108.3209149.

26 Xun Ge and Shou Lin. Completions of partial metric spaces. *Topology and its Applications*, 182:16–23, 2015. doi:10.1016/j.topol.2014.12.013.

27 Guillaume Geoffroy and Paolo Pistone. A partial metric semantics of higher-order types and approximate program transformations. In *Computer Science Logic 2021 (CSL 2021)*, volume 183 of *LIPIcs–Leibniz International Proceedings in Informatics*, pages 35:1–35:18, 2021. doi:10.4230/LIPIcs.CSL.2021.23.

28 Dirk Hofmann and Isar Stubbe. Topology from enrichment: the curious case of partial metrics. *Cahiers de Topologie et Géométrie DIfférentielle Catégorique, LIX*, 4:307–353, 2018.

29 Gunther Jäger and T. M. G. Ahsanullah. Characterization of quantale-valued metric spaces and quantale-valued partial metric spaces by convergence. *Applied General Topology*, 19(1):129–144, 2018.

30 Richard Kennaway, Jan Willem Klop, M. Ronan Sleep, and Fer-Jan de Vries. Infinitary lambda calculus. *Theor. Comput. Sci.*, 175(1):93–125, 1997. doi:10.1016/S0304-3975(96)00171-5.

31 Giulio Manzonetto. *Models and theories of lambda calculus*. PhD thesis, Paris Diderot University, France, 2008. URL: https://tel.archives-ouvertes.fr/tel-00715207.

32 Radu Mardare, Prakash Panangaden, and Gordon Plotkin. Quantitative algebraic reasoning. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2016)*. IEEE Computer Society, 2016.

33 Damiano Mazza. Non-linearity as the metric completion of linearity. In Masahito Hasegawa, editor, *Typed Lambda Calculi and Applications*, pages 3–14, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. doi:10.1007/978-3-642-38946-7_3.

34 Volodymyr Mykhaylyuk and Vadym Myronyk. Metrizability of partial metric spaces. *Topology and its Applications*, 308:107949, 2022. doi:10.1016/j.topol.2021.107949.

35 S.J. O'Neill. Partial metrics, valuations and domain theory. *Annals of the New York Academy of Sciences*, 806:304–315, 1996.

36 Paolo Pistone. On generalized metric spaces for the simply typed lambda-calculus. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–14. IEEE, 2021. doi:10.1109/LICS52264.2021.9470696.

37 Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pages 157–168. ACM, 2010. doi:10.1145/1863543.1863568.

38 Salvador Romaguera, Pedro Tirado, and Óscar Valero. Complete partial metric spaces have partially metrizable computational models. *Int. J. Comput. Math.*, 89(3):284–290, 2012. doi:10.1080/00207160.2011.559229.

39 M. P. Schellekens. The correspondence between partial metrics and semivaluations. *Theoretical Computer Science*, 315(1):135–149, May 2004. doi:10.1016/j.tcs.2003.11.016.

40 Michel P. Schellekens. A characterization of partial metrizability: domains are quantifiable. *Theor. Comput. Sci.*, 305(1-3):409–432, 2003. Topology in Computer Science. doi:10.1016/S0304-3975(02)00705-3.

41 Michael B. Smyth. The constructive maximal point space and partial metrizability. *Ann. Pure Appl. Log.*, 137(1-3):360–379, 2006. doi:10.1016/j.apal.2005.05.032.

42 Pawel Waszkiewicz. Distance and measurement in domain theory. *Electronic Notes in Theoretical Computer Science*, 45, 2001. doi:10.1016/S1571-0661(04)80975-7.

A Kleene Algebra with Tests for Union Bound Reasoning About Probabilistic Programs

Leandro Gomes
Université de Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000, France

Patrick Baillot
Université de Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000, France

Marco Gaboardi
Boston University, MA, USA

Abstract

Kleene Algebra with Tests (KAT) provides a framework for algebraic equational reasoning about imperative programs. The recent variant Guarded KAT (GKAT) allows to reason on non-probabilistic properties of probabilistic programs. Here we introduce an extension of this framework called approximate GKAT (aGKAT), which equips GKAT with a partially ordered monoid (real numbers) enabling to express satisfaction of (deterministic) properties *except* with a probability up to a certain bound. This allows to represent in equational reasoning "à la KAT" proofs of probabilistic programs based on the union bound, a technique from basic probability theory. We show how a propositional variant of approximate Hoare Logic (aHL), a program logic for union bound, can be soundly encoded in our system aGKAT. We then illustrate the use of aGKAT with an example of accuracy analysis from the field of differential privacy.

2012 ACM Subject Classification Theory of computation → Algebraic semantics; Theory of computation → Pre- and post-conditions; Theory of computation → Logic and verification; Theory of computation → Hoare logic

Keywords and phrases Kleene algebras with tests, Hoare logic, equational reasoning, probabilistic programs, union bound, formal verification

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.35

Related Version *Extended Version*: `https://hal.science/hal-04196675v3` [13]

Funding *Leandro Gomes*: This work is financed by National Funds through FCT - Fundação para a Ciência e a Tecnologia, I.P. (Portuguese Foundation for Science and Technology) within the project IBEX, with reference 10.54499/PTDC/CCI-COM/4280/2021.
Patrick Baillot: Work partially supported by ANR Project HOPR (ANR-24-CE48-5521-01).

1 Introduction

Kleene algebra with tests (KAT) has been introduced in [21] as an algebraic framework for program verification. A KAT is a two-sorted structure, consisting of a Kleene algebra and a Boolean algebra of tests: the Kleene algebra part accounts for programs, with sequential composition, branching and iteration; the Boolean algebra part accounts for the predicates used to build if-then-else instructions, while loops and assertions, as well as, being KAT able to subsume propositional Hoare logic [22], for the pre and post-conditions. This framework allowed to give algebraic proofs corresponding to several approaches in program verification, see e.g. [22, 1, 24], and has been implemented as a library for the Coq proof assistant [28]. It has also been followed by several variants, like NetKAT [2], which allows to reason about software defined networks, Concurrent NetKAT [31] for concurrent networks, CKAO [18] for concurrent programs and more recently TopKAT for reasoning about incorrectness [32].

Recently the variant Guarded KAT (GKAT) [30] has been proposed as a restriction of KAT where all sums and iterations are guarded by tests. It offers several advantages over KAT, including the fact that the complexity of its equational theory is lower (almost linear time, provided that the number of tests is fixed) and the existence of a probabilistic model. The latter paves the way for using GKAT for reasoning about probabilistic programs. However an important feature of this system is that the tests of GKAT remain the same as those of KAT, namely they express Boolean properties on states. Therefore the framework of GKAT allows to encode probabilistic programs, but the assertions about them are non-probabilistic.

In this paper our goal is to extend the GKAT approach to reason about probabilistic programs satisfying properties *with a given probability bound* β. The objective is not to design an expressive framework for advanced probabilistic proofs, but instead to allow for simple probabilistic reasoning with a low technical overhead.

Concretely we target proofs based on the *union bound principle*, a property from basic probability theory, which is a simple consequence of the definition of probability measure, and can be stated as follows: given some properties $A_1, \ldots, A_n$, one has $Pr[\cup_{i=1}^{n} A_i] \leq \sum_{i=1}^{n} Pr[A_i]$. This principle is ubiquitous when reasoning about properties of randomized algorithms [27] and in their application in security, privacy [9], learning theory [19], etc.

A previous approach for reasoning about probabilistic imperative programs using the union bound principle had been provided by the *union bound program logic* aHL [5]. This is a Hoare logic for reasoning about probabilistic programs with non-probabilistic assertions but with judgements carrying a numeric index for tracking the failure probability. That is, judgments have the form $\vdash_\beta c : \phi \Rightarrow \psi$ where β is an upper bound on the probability that $\neg\psi$ is true after executing c starting from a memory satisfying ϕ. The authors illustrated how this logic could be used for the verification of accuracy of some algorithms, in particular in the setting of differential privacy. A relational variant of this logic is handled by the Easycrypt tool [8, 4] , which can be used for proving properties of cryptographic protocols as well as differential privacy properties of programs.

A natural idea is thus to adapt the union bound logic aHL to the GKAT framework. To do this and capture the union bound reasoning in an algebraic framework we extend GKAT with an additional relation, denoted $\lhd$, relating GKAT expressions with elements of a partially ordered monoid, typically real numbers on $[0, 1]$. We call this new system *approximate* GKAT (aGKAT). An important feature of this structure is that we want the new setting to subsume standard GKAT, requiring aGKAT to satisfy the theory of GKAT. A second feature is that we want the probabilistic model of sub-Markov kernels to be a model of our new structure, when we consider the monoid of real numbers. For this particular instantiation, the meaning of $\lhd$ will be that $c \lhd \beta$ holds if *the probability of successful execution of program* c *is bounded by* β. The theory of aGKAT extends the one of GKAT, by a small set of axioms characterizing the properties of the new relation $\lhd$. We illustrate how this theory allows for a concise form of equational reasoning for establishing probability bounds on some GKAT programs. Moreover, in order to demonstrate the expressivity of aGKAT, we encode aHL in it. This is inspired by the classical result of Kozen [22] showing that propositional Hoare logic can be encoded in KAT.

Outline. In Sect. 2 we will recall GKAT and its probabilistic model, and in Sect. 3 we will recall the Hoare logic aHL. Then in Sect. 4 we will define our system, aGKAT, its theory and its semantics. After that in Sect. 5 we will provide an encoding of the logic aHL in aGKAT and prove its soundness. Sect. 6 will be devoted to an example, the analysis in aGKAT of the accuracy of the probabilistic algorithm Report-noisy-max. Finally, Sec. 7 overviews related work and Sec. 8 enumerates possible directions for future work.

2 Guarded Kleene algebra with tests

This section recalls the language and the semantics of *Guarded Kleene Algebra with Tests* (GKAT) [30], an abstraction of imperative programs where conditionals and loops are encoded as guarded sums $(c_1 +_b c_2)$ and guarded iterations $(c^{(b)})$, respectively, guarded by Boolean predicates b. The structure is a restriction of KAT in which we are not allowed to freely use operators $+$ and * to build terms. In other words, GKAT does not allow nondeterminism.

2.1 Syntax

The syntax of GKAT is defined with a set of *actions* Σ and a finite set of primitive tests $\mathtt{T}$, which are disjoint. We denote actions by a and primitive tests by p. The sets of Boolean expressions $\mathtt{BExp}$ (also called tests) and GKAT expressions $\mathtt{Exp}$ (also called programs) are then defined by the following grammars:

$b, b_1, b_2 \in \mathtt{BExp} ::=$	
0	**false**
1	**true**
$p \in \mathtt{T}$	p
$b_1 \cdot b_2$	b_1 **and** b_2
$b_1 + b_2$	b_1 **or** b_2
$\bar{b}$	**not** b

$c, c_1, c_2 \in \mathtt{Exp} ::=$	
$a \in \Sigma$	**do** a
$b \in \mathtt{BExp}$	**assert** b
$c_1 \cdot c_2$	$c_1; c_2$
$c_1 +_b c_2$	**if** b **then** c_1 **else** c_2
$c^{(b)}$	**while** b **do** c

where, for any $b, b_1, b_2 \in \mathtt{BExp}$, operators $\cdot$, $+$ and $^-$ denote conjunction, disjunction and negation, respectively, and, for any $c, c_1, c_2 \in \mathtt{Exp}$, the operator $\cdot$ denotes sequential composition. The notations on the r.h.s. are given to help intuition and will sometimes be used when writing programs. We introduce command **skip** as a shorthand for **assert** 1, which is encoded by the Boolean expression 1.
The precedence of the operators is the usual one, i.e. the operator $\cdot$ has higher precedence than operator $+_b$, and $()^{(b)}$ has higher precedence that $\cdot$.[1] To simplify the writing, we often omit the operator $\cdot$ by writing c_1c_2 for the expression $c_1 \cdot c_2$, for any $c_1, c_2 \in \mathtt{Exp}$.

We are interested in using GKAT for representing probabilistic programs. For that, let us first fix a few definitions. Given a set S, $\mathcal{D}(S)$ is the set of *probability sub-distributions*[2] over S with countable support, i.e. the set of functions $f : S \to [0, 1]$ such that $Supp(f) = \{x \in S \mid f(x) > 0\}$ is countable and f sums up to at most 1, i.e. $\sum_{s \in S} f(s) \leq 1$. In particular, the *Dirac* distribution $\delta_s \in \mathcal{D}(S)$ is the map $w \to [w = s] = \begin{cases} 1, \text{ if } w = s \\ 0, \text{ otherwise} \end{cases}$

▶ **Example 1** (Imperative programming language)**.** Take a set $\mathtt{Var}$ of variables and a set $\mathtt{Distr}$ of sub-distributions over $\mathbb{R}$ with discrete support. Consider a simple imperative programming language defined by the following grammar:

$$\textit{terms } t \in \mathtt{Terms} ::= x \in \mathtt{Var} \mid r \in \mathbb{R} \mid t_1 + t_2 \mid t_1 - t_2 \mid t_1 \times t_2$$
$$\textit{distributions } d \in \mathtt{Distr}$$
$$\textit{tests } b \in \mathtt{Tests} ::= \textbf{false} \mid \textbf{true} \mid t_1 < t_2 \mid t_1 = t_2 \mid \textbf{not } b \mid b_1 \textbf{ and } b_2 \mid b_1 \textbf{ or } b_2$$

[1] For example the GKAT expression $c_1^{(b_1)} \cdot c_2 +_{b_2} c_3$ reads as $((c_1^{(b_1)}) \cdot c_2) +_{b_2} c_3$.
[2] Some examples of distributions are the tossing of a fair coin, with probability 0.5 for 0 and 1, and the (discrete version of the) Laplacian distribution $\mathcal{L}_p(a)$ centered in a with parameter p. The density function of $\mathcal{L}_p(a)$ is given by $\frac{1}{2p} \exp(\frac{|x-a|}{p})$.

$$\textit{commands } c \in \texttt{Comm} ::= \textbf{skip} \mid x \leftarrow t \mid x \xleftarrow{\$} d \mid c_1; c_2 \mid \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2 \mid \textbf{while } b \textbf{ do } c$$

This language can be modeled in GKAT by taking as sets of actions and primitive tests respectively $\Sigma = \{x \leftarrow t,\ x \xleftarrow{\$} d \mid x \in \texttt{Var}, t \in \texttt{Terms}, d \in \texttt{Distr}\}$ and $\texttt{T} = \{t_1 < t_2,\ t_1 = t_2 \mid t_1, t_2 \in \texttt{Terms}\}$ [3] . The first action evaluates term t and assigns the result to x and the second one samples from d and assigns the result to x. Observe that while programs c may be probabilistic, due to the use of samplings, the tests b as for them are deterministic, i.e. they do not use any probabilistic primitives. In particular the conditional branching in programs is only done on deterministic tests.

2.2 Semantics

We now present the semantic interpretation of GKAT that we will be using, the *Probabilistic model* [30] [4]. We first review some basic concepts needed for the semantics. The *Iverson bracket* $[b]$, for $b \in \texttt{BExp}$, is the function taking value 1 if b is true and 0 otherwise. Typical models of probabilistic imperative programming languages interpret programs as *Markov kernels* on a set S, i.e. maps from S to probability distributions. The semantic model defined below interprets programs as *sub-Markov* kernels, i.e. Markov kernels on sub-distributions.

▶ **Definition 2** (Probabilistic interpretation). *Let $i = (\texttt{State}, eval, sat)$ be a triple where:*

- *$\texttt{State}$ is a set of states,*
- *for each action $a \in \Sigma$, $eval(a) : \texttt{State} \to \mathcal{D}(\texttt{State})$ is a sub-Markov kernel,*
- *for each primitive test $p \in \texttt{T}$, $sat(p) \subseteq \texttt{State}$ is a set of states.*

The *probabilistic interpretation* of an expression c with respect to i is the sub-Markov kernel $\mathcal{P}_i[\![c]\!] : \texttt{State} \to \mathcal{D}(\texttt{State})$ defined as follows:

1. $\mathcal{P}_i[\![a]\!] := eval(a)$
2. $\mathcal{P}_i[\![b]\!](\sigma) := [\sigma \in sat^\dagger(b)] \times \delta_\sigma$
3. $\mathcal{P}_i[\![c_1 \cdot c_2]\!](\sigma)(\sigma') := \sum_{\sigma''} \mathcal{P}_i[\![c_1]\!](\sigma)(\sigma'') \times \mathcal{P}_i[\![c_2]\!](\sigma'')(\sigma')$
4. $\mathcal{P}_i[\![c_1 +_b c_2]\!](\sigma) := [\sigma \in sat^\dagger(b)] \times \mathcal{P}_i[\![c_1]\!](\sigma) + [\sigma \in sat^\dagger(\bar{b})] \times \mathcal{P}_i[\![c_2]\!](\sigma)$
5. $\mathcal{P}_i[\![c^{(b)}]\!](\sigma)(\sigma') := \lim_{n\to\infty} \mathcal{P}_i[\![(c +_b 1)^n \cdot \bar{b}]\!](\sigma)(\sigma')$

where $\text{sat}^\dagger : \texttt{BExp} \to 2^{\texttt{State}}$ is the lifting of $\text{sat} : \texttt{T} \to 2^{\texttt{State}}$ to arbitrary Boolean expressions over $\texttt{BExp}$, and $\times$ denotes both multiplication on real numbers and the pointwise multiplication on sub-distributions. For instance the definition of $\mathcal{P}_i[\![b]\!](\sigma)$ means that it is either δ_σ if σ belongs to $sat^\dagger(b)$, or the constant sub-distribution equal to 0 otherwise. Intuitively $\mathcal{P}_i[\![c]\!](\sigma)(\sigma')$ is the probability that the execution of c on initial state σ terminates on state σ', and $\sum_{\sigma'} \mathcal{P}_i[\![c]\!](\sigma)(\sigma')$ is the probability that the execution of c on initial state σ terminates on a state (we then also say that it is a successful execution). Observe thus that we really need to consider sub-distributions and not only distributions. Let us recall that the existence of the limit in point 5. has been shown in [30].

▶ Remark 3 (Finite state case). In the case where $\texttt{State}$ is a finite set of size n, say $\{s_1, \ldots, s_n\}$ then a sub-Markov kernel f can be represented as an $n \times n$ matrix $\mathcal{M} = (a_{i,j})_{i,j\in[1,n]}$. Each coefficient $a_{i,j}$ is defined as $a_{i,j} = f(s_i)(s_j)$. So in particular the sum over each line is inferior or equal to 1. We denote $f = \mathcal{M}$. For tests b, the matrix $\mathcal{P}_i[\![b]\!]$ has only diagonal coefficients,

[3] Note that technically speaking according to the definition of GKAT the set $\texttt{T}$ should be chosen finite, which is not the case here, but as observed in [30] Sect. 2.3 Example 2.5 we can use a finite subset $\texttt{T}'$ of $\texttt{T}$ for reasoning on pairwise equivalence of programs which terminate.

[4] Note that more interpretations of GKAT are presented in [30], namely a relational model and a language model.

with value $a_{i,i} = 1$ if $s_i \in \mathrm{sat}^\dagger(b)$, $a_{i,i} = 0$ if s_i not in $\mathrm{sat}^\dagger(b)$. In the case of $c_1 \cdot c_2$, the matrix $\mathcal{P}_i[\![c_1 \cdot c_2]\!]$ is obtained by the matrix product of $\mathcal{P}_i[\![c_1]\!]$ and $\mathcal{P}_i[\![c_2]\!]$. See [13] for an example.

In the following we will consider programs over a finite set of variables $\mathtt{Var}$ and the set of states will be the set of *memories*, that is to say functions in $\mathtt{Var} \to D$ where D is the domain of variables (we can take for instance $D = \mathbb{Q}$, the rational numbers). If $x \in \mathtt{Var}$ and σ is a memory, then $\sigma[x \leftarrow t]$ is the memory identical to σ except that it maps x to the evaluation of t in memory σ.
The interpretation of actions $a \in \Sigma$ as sub-Markov Kernels is then given by $eval(x \leftarrow t)(\sigma) := \delta_{\sigma[x \leftarrow t]}$ and $eval(x \overset{\$}{\leftarrow} d)(\sigma) := \sum_{t \in Supp(d)} d(t) \cdot \delta_{\sigma[x \leftarrow t]}$.
In the sequel memories will often be denoted as m.

2.3 Axioms

The theory of GKAT introduced in [30] is given by the axioms from Fig. 1. Note in particular the fixpoint axiom (13). Intuitively, it says that if expression c_3 chooses (using guard b) between executing c_1 and looping again, and executing c_2, then c_3 is a b-guarded loop followed by c_2. However, the rule is not sound in general. In order to overcome this limitation,

$$c +_b c = c \quad (1)$$
$$c_1 +_b c_2 = c_2 +_{\bar{b}} c_1 \quad (2)$$
$$(c_1 +_{b_1} c_2) +_{b_2} c_3 = c_1 +_{b_1 \cdot b_2} (c_2 +_{b_2} c_3) \quad (3)$$
$$c_1 +_b c_2 = b \cdot c_1 +_b c_2 \quad (4)$$
$$c_1 \cdot c_3 +_b c_2 \cdot c_3 = (c_1 +_b c_2) \cdot c_3 \quad (5)$$
$$(c_1 \cdot c_2) \cdot c_3 = c_1 \cdot (c_2 \cdot c_3) \quad (6)$$
$$0 \cdot c = 0 \quad (7)$$
$$c \cdot 0 = 0 \quad (8)$$
$$1 \cdot c = c \quad (9)$$
$$c \cdot 1 = c \quad (10)$$
$$c^{(b)} = c \cdot c^{(b)} +_b 1 \quad (11)$$
$$(c +_{b_2} 1)^{(b_1)} = (b_2 \cdot c)^{(b_1)} \quad (12)$$
$$\frac{c_3 = c_1 \cdot c_3 +_b c_2}{c_3 = c_1^{(b)} \cdot c_2} \text{ if } E(c_1) = 0 \quad (13)$$

Figure 1 Axiomatisation of Guarded Kleene algebra with tests.

following [30] (Section 3.1, Definition 3.2), the side condition $E(c_1) = 0$ is introduced, ensuring that command c_1 is *productive*, i.e. that it performs some action. To this end, the function E is inductively defined as follows: $E(b) := b$, $E(a) := 0$, $E(c_1 +_b c_2) := b \cdot E(c_1) + \bar{b} \cdot E(c_2)$, $E(c_1 \cdot c_2) := E(c_1) \cdot E(c_2)$, $E(c^{(b)}) := \bar{b}$.
We can see $E(c)$ as the weakest test that guarantees that command c terminates successfully but does not perform any action.

Moreover, note particularly the following observation: in KAT the encoding $c_1(bc_2 + \bar{b}c_3) = c_1bc_2 + c_1\bar{b}c_3$ is not an **if-then-else** statement; it is rather a nondeterministic choice between executing c_1, then testing b and executing c_2, and executing c_1, then testing $\bar{b}$ and executing c_3. The corresponding encoding in GKAT would be $c_1(c_2 +_b c_3) = c_1c_2 +_b c_1c_3$, an equality which is actually *not* valid in GKAT. Since GKAT is restricted to deterministic programs, there is no valid correspondence between the KAT encoding, which is not an **if-then-else** statement, and the hypothetical correspondent GKAT encoding, which is not valid. That is why left distributivity does not hold in GKAT for any $c \in \mathtt{Exp}$; it only holds for the particular case of $c_1 \in \mathtt{BExp}$, i.e. if c_1 is a test.

We define the relation $\leq$ on tests as: $b_1 \leq b_2$ iff $b_1 + b_2 = b_2$. Contrarily to KAT [21], the relation $\leq$ is not defined on an arbitrary GKAT expression, only on tests. In [13] we recall additional derivable equations in GKAT from [30].

Since any test is a program ($\mathtt{BExp} \subseteq \mathtt{Exp}$), the grammar also allows to write expressions as $b_1 +_b b_2$, for any $b \in \mathtt{BExp}$. We thus establish the following proposition[5] (proof in [13]) which expresses the guarded sum $+_b$, for any $b \in \mathtt{BExp}$, in terms of the disjunction $+$ on tests.

▶ **Proposition 4.** *For any tests b, b_1, b_2 one has: $b_1 +_b b_2 = bb_1 + \bar{b}b_2$.*

By Boolean reasoning, we can observe that $bb + \bar{b}\bar{b} = 1$. This observation will be useful later to prove the soundness of some aHL rules in aGKAT.

We also state the following proposition (see [13] for the proof):

▶ **Proposition 5.** *For any tests b_1, b_2 one has: $b_1 + b_2 = b_1 +_{b_1} b_2$.*

3 Union bound logic - Approximate Hoare logic

In this section we recall *Approximate Hoare logic (aHL)* [5], a logic based on the union bound, a tool from probability theory for analyzing randomised algorithms. A judgment in aHL is of the form $\vdash_\beta c : \phi \Rightarrow \psi$ where: ϕ, ψ are first-order formulas representing non probabilistic pre- and post-conditions[6], respectively; β is a value in $[0, 1]$ and it is an upper bound on the probability that the post-condition ψ does *not* hold on the output distribution, assuming that ϕ holds on an initial memory m. We assume a probabilistic interpretation i and we will denote $m \models \phi$ if ϕ is valid in memory m. The validity of the judgement is thus stated by:

▶ **Definition 6** (Validity of aHL judgment)**.** *A judgment $\vdash_\beta c : \phi \Rightarrow \psi$ is valid if for every memory m such that $m \models \phi$, we have $\mathcal{P}_i[\![c]\!](m)[\bar{\psi}] \leq \beta$.*

Figure 2 presents the deduction rules of aHL. Let us comment on some of these rules. The rule (*Rand*) handles sampling from a distribution d; we can assume a postcondition ψ after the sampling, provided that under the assumption of precondition ϕ, the statement ψ fails with probability at most β.

The other rules are similar to standard Hoare logic rules annotated with suitable probability indexes β. The rule (*Seq*) says that when composing two programs c and c', the failure probabilities of the two programs with respect to their postconditions add together. The (*Cond*) rule states that if the two branches of the conditional have the same index β, then we can keep the index β for the conditional. In rule (*Weak*) the premise $\models \phi' \Rightarrow \phi$ means that, in any model, ϕ' implies ϕ. This (*Weak*) rule allows to strengthen the precondition, weaken the postcondition, and increase the index β (which means overapproximating the failure probability). The (*And*) rule can be seen as an application of the union bound principle. It enables to combine two postconditions by a conjunction, provided we add up the failure probabilities. As to the (*Or*) rule, it allows to take the disjunction of two preconditions, if they have the same failure probability, and keep this index for the disjunction. Note that thanks to the (*Weak*) rule we could also in (*Or*) consider two indexes β and β' in the premises, and their maximum in the conclusion (the same is also true for (*Cond*)). The rule (*False*) might first seem a bit strange as it allows to conclude false, but note that its index is 1, which means that false holds in the final memory with probability 0. Finally, considering the

[5] We thank the anonymous reviewer of another paper for pointing out to us the fact that this property is derivable in GKAT.

[6] Note that ϕ and ψ are properties of memories rather that properties of distributions over memories.

(*While*) rule, observe that it is slightly more restrictive than the corresponding classical one of Hoare logic. Its side conditions ensure that the loop terminates in at most k iterations except with probability $k\beta$. Its first side condition states that the variable b_v only takes non-negative integer values.

- *Skip*:

$$\frac{}{\vdash_0 \mathtt{skip} : \phi \Rightarrow \phi}$$

- *Assn*:

$$\frac{}{\vdash_0 \mathbf{do}\ x \leftarrow t : \phi[t/x] \Rightarrow \phi}$$

- *Seq*:

$$\frac{\vdash_\beta c : \phi \Rightarrow \phi' \quad \vdash_{\beta'} c' : \phi' \Rightarrow \phi''}{\vdash_{\beta+\beta'} c; c' : \phi \Rightarrow \phi''}$$

- *Weak*:

$$\frac{\models \phi' \Rightarrow \phi \quad \vdash_\beta c : \phi \Rightarrow \psi \quad \models \psi \Rightarrow \psi' \quad \beta \leq \beta'}{\vdash_{\beta'} c : \phi' \Rightarrow \psi'}$$

- *Or*:

$$\frac{\vdash_\beta c : \phi \Rightarrow \psi \quad \vdash_\beta c : \phi' \Rightarrow \psi}{\vdash_\beta c : \phi \vee \phi' \Rightarrow \psi}$$

- *While*:

$$\frac{b_v : \mathbb{N}, \qquad \models (\phi \wedge b_v = 0 \to \bar{b}),\ \vdash_\beta c : \phi \Rightarrow \phi,\ \forall_{\eta>0} : \vdash_0 c : \phi \wedge b \wedge (b_v = \eta) \Rightarrow (b_v < \eta)}{\vdash_{k\cdot\beta} \mathbf{while}\ b\ \mathbf{do}\ c : \phi \wedge (b_v \leq k) \Rightarrow \phi \wedge \bar{b}}$$

- *Rand*:

$$\frac{\forall m : m \models \phi \Rightarrow \mathcal{P}_i[\![x \xleftarrow{\$} d]\!](m)[\bar{\psi}] \leq \beta}{\vdash_\beta \mathbf{do}\ x \xleftarrow{\$} d : \phi \Rightarrow \psi}$$

- *Cond*:

$$\frac{\vdash_\beta c : \phi \wedge b \Rightarrow \psi \quad \vdash_\beta c' : \phi \wedge \bar{b} \Rightarrow \psi}{\vdash_\beta \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c' : \phi \Rightarrow \psi}$$

- *And*:

$$\frac{\vdash_\beta c : \phi \Rightarrow \psi \quad \vdash_{\beta'} c : \phi \Rightarrow \psi'}{\vdash_{\beta+\beta'} c : \phi \Rightarrow \psi \wedge \psi'}$$

- *False*:

$$\frac{}{\vdash_1 c : \phi \Rightarrow \bot}$$

Figure 2 Approximate Hoare Logic rules (aHL).

4 Approximate Guarded Kleene algebra with tests (aGKAT)

4.1 Definition and theory of aGKAT

Recalling that GKAT encodes only Boolean assertions on probabilistic programs, we want to extend this kind of reasoning in order to capture aHL properties. We want to define a structure which would allow to express the fact that a probabilistic program c satisfies a deterministic postcondition, *except* with a probability up to a certain bound. For that we will extend GKAT with a relation between a GKAT expression and a value β from a partially ordered set. Such a set is defined as follows:

▶ **Definition 7.** *A preordered double monoid (pod-monoid) is a* $\mathcal{M} = (M, \leq, \cdot, 1, +, 0)$ *where:*

- $\leq$ *is a preorder on* M*,*

- $(M, \cdot, 1)$ *and* $(M, +, 0)$ *are two monoid structures, whose operations* $\cdot$ *and* $+$ *are monotone w.r.t.* $\leq$.

Note that we do not include any axiom relating $\cdot$ and $+$. This structure is thus sufficient to model the probability bounds from aHL. In the sequel we will consider the pod-monoid consisting of the real unit interval $[0, 1]$ equipped with multiplication and addition truncated to 1, that is to say $\min((\beta_1 + \beta_2), 1)$, where $+$ is the ordinary addition.

We then give the main definition of this section.

▶ **Definition 8.** *Approximate GKAT, denoted as aGKAT, is an extension of GKAT with a pod-monoid* $\mathcal{M}$ *and a predicate symbol* $\triangleleft$ *on* $\mathtt{Exp} \times \mathcal{M}$. *The theory of aGKAT is the union of axioms of pod-monoid, and those of Fig. 1 and Fig. 3.*

$$(c_1 = c_2 \ \wedge \ c_1 \triangleleft \beta) \ \Rightarrow \ c_2 \triangleleft \beta \quad (14)$$

$$(c \triangleleft \beta_1 \ \wedge \ \beta_1 \leq \beta_2) \ \Rightarrow \ c \triangleleft \beta_2 \quad (15)$$

$$(c \cdot c_1 \triangleleft \beta_1 \ \wedge \ c \cdot c_2 \triangleleft \beta_2) \ \Rightarrow \ c \cdot (c_1 +_b c_2) \triangleleft \beta_1 + \beta_2 \quad (16)$$

$$(c_1 \triangleleft \beta_1 \ \wedge \ c_2 \triangleleft \beta_2) \ \Rightarrow \ c_1 \cdot c_2 \triangleleft \beta_1 \cdot \beta_2 \quad (17)$$

$$(c_1 \triangleleft \beta \ \wedge \ c_2 \triangleleft \beta) \ \Rightarrow \ c_1 +_b c_2 \triangleleft \beta \quad (18)$$

$$c \triangleleft 1 \quad (19)$$

$$0 \triangleleft 0 \quad (20)$$

Figure 3 Axioms on the relation $\triangleleft$.

Recall that the intended meaning of $\triangleleft$ in the case where $\mathcal{M} = ([0, 1], \leq, \cdot, 1, +, 0)$ is that $c \triangleleft \beta$ holds if the probability of successful execution of program c is bounded by β. Observe that the $\triangleleft$-axioms of Fig. 3 are arguably simple, as they are Horn clauses and none deal with guarded iteration $c^{(b)}$.

Let us explain some intuitions underlying these axioms. Axiom (19) simply says that any program has a probability of successful execution bounded by 1, while (20) states that program 0 (which is **assert false**) has probability 0 of successful execution. Axiom (15) says that the statement still holds if we increase the probability β_1. Axiom (14) states that programs which are equal (up to the GKAT axioms of Fig. 1) admit the same probability of successful execution. Axiom (18) says that if the two branches of a conditional admit a bound β for their successful execution, so does the conditional itself. As to Axiom (17), its meaning is that the probability of successful execution of the composition of two programs c_1 and c_2 is bounded by the product of the probabilities of successful execution of respectively c_1 and c_2.

Maybe the less intuitive axiom is Axiom (16). Note that the difference with Axiom (18) is that for Axiom (18) any initial state s either satisfies b or $\overline{b}$, and so only one branch of $c_1 +_b c_2$ is explored. By contrast in Axiom (16) any initial state s might lead by the probabilistic execution of c *both* to states satisfying b and to states satisfying $\overline{b}$, so triggering both branches of the conditional. We will come back to this axiom in Remark 12 below.

After these intuitive considerations, we now formally define a semantic interpretation of aGKAT as follows:

▶ **Definition 9.** *A probabilistic interpretation of aGKAT is obtained by extending a probabilistic interpretation* $\mathcal{P}_i$ *of GKAT given in Sect. 2.2 in the following way:*

- *we consider the triple* $i = (\mathtt{State}, eval, sat)$ *of Def. 2 interpreting GKAT,*
- *the pod-monoid* $\mathcal{M}$ *is interpreted as indicated above by* $([0, 1], \leq, \cdot, 1, +, 0)$ *where* $\cdot$ *is the product and* $+$ *the truncated sum,*
- *the predicate* $\triangleleft$ *is interpreted by the relation between sub-Markov kernels* f *and* $[0, 1]$*-reals* β *consisting in the pairs* (f, β) *satisfying* $\forall s \in \mathtt{State}$, $\Sigma_{s' \in \mathtt{State}} f(s)(s') \leq \beta$, *i.e. for any* s, *the total mass of the sub-distribution* $f(s)$ *is bounded by* β.

We still denote the interpretation of an expression c as $\mathcal{P}_i[\![c]\!]$.

If i is an interpretation and F a 1st-order formula on the signature consisting of terms in Exp and in real numbers and predicates $=$ and $\triangleleft$, we write $i \models F$ if F is valid in the model defined by i. By abuse we will simply write $\models F$ if i is clear from the context. So by the definition above we have in particular that $i \models c \triangleleft \beta$ if $\forall s \in \texttt{State}$, $\sum_{s' \in \texttt{State}} \mathcal{P}_i[\![c]\!](s)(s') \leq \beta$.

We now establish the following proposition.

▶ **Proposition 10** (Soundness of aGKAT). *Any probabilistic interpretation of aGKAT is a model of its theory, i.e.:*

1. *the interpretation of aGKAT expressions satisfies the axioms of GKAT (Fig. 1) and that of $([0,1], \leq, \cdot, 1, +, 0)$ satisfies the axioms of pod-monoid,*
2. *the axioms of Fig. 3 (axioms (14) to (20)) are satisfied.*

Proof. (Prop. 10) See [13]. The most delicate case is that of Axiom (16). ◀

▶ Remark 11. Proposition 10 implies that if i is a probabilistic interpretation and if a statement $c \triangleleft \beta$ is derivable from the aGKAT axioms and possibly some semantic hypothesis of the shape $i \models A$, then $i \models c \triangleleft \beta$ holds. Note that Prop. 10 implies in particular that if i is a probabilistic interpretation and $A \Rightarrow B$ is an instance of an axiom in Fig. 3, then if $i \models A$ we can deduce that $i \models B$. This is because as i is a model, classical logic rules are sound in it.

▶ Remark 12. Note that by analogy with Axiom (18), one could have expected an axiom stronger than Axiom (16), namely that if $(c \cdot c_1 \triangleleft \beta \ \wedge \ c \cdot c_2 \triangleleft \beta)$ then one would have $c \cdot (c_1 +_b c_2) \triangleleft \beta$ (this would then generalize Axiom (18) when taking $c = 1$). However it turns out that this candidate additional axiom is *not* valid in the probabilistic model. A counter-example is given in [13].

▶ **Proposition 13.** *The following property is derivable in aGKAT:*

$$(c \cdot b_1 \triangleleft \beta_1 \ \wedge \ c \cdot b_2 \triangleleft \beta_2) \Rightarrow c \cdot (b_1 + b_2) \triangleleft \beta_1 + \beta_2$$

Proof. Observe that $b_1 + b_2 = b_1 +_{b_1} b_2$ by Prop. 5 and use Axiom (16). ◀

The proposition below refines in some sense Axiom (16).

▶ **Proposition 14.** *The following property is derivable in aGKAT:*

$$(c \cdot b \cdot c_1 \triangleleft \beta_1 \ \wedge \ c \cdot \bar{b} \cdot c_2 \triangleleft \beta_2) \Rightarrow c \cdot (c_1 +_b c_2) \triangleleft \beta_1 + \beta_2$$

Proof. Observe that $c_1 +_b c_2 = b \cdot c_1 +_b \bar{b} \cdot c_2$ by Axiom (4), Axiom (2), applied two times. Then apply Axiom (16) to c, $c'_1 = bc_1$ and $c'_2 = \bar{b}c_2$. ◀

▶ Remark 15 (Axiom (16), left distributivity and union bound). Recall that KAT [21] has an axiom of *left distributivity* $c \cdot (c_1 + c_2) = c \cdot c_1 + c \cdot c_2$. It does not hold in GKAT with the guarded sum $+_b$ though. In some sense axiom (16) (or its refinement Prop. 14) can be seen as a kind of compensation for this lack of left distributivity because it allows, when one is reasoning about an expression $c \cdot (c_1 +_b c_2)$ (in order to establish a bound β), to continue the proof with two branches, respectively on $c \cdot c_1$ and on $c \cdot c_2$. If one obtains two bounds $c \cdot c_i \triangleleft \beta_i$, for $i = 1, 2$ then one can deduce that $c \cdot (c_1 +_b c_2) \triangleleft \beta_1 + \beta_2$.

Moreover if c_1 and c_2 are tests b_1 and b_2, then by Prop. 5 $b_1 + b_2 = b_1 +_{b_1} b_2$. So $c \cdot (b_1 + b_2) = c \cdot (b_1 +_{b_1} b_2) \triangleleft \beta_1 + \beta_2$. So the probability that after execution of c the test $(b_1 + b_2)$ is satisfied is inferior to the sum of the probability that b_1 is satisfied and of the probability that b_2 is satisfied. This is the application of the binary union bound principle on post-conditions, and it can easily be applied to an arbitrary union bound.

4.2 Semantic reasoning

When reasoning about concrete programs, we want to establish properties on their semantic interpretations. That might sometimes require, besides the axioms of aGKAT, the use of some semantic properties. One such example is that some actions can be commuted without changing the semantics of the program. We establish thus some notations:

▶ **Definition 16.** *Given two GKAT program c and c' and a probabilistic interpretation i, we write $c \equiv c'$ if $i \models c = c'$, i.e. $\mathcal{P}_i[\![c]\!] = \mathcal{P}_i[\![c']\!]$.*

This definition is required to establish the following proposition.

▶ **Proposition 17.** *Consider GKAT programs c and c', and a probabilistic interpretation i.*

1. *If b is a test which only depends on the values of some variables $x_1, \ldots, x_n$ and if c leaves the values of those variables unchanged, then we have $c \cdot b \equiv b \cdot c$,*
2. *If $c \equiv c'$ and $i \models c \triangleleft \beta$, then $i \models c' \triangleleft \beta$.*

Observe that (1) holds because the syntax of programs does not allow any form of aliasing and (2) because the property $i \models c \triangleleft \beta$ only depends on the semantic interpretation $\mathcal{P}_i[\![c]\!]$.

Let us now illustrate the use of aGKAT on a small example.

▶ **Example 18** (Double tossing)**.** Consider the program c below:

$$c \quad = \quad (x \overset{\$}{\leftarrow} Coin) \cdot (c_1 +_{(x=1)} y \leftarrow 0), \qquad \text{where } c_1 = (x \overset{\$}{\leftarrow} Coin) \cdot (y \leftarrow 1 +_{(x=1)} y \leftarrow 0)$$

Consider the interpretation i where $Coin$ is the distribution of a fair coin, that takes value 0 (resp. 1) with probability 1/2 (resp. 1/2). This can be represented either by adding to the theory two axioms describing the behaviour of $Coin$, namely axioms $(x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \triangleleft 1/2$ and $(x \overset{\$}{\leftarrow} Coin) \cdot (x \neq 1) \triangleleft 1/2$, or by using the following semantic properties of i: $\models (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \triangleleft 1/2$ and $\models (x \overset{\$}{\leftarrow} Coin) \cdot (x \neq 1) \triangleleft 1/2$. We want to prove that after the execution of c, the probability that y equals 0 is below 3/4, and the probability that y equals 1 is below 1/4, i.e. $\models c \cdot (y = 0) \triangleleft 3/4$ and $\models c \cdot (y = 1) \triangleleft 1/4$.

Recall first that as i is a model, by Prop. 10, it satisfies all axioms of Fig. 1 and Fig. 3, and all classical logic rules are sound in it (see Remark 11). Now, by using Axiom (5), $c \cdot (y = 0)$ can be rewritten as follows:

$$\begin{aligned} c \cdot (y = 0) \quad &= \quad (x \overset{\$}{\leftarrow} Coin) \cdot (c'_1 +_{(x=1)} y \leftarrow 0(y = 0)) \\ c'_1 \quad &= \quad (x \overset{\$}{\leftarrow} Coin) \cdot (y \leftarrow 1(y = 0) +_{(x=1)} y \leftarrow 0(y = 0)) \end{aligned}$$

Let us name the following expressions, corresponding to the various possible branches of executions of $c \cdot (y = 0)$:

$$\begin{aligned} c_2 \quad &= \quad (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \cdot (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \cdot (y \leftarrow 1) \cdot (y = 0) \\ c_3 \quad &= \quad (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \cdot (x \overset{\$}{\leftarrow} Coin) \cdot (x \neq 1) \cdot (y \leftarrow 0) \cdot (y = 0) \\ c_4 \quad &= \quad (x \overset{\$}{\leftarrow} Coin) \cdot (x \neq 1) \cdot (y \leftarrow 0) \cdot (y = 0) \end{aligned}$$

First, from the model we know that $(y \leftarrow 1) \cdot (y = 0) \equiv 0$, so $c_2 \equiv 0$, so $\models c_2 \triangleleft 0$.

Then, as $\models (x \overset{\$}{\leftarrow} Coin) \cdot (x \neq 1) \triangleleft 1/2$, by Axioms (19) and (17) we have $\models c_4 \triangleleft 1/2$.

Then, as $\models (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \triangleleft 1/2$, by Axioms (17) and (19) we have $\models c_3 \triangleleft 1/4$.

By applying Prop. 14 to c_2 and c_3 we get: $\models (x \overset{\$}{\leftarrow} Coin) \cdot (x = 1) \cdot c'_1 \quad \triangleleft \quad 1/4 \qquad (21)$

By applying again Prop. 14, this time to (21) and by $\models c_4 \triangleleft 1/2$ we finally obtain $\models c \cdot (y = 0) \triangleleft 3/4$ $(= 1/4 + 1/2)$. We give in [13] a step-by-step fully explicit version of the proof above. The proof that $\models c \cdot (y = 1) \triangleleft 1/4$ holds is similar.

5 Encoding aHL in aGKAT

We want to relate deduction in aHL and reasoning in aGKAT, by following the approach of [22] on the encoding of propositional Hoare logic in KAT. We consider the programming language of Example 1 but the results remain valid if we consider extended grammars of terms, distributions, tests and commands, where the class of tests is closed by substitution of terms t (as in Example 1). We will encode aHL derivations consisting of judgements $\vdash_\beta c : \phi \Rightarrow \phi'$ where ϕ and ϕ' belong to the class of tests.

Concretely the idea will be to encode the aHL judgement $\vdash_\beta c : \phi \Rightarrow \phi'$ by the aGKAT statement $\phi \cdot c \cdot \overline{\phi'} \triangleleft \beta$. Similarly to [22], showing that an aHL rule is sound in aGKAT will consist in proving that the conjunction of the aGKAT equations encoding the premises of the aHL rule implies the equation encoding the conclusion of the rule.

Observe that similarly as for Hoare logic, some rules of aHL, namely axiom rules (*Assn*) and (*Rand*), do not depend on aHL judgements as premises but rather on an interpretation of actions and predicates, and possibly a semantic condition (for (*Rand*)). Thus we do not expect to derive their encoding as an equation valid in the theory of aGKAT. Instead, one could add new axioms corresponding to (*Assn*) and (*Rand*) for specific distributions (as mentioned in Example 18), or alternatively when dealing with examples consider a particular interpretation i and thus reason on equalities of expressions in the model.

Fig. 4 lists the interpretations of the rules of aHL (Figure 2) in aGKAT, by encoding aHL judgments as aGKAT equations. Note that the rule (*Assn*) uses the test $\phi[t/x]$ obtained by substituting the term t in ϕ, which does belong to the class of tests by definition.

- Skip:
$$\phi 1 \bar{\phi} \triangleleft 0 \tag{22}$$
- Assn:
$$\phi[t/x](x \leftarrow t)\bar{\phi} \triangleleft 0 \tag{23}$$
- Rand:
$$(\forall m, m \models \phi \Rightarrow \mathcal{P}_i[\![x \xleftarrow{\$} d]\!](m)[\bar{\psi}] \leq \beta) \Rightarrow \phi(x \xleftarrow{\$} d)\bar{\psi} \triangleleft \beta \tag{24}$$
- Seq:
$$(\phi c \bar{\phi'} \triangleleft \beta) \wedge (\phi' c' \bar{\phi''} \triangleleft \beta') \Rightarrow \phi c c' \bar{\phi''} \triangleleft \beta + \beta' \tag{25}$$
- Cond:
$$(\phi b c \bar{\psi} \triangleleft \beta) \wedge (\phi \bar{b} c' \bar{\psi} \triangleleft \beta) \Rightarrow \phi(c +_b c')\bar{\psi} \triangleleft \beta \tag{28}$$
- Weak:
$$(\phi \leq \phi') \wedge (\phi c \bar{\psi} \triangleleft \beta) \wedge (\psi' \leq \psi) \wedge (\beta \leq \beta') \Rightarrow \phi' c \bar{\psi'} \triangleleft \beta' \tag{26}$$
- And:
$$(\phi c \bar{\psi} \triangleleft \beta) \wedge (\phi c \bar{\psi'} \triangleleft \beta') \Rightarrow \phi c \psi \bar{\psi'} \triangleleft \beta + \beta' \tag{29}$$
- Or:
$$(\phi c \bar{\psi} \triangleleft \beta) \wedge (\phi' c \bar{\psi} \triangleleft \beta) \Rightarrow (\phi + \phi') c \bar{\psi} \triangleleft \beta \tag{27}$$
- False:
$$\phi c \bar{\bot} \triangleleft 1 \tag{30}$$
- While:
$$(\models b_v \in \mathbb{N}) \wedge (\models (\phi \wedge (b_v \leq 0)) \to \bar{b}) \Rightarrow (\phi c \bar{\phi} \triangleleft \beta) \wedge (\forall_{\eta > 0} \cdot \phi b [b_v = \eta] c \overline{[b_v < \eta]} \triangleleft 0) \Rightarrow \phi[b_v \leq k] c^{(b)} \overline{\phi \bar{b}} \triangleleft k\beta \tag{31}$$

Figure 4 Interpretation of aHL rules in aGKAT.

The next theorem establishes the main result of the paper.

▶ **Theorem 19.** *All the rules of the system aHL, union bound logic, except* (Assn) *and* (Rand)*, have an aGKAT interpretation (in Fig. 4) that is derivable from the axioms of aGKAT.*

Note that the interpretation of the (**While**) rule, (31) in Fig. 4, is not a plain aGKAT formula, but has some semantic premises. This is because the aHL (**While**) rule itself is expressed with semantic premises. The proof of Theorem 19 can be found in [13]. The most interesting cases are (**Seq**) and (**And**) rules, and the most difficult one is that of (**While**).

Observe that an interesting feature of aGKAT is that none of its $\triangleleft$-axioms (Fig. 3) refers to guarded iteration $c^{(b)}$, and nevertheless aGKAT is as expressive as aHL and allows to derive its (**While**) rule. Another interesting specificity of aGKAT w.r.t. to aHL is that in aGKAT the axioms for reasoning on program equivalence (those of GKAT, Fig. 1) are disjoint from those for reasoning on probabilities (Fig. 3).

6 Example

We now consider the example of the *Report-noisy-max* algorithm, which has been analysed in [5] with the logic aHL. Our analysis here using aGKAT will be similar, but the equational approach of aGKAT will simplify some steps. The full proof can be found in [13].

We consider a finite set $\mathcal{R}$ and a quality score function $qscore$, which takes as input a pair of an element r of $\mathcal{R}$ and a database d, and returns a real number. The goal of the algorithm is to find an element r^* of $\mathcal{R}$ which approximately minimizes the function $qscore$ on d. The algorithm is randomized and only computes an approximate minimization because it is designed to satisfy a differential privacy property (see [9]). The algorithm proceeds by computing for each element r of $\mathcal{R}$ the quality score $qscore(r, d)$ and adding to it a Laplacian noise (according to the Laplace mechanism for differential privacy [9]) and returning the element r^* with the highest noisy value.

Here we do not deal with the privacy property of this program, but instead our objective is to study its *accuracy*, that is to say to bound the difference between the value of $qscore(r^*, d)$ and the real minimum of $qscore(\cdot, d)$ on $\mathcal{R}$. The algorithm is encoded in GKAT as the program $c = (flag \leftarrow 1); (best \leftarrow 0); (\mathcal{R}_0 \leftarrow \mathcal{R}); (\mathcal{R}' \leftarrow \emptyset); c'^{[\mathcal{R} \neq \emptyset]}; return(r^*)$ where

$$c' = (r \leftarrow pick(\mathcal{R})); (noisy[r] \overset{\$}{\leftarrow} \mathcal{L}_{\epsilon/2}(qscore(r, d))); (c_1 +_b 1); (\mathcal{R} \leftarrow \mathcal{R} \backslash \{r\}); (\mathcal{R}' \leftarrow \mathcal{R}' \cup \{r\})$$
$$\text{where } c_1 = (flag \leftarrow 0); (r^* \leftarrow r); (best \leftarrow noisy[r]) \qquad b = (noisy[r] > best) + (flag == 1)$$

The variable $flag$ has Boolean values ($\{0, 1\}$), $\mathcal{R}$, $\mathcal{R}_0$ and $\mathcal{R}'$ are sets, r, r^* range over elements of $\mathcal{R}$, $noisy[r]$ and $best$ range over reals. The variable *flag* is used for initialization purpose. The notation $noisy[r]$ is an array-like notation for representing n variables, where n is the size of the set $\mathcal{R}$. Note that variable $\mathcal{R}'$ does not play any role in the algorithm, it is just used to express properties of the execution.

This program uses the following kinds of actions and tests:

- actions for operations on sets: picking an (arbitrary) element r from a set $(r \leftarrow pick(\mathcal{R}))$, removing ($\mathcal{R} \leftarrow \mathcal{R} \backslash \{r\}$)) and adding an element $(\mathcal{R}' \leftarrow \mathcal{R}' \cup \{r\})$,
- sampling from a Laplacian distribution centered in a with parameter p : $(x \overset{\$}{\leftarrow} \mathcal{L}_p(a))$,
- tests: inequalities for reals, equality for Boolean value, comparison to empty set for sets $[\mathcal{R} \neq \emptyset]$; we will also need a finite number of additional tests for expressing properties on the execution, that we will see later.

We recall the following accuracy property of the Laplace distribution [5]:

▶ **Lemma 20.** *Let $\beta \in [0, 1]$, ν a sample from $\mathcal{L}_p(a)$. Then $Pr_{\mathcal{L}_p(a)}[|\nu - a| > \frac{1}{p} \log(\frac{1}{\beta})] < \beta$.*

Therefore we have $\models (x \xleftarrow{\$} \mathcal{L}_p(a))[|x-a| > \frac{1}{p}\log(\frac{1}{\beta})] \triangleleft \beta$. Hence for the sampling in c:

$$\models (noisy[r] \xleftarrow{\$} \mathcal{L}_{\epsilon/2}(qscore(r,d))) \cdot \overline{b_1} \triangleleft \frac{\beta}{|\mathcal{R}_0|} \tag{32}$$

where $\overline{b_1} = [\,|noisy[r] - qscore(r,d)| > \frac{2}{\epsilon}\log(\frac{|\mathcal{R}_0|}{\beta})]$.

Now we want to establish a property for the whole program c'. Observe that $\overline{b_1}$ only depends on the values of $noisy[r]$ and $qscore(r,d)$. Moreover $noisy[r]$ and $qscore(r,d)$ are not changed by the last 3 actions of c'. Therefore by applying Prop.17.1 we get $c'; \overline{b_1} \equiv c''$, where c'' is obtained by inserting in c' the test $\overline{b_1}$ just after $(noisy[r] \xleftarrow{\$} \mathcal{L}_{\epsilon/2}(qscore(r,d)))$. As we know by (19) that for any c_0 we have $\models c_0 \triangleleft 1$, by combining this with axiom (17) and (32) we get $\models c'' \triangleleft \frac{\beta}{|\mathcal{R}_0|}$. This is a step where aGKAT has provided us a concise and simple reasoning. Therefore, as $c'; \overline{b_1} \equiv c''$, we get by Prop. 17.2: $\models c' \cdot \overline{b_1} \triangleleft \frac{\beta}{|\mathcal{R}_0|}$.

We want to prove an invariant for the body c' of the *while* loop in c. For that consider the test b_2 corresponding to the predicate $\phi_2 = \forall r \in \mathcal{R}',\ |noisy[r] - qscore(r,d)| \leq \frac{2}{\epsilon}\log(\frac{|\mathcal{R}_0|}{\beta})$.

$$\text{We have:} \quad b_2 \cdot b_1 \cdot (\mathcal{R}' \leftarrow \mathcal{R}' \cup \{r\}) \cdot \overline{b_2} \quad \equiv \quad 0 \tag{33}$$

Let c_2 be c' deprived of the last action, i.e. $c' = c_2 \cdot (\mathcal{R}' \leftarrow \mathcal{R}' \cup \{r\})$. The reasoning we did on c' before can be repeated for c_2, and so as for c' we get: $\models c_2 \cdot \overline{b_1} \triangleleft \frac{\beta}{|\mathcal{R}_0|}$. Therefore by axioms (17) and (19) we get $\models b_2 \cdot c_2 \cdot \overline{b_1} \triangleleft \frac{\beta}{|\mathcal{R}_0|}$ (here again aGKAT helps us with conciseness). Moreover as c_2 does not modify $\mathcal{R}'$, by using Prop.17.1 we get $\models b_2 \cdot c_2 \cdot \overline{b_2} \triangleleft 0$. Thus by using the aGKAT encoding (Theorem 19) of the aHL rule (And) we obtain from the two previous statements: $\models b_2 \cdot c_2 \cdot \overline{b_1 \cdot b_2} \triangleleft \frac{\beta}{|\mathcal{R}_0|}$. Equation (33) gives us $\models (b_1 \cdot b_2) \cdot (\mathcal{R}' \leftarrow \mathcal{R}' \cup \{r\}) \cdot \overline{b_2} \triangleleft 0$.

By using the aGKAT encoding of the aHL rule (Seq) we get from the two last statements: $\models b_2 \cdot c' \cdot \overline{b_2} \triangleleft \frac{\beta}{|\mathcal{R}_0|}$. By using the aGKAT encoding of the aHL rule (While) we get from this last statement, since the loop runs for $|\mathcal{R}_0|$ iterations, $\models b_2 \cdot c'^{[\mathcal{R} \neq \emptyset]} \cdot \overline{b_2} \triangleleft \beta$.

Finally as $(\mathcal{R}' \leftarrow \emptyset) \cdot \overline{b_2} \equiv 0$ we deduce from this statement using (Seq) that $\models c \cdot \overline{b_2} \triangleleft \beta$. So we have proven using aGKAT that the property corresponding to the following judgement holds: $\vdash_\beta c : \top \Rightarrow \forall r \in \mathcal{R}',\ |noisy[r] - qscore(r,d)| \leq \frac{2}{\epsilon}\log(\frac{|\mathcal{R}_0|}{\beta})$.

In [13] we continue the proof to finally obtain an accuracy bound for the algorithm.

7 Related work

Several works have explored the use of program logics for the verification of probabilistic programs. Some of these works have explored approaches based on Hoare-like logics [16] while some other ones have developed the approach of weakest-pre-expectations, e.g. [25, 17]. The paper [7] has extended standard Hoare logic to deal with a language containing a probabilistic choice operator, and in which predicates express claims about the state of a probabilistic program. In this work, a semantics for the language is given and a Hoare-style deduction system presented, and proven to be correct w.r.t. the semantics. Another example is the union bound logic aHL [5] that we already presented. More recently, Graded Hoare logic (GHL) was introduced in [10] as a parameterisable framework for extending Hoare logic with a preordered monoidal analysis, with a few examples of applications: the union bound logic aHL [5]; logics for analysis of computation time; or the logic for reasoning about program counter security [26]. Other works even explore extensions of propositional dynamic logic to probabilistic programs, as [23]. This article proposes a probabilistic analog of PDL, which generalises the non-deterministic logical constructs and proof rules in PDL to arithmetic analogs in the probabilistic version.

Other approaches to probabilistic program verification were also introduced in the literature, relying on algebraic structures to wrap the apparatus of logical systems into more elegant frameworks. Some of these approaches are extension of Kleene algebra with tests, of which we give a few examples. A probabilistic extension of GKAT (ProbGKAT) was introduced in [29] for reasoning about imperative programs with probabilistic branching. One difference to our approach is the syntactic introduction of the probability, by the operator $\oplus_r$, where r is a probability; we rather do it semantically, by taking samplings as basic instructions, more in the style of the probabilistic assignment operator of the language `pIMP` [15] for instance. Additionally, [29] provides an axiomatisation of bisimilarity of ProbGKAT expressions and proves its completeness.

Another KAT extension was given in [11] which aimed at capturing fuzzy programs by replacing the Boolean algebra of KAT by a lattice, with the goal of being able to reason on fuzzy (non Boolean) properties [14].

A variant of GKAT was introduced in [12] as a relational structure to reason about properties of pairs of probabilistic programs (e.g. non-interference between variables), in the style of the system BiKAT [3] in the non-probabilistic setting. Still in the domain of relational reasoning, we can mention relational differential dynamic logic [20], which is specifically designed for the verification of *cyber-physical systems*, in a process that the authors called *synchronizing* the dynamics for comparing two systems.

8 Discussion and future work

We believe that a promising aspect of aGKAT and of the axiomatic presentation we introduced in this paper, is that they can contribute to extend the range of applicability of (co)-algebraic techniques of verification illustrated e.g. in [1, 24, 30, 2] to the realm of approximate reasoning on program effects [5, 6, 10]. This suggests several exciting research directions, which we discuss below.

Towards decision procedures. Recall that the paper [30] has given a decision procedure for the equivalence of GKAT programs whose complexity is almost linear time, assuming that the number of tests is fixed. This procedure is based on a new automata construction. One could investigate in an analogous way decision problems in aGKAT for statements of the form $c \triangleleft \beta$. The problem could be expressed with some semantic hypothesis, typically some probabilistic assumptions on the randomized primitives used by the program. Our axioms on the relation $\triangleleft$ (Fig. (3)) are quite promising in this respect since they are Horn clauses. Combining automata methods [30] and Horn clauses deduction techniques might lead to some efficient procedures. In [29], the authors present a decision procedure for demonstrating the existence of bisimilarity between two ProbGKAT expressions. Note that the probabilistic constructs of ProbGKAT can be encoded in aGKAT. Consider for $r \in [0, 1]$ the distribution $\mathtt{flip}(r)$ which returns 0 (resp. 1) with probability r (resp. $(1 - r)$). Then by using a fresh variable x, $c_0 \oplus_r c_1$ can be encoded as $(x \overset{\$}{\leftarrow} \mathtt{flip}(r)) \cdot (c_0 +_{[x=0]} c_1)$, and $c^{[r]}$ as $(x \overset{\$}{\leftarrow} \mathtt{flip}(r)) \cdot ((x \overset{\$}{\leftarrow} \mathtt{flip}(r)) \cdot c)^{[x=0]}$. Using this encoding and axiom (16) one obtains that from $c_i \triangleleft \beta_i$ for $i = 0, 1$ one can derive $c_0 \oplus_r c_1 \triangleleft r\beta_0 + (1 - r)\beta_1$.

Lower bounds. The system aGKAT has been defined to derive probabilistic upper bounds on successful termination, which allows to obtain upper bounds on the satisfaction or failure of a postcondition. It would be interesting to investigate if this approach can be adapted to derive probabilistic lower bounds, to prove that a postcondition holds with probability *at least* β.

Extension to other effects. In the present paper we restricted ourselves to a specific pod-monoid with specific interval of values and set of operators, which were enough to capture aHL and handle the initial intended goals of reasoning on probabilistic properties. However we would like to push this approach further. By using a generic and external structure to the main algebraic model of programs, we could follow a parametric approach, and obtain more freedom on the structure chosen to capture a wider range of quantitative analysis of effects, like for instance: the analysis of computation time model, by taking the natural numbers and the arithmetic sum as the monoidal composition; the program counter security model, by taking a set of binary values and the string concatenation as the composition; and the union bound logic itself. Those are a few concrete models considered for a generic version of Hoare Logic, analysed in [10].

We want to stress however that it is not trivial to capture in the same generic setting both the union bound logic and the logic for analysis of computation time. The system aGKAT as it stands does not allow to do that. In particular axiom (19) implies that the neutral element of the first monoid is also maximal for the order; this is not the case in the monoid $(\mathbb{N}, +)$ (or even $(\mathbb{N}^\infty, +)$) used for the Hoare logic for analysis of computation time [10].

The framework of pRHL could also benefit from an algebraic approach, calling for a structure taking into account the parametric reasoning about judgments themselves. One would need to embed the parameters into the structure itself, resorting, for example, to a relation between algebraic terms and the elements from the structure which model these parameters.

Towards a stronger completeness. Another possible direction for future work would be to study completeness of aGKAT with respect to some class of Horn clauses which embed aHL rules. That would mean to prove that the theory of aGKAT could always derive equations that represent valid aHL rules. We could draw inspiration from Kozen's classical work [22], in which an analogous result was proven for KAT with respect to a class of Horn clauses which embed propositional Hoare logic.

Towards relational properties. In this paper we have considered properties on single executions of a program, but some important questions can be expressed as relational properties on pairs of execution, for instance non-interference, continuity or sensitivity properties. An extension of Kleene algebra with tests called BiKAT for relational properties was introduced in [3] and another framework for probabilistic relational properties was proposed in [12]. It would interesting to explore if the approximation construction we defined in the present paper could be applied to the probabilistic relational setting of [12]. This would be analogous to the move in the relational Hoare logic setting, from pRHL to apRHL.

Non-determinism and probabilities. One of the advantages of the syntactical restriction of GKAT is to facilitate the inclusion of probabilistic models, by neglecting nondeterminism. While usually avoided, and always difficult, one possible direction for future work could be to consider a language with both nondeterminism and probabilities, capturing more application scenarios.

References

1 D. Kozen A. Angus. Kleene algebra with tests and program schematology. Technical report, Computer Science Department, Cornell University, Ithaca, NY 14853-7501, USA, July 2001. Technical Report TR2001-1844.

2 Carolyn Jane Anderson, Nate Foster, Arjun Guha, Jean-Baptiste Jeannin, Dexter Kozen, Cole Schlesinger, and David Walker. NetKAT: semantic foundations for networks. In Suresh Jagannathan and Peter Sewell, editors, *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pages 113–126. ACM, 2014. doi:10.1145/2535838.2535862.

3 Timos Antonopoulos, Eric Koskinen, Ton Chanh Le, Ramana Nagasamudram, David A. Naumann, and Minh Ngo. An algebra of alignment for relational verification. *Proc. ACM Program. Lang.*, 7(POPL):573–603, 2023. doi:10.1145/3571213.

4 Gilles Barthe, François Dupressoir, Benjamin Grégoire, César Kunz, Benedikt Schmidt, and Pierre-Yves Strub. Easycrypt: A tutorial. In Alessandro Aldini, Javier López, and Fabio Martinelli, editors, *Foundations of Security Analysis and Design VII - FOSAD 2012/2013 Tutorial Lectures*, volume 8604 of *Lecture Notes in Computer Science*, pages 146–166. Springer, 2013. doi:10.1007/978-3-319-10082-1_6.

5 Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. A program logic for union bounds. 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 107:1–107:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.107.

6 Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. Probabilistic relational reasoning for differential privacy. *ACM Trans. Program. Lang. Syst.*, 35(3):9:1–9:49, 2013. doi:10.1145/2492061.

7 Jerry den Hartog and Erik P. de Vink. Verifying probabilistic programs using a Hoare like logic. *Int. J. Found. Comput. Sci.*, 13(3):315–340, 2002. doi:10.1142/S012905410200114X.

8 Easycrypt development team. Easycrypt, 2024. URL: https://formosa-crypto.org/projects/.

9 Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014. doi:10.1561/0400000042.

10 Marco Gaboardi, Shin-ya Katsumata, Dominic Orchard, and Tetsuya Sato. Graded hoare logic and its categorical semantics. In Nobuko Yoshida, editor, *Programming Languages and Systems - 30th European Symposium on Programming, ESOP 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings*, volume 12648 of *Lecture Notes in Computer Science*, pages 234–263. Springer, 2021. doi:10.1007/978-3-030-72019-3_9.

11 L. Gomes, A. Madeira, and L. S. Barbosa. Generalising KAT to verify weighted computations. *Scient. Annals of Comp. Sc.*, 29(2):141–184, 2019. doi:10.7561/SACS.2019.2.141.

12 Leandro Gomes, Patrick Baillot, and Marco Gaboardi. BiGKAT: an algebraic framework for relational verification of probabilistic programs. working paper or preprint, March 2023. URL: https://hal.science/hal-04017128.

13 Leandro Gomes, Patrick Baillot, and Marco Gaboardi. A Kleene algebra with tests for union bound reasoning about probabilistic programs. working paper or preprint, July 2024. URL: https://hal.science/hal-04196675v3.

14 Leandro Gomes, Alexandre Madeira, and Luís Soares Barbosa. A semantics and a logic for fuzzy arden syntax. *Soft Comput.*, 25(9):6789–6805, 2021. doi:10.1007/s00500-021-05593-9.

15 Ichiro Hasuo, Yuichiro Oyabu, Clovis Eberhart, Kohei Suenaga, Kenta Cho, and Shin-ya Katsumata. Control-data separation and logical condition propagation for efficient inference on probabilistic programs. *J. Log. Algebraic Methods Program.*, 136:100922, 2024. doi:10.1016/J.JLAMP.2023.100922.

16 Claire Jones. *Probabilistic non-determinism.* PhD thesis, University of Edinburgh, UK, 1990. URL: https://hdl.handle.net/1842/413.

17 Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. Weakest precondition reasoning for expected runtimes of randomized algorithms. *J. ACM*, 65(5):30:1–30:68, 2018. doi:10.1145/3208102.

18 Tobias Kappé, Paul Brunet, Alexandra Silva, Jana Wagemaker, and Fabio Zanasi. Concurrent kleene algebra with observations: From hypotheses to completeness. In *Proceedings of FOSSACS 2020*, volume 12077 of *LNCS*, pages 381–400. Springer, 2020. doi:10.1007/978-3-030-45231-5_20.

19 Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994. URL: https://mitpress.mit.edu/books/introduction-computational-learning-theory.

20 Juraj Kolcák, Jérémy Dubut, Ichiro Hasuo, Shin-ya Katsumata, David Sprunger, and Akihisa Yamada. Relational differential dynamic logic. In Armin Biere and David Parker, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 26th International Conference, TACAS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings, Part I*, volume 12078 of *Lecture Notes in Computer Science*, pages 191–208. Springer, 2020. doi:10.1007/978-3-030-45190-5_11.

21 D. Kozen. Kleene algebra with tests. *ACM Trans. on Prog. Lang. and Systems*, 19(3):427–443, 1997. doi:10.1145/256167.256195.

22 D. Kozen. On Hoare logic and Kleene algebra with tests. *ACM Trans. on Comp. Logic*, 1(212):1–14, 2000. doi:10.1109/LICS.1999.782610.

23 Dexter Kozen. A probabilistic PDL. *J. Comput. Syst. Sci.*, 30(2):162–178, 1985. doi:10.1016/0022-0000(85)90012-1.

24 Dexter Kozen and Maria-Christina Patron. Certification of compiler optimizations using Kleene algebra with tests. In John W. Lloyd, Verónica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catuscia Palamidessi, Luís Moniz Pereira, Yehoshua Sagiv, and Peter J. Stuckey, editors, *Computational Logic - CL 2000, First International Conference, London, UK, 24-28 July, 2000, Proceedings*, volume 1861 of *Lecture Notes in Computer Science*, pages 568–582. Springer, 2000. doi:10.1007/3-540-44957-4_38.

25 Annabelle McIver and Carroll Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Monographs in Computer Science. Springer, 2005. doi:10.1007/B138392.

26 David Molnar, Matt Piotrowski, David Schultz, and David A. Wagner. The program counter security model: Automatic detection and removal of control-flow side channel attacks. In Dongho Won and Seungjoo Kim, editors, *Information Security and Cryptology - ICISC 2005, 8th International Conference, Seoul, Korea, December 1-2, 2005, Revised Selected Papers*, volume 3935 of *Lecture Notes in Computer Science*, pages 156–168. Springer, 2005. doi:10.1007/11734727_14.

27 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995. doi:10.1017/cbo9780511814075.

28 Damien Pous. Kleene algebra with tests and coq tools for while programs. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving - 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013. Proceedings*, volume 7998 of *Lecture Notes in Computer Science*, pages 180–196. Springer, 2013. doi:10.1007/978-3-642-39634-2_15.

29 Wojciech Rozowski, Tobias Kappé, Dexter Kozen, Todd Schmid, and Alexandra Silva. Probabilistic guarded KAT modulo bisimilarity: Completeness and complexity. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 136:1–136:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.136.

30 Steffen Smolka, Nate Foster, Justin Hsu, Tobias Kappé, Dexter Kozen, and Alexandra Silva. Guarded Kleene algebra with tests: verification of uninterpreted programs in nearly linear time. *Proc. ACM Program. Lang.*, 4(POPL):61:1–61:28, 2020. doi:10.1145/3371129.

31 Jana Wagemaker, Nate Foster, Tobias Kappé, Dexter Kozen, Jurriaan Rot, and Alexandra Silva. Concurrent NetKAT - modeling and analyzing stateful, concurrent networks. In Ilya Sergey, editor, *Programming Languages and Systems - 31st European Symposium on Programming, ESOP 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*, volume 13240 of *Lecture Notes in Computer Science*, pages 575–602. Springer, 2022. doi:10.1007/978-3-030-99336-8_21.

32 Cheng Zhang, Arthur Azevedo de Amorim, and Marco Gaboardi. On incorrectness logic and Kleene algebra with top and tests. *Proc. ACM Program. Lang.*, 6(POPL):1–30, 2022. doi:10.1145/3498690.

APPENDIX

A An example showing that aGKAT is more expressive than aHL

We have shown that aGKAT allows to encode aHL, but in this section we will show that aGKAT is more expressive than aHL, in the sense that in can prove some bounds that aHL cannot.

▶ **Example 21** (While program)**.** Let d be the distribution corresponding to a fair dice with three outcomes, that is to say that d has support $\{0,1,2\}$ and $d(0) = d(1) = d(2) = 1/3$.

We consider the program c below:

$$c \quad = \quad (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} \cdot [x = 1]$$

So c can be described as follows:

it samples d a first time and assigns the result to x; then until it obtains $(x \neq 0)$ it repeats sampling d and assigning the result to x; if at some point it obtains $(x \neq 0)$, then if $(x = 1)$ it terminates successfully, otherwise (that is to say if $(x = 2)$) it aborts.

The analysis of the probability of successful termination of c goes as follows:

with the first sample one obtains $(x = 1)$ with probability $1/3$ and then the program terminates successfully; or one obtains $(x = 2)$ with probability $1/3$ and then the program aborts; or one obtains $(x = 0)$ with probability $1/3$ and we execute c again.

So the probability of successful termination is:

$$\Sigma_{i=1}^{+\infty}(\frac{1}{3})^i = \frac{1}{3} \cdot \frac{1}{1-\frac{1}{3}} = \frac{1}{3} \cdot \frac{3}{2} = \frac{1}{2}$$

Let us now proceed with an analysis in aGKAT. We represent the properties of d with the following 4 axioms:

$$(x \xleftarrow{\$} d)[x = i] \triangleleft 1/3 \text{ for } i = 0, 1, 2 \ , (x \xleftarrow{\$} d)[x \neq 0, 1, 2] \triangleleft 0.$$

We can then derive the following proof by using GKAT axioms:

$$\begin{aligned}
& c \\
= \ & (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} \cdot [x = 1] \\
= \ & (x \xleftarrow{\$} d) \cdot ((x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} +_{[x=0]} 1) \cdot [x = 1] && \text{by ax. (11)} \\
= \ & (x \xleftarrow{\$} d) \cdot (1 +_{[x\neq 0]} (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])}) \cdot [x = 1] && \text{by ax. (2)} \\
= \ & (x \xleftarrow{\$} d) \cdot ([x \neq 0] +_{[x\neq 0]} [x = 0] \cdot (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])}) \cdot [x = 1] && \text{by ax. (4) and (2)} \\
= \ & (x \xleftarrow{\$} d) \cdot ([x \neq 0] \cdot [x = 1] +_{[x\neq 0]} [x = 0] \cdot (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} \cdot [x = 1]) && \text{by ax. (5)} \\
= \ & (x \xleftarrow{\$} d) \cdot ([x = 1] +_{[x\neq 0]} [x = 0] \cdot (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} \cdot [x = 1]) && \text{by properties of tests} \\
= \ & (x \xleftarrow{\$} d) \cdot ([x = 1] +_{[x\neq 0]} [x = 0] \cdot c)
\end{aligned}$$

We know that $(x \xleftarrow{\$} d) \cdot [x = 1] \triangleleft 1/3$ and $(x \xleftarrow{\$} d) \cdot [x = 0] \triangleleft 1/3$. By using axioms (19) and (17) we deduce that $(x \xleftarrow{\$} d) \cdot [x = 0] \cdot c \triangleleft 1/3$.

By applying axiom (16) to the last line of the derivation we obtain $c \triangleleft 1/3 + 1/3 = 2/3$.

We can then refine this bound by proceeding by recursion. Denote $\beta_0 = 1$ and $\beta_{i+1} = \frac{1}{3} \cdot (1 + \beta_i)$ for $i \geq 0$. That is to say that $\beta_i = \Sigma_{j=1}^{i} (\frac{1}{3})^j + (\frac{1}{3})^i$. Let us prove by recursion on i that one can derive $c \triangleleft \beta_i$ for any $i \geq 0$.

The property holds for $i = 0$. Let us assume it holds for i. By applying as before axiom (16) to the last line of the derivation and by using the recursion hypothesis we can derive $c \triangleleft 1/3 + 1/3\beta_i = \beta_{i+1}$. So by recursion we conclude that for any $i \geq 0$ we can derive $c \triangleleft \beta_i$.

As moreover the sequence (β_i) converges to $\Sigma_{i=1}^{+\infty} (\frac{1}{3})^i = \frac{1}{2}$ we can deduce meta-theoretically that $c \triangleleft \frac{1}{2}$ (although we cannot derive this limit bound within our system).

However we can verify that one cannot derive in aHL the property $c \triangleleft \beta_2$, that is to say $c \triangleleft \frac{5}{9}$. Indeed the aHL judgement corresponding to $c \triangleleft \frac{5}{9}$ is $\vdash_{5/9} (x \xleftarrow{\$} d) \cdot (x \xleftarrow{\$} d)^{([x=0])} : T \Rightarrow [x \neq 1]$. However in order to be able to apply a $(While)$ rule in aHL one needs to have an integer variable b_v that strictly decreases at each execution of the body of the loop, which is not the case here. So one cannot apply any $(While)$ rule, and thus one cannot prove this bound.

This example thus shows that aGKAT is more expressive than aHL, in the sense that it can prove probability bounds that aHL cannot.

Kleene Algebra with Commutativity Conditions Is Undecidable

Arthur Azevedo de Amorim
Rochester Institute of Technology, NY, USA

Cheng Zhang[1]
University College London, UK

Marco Gaboardi
Boston University, MA, USA

Abstract

We prove that the equational theory of Kleene algebra with commutativity conditions on primitives (or atomic terms) is undecidable, thereby settling a longstanding open question in the theory of Kleene algebra. While this question has also been recently solved independently by Kuznetsov, our results hold even for weaker theories that do not support the *induction axioms* of Kleene algebra.

2012 ACM Subject Classification Theory of computation → Automated reasoning; Theory of computation → Regular languages

Keywords and phrases Kleene Algebra, Hypotheses, Complexity

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.36

Funding *Arthur Azevedo de Amorim*: National Science Foundation Grant No. 2314323.
Cheng Zhang: National Science Foundation Award No. 1845803 and No. 2040249.
Marco Gaboardi: National Science Foundation Award No. 1845803 and No. 2040249.

Acknowledgements We want to thank Todd Schmid and Alexandra Silva for the valuable discussion during this work; and also acknowledge the useful feedback provided by the anonymous reviewers of CSL2025. Finally, we want to thank Stepan Kuznetsov for his detailed and valuable feedback on this work.

1 Introduction

Kleene algebra generalizes the algebra of regular languages while retaining many of its pleasant properties, such as having a decidable equational theory. This enables numerous applications in program verification, by translating programs and specifications into Kleene-algebra terms and then checking these terms for equality. This idea has proved fruitful in many domains, including networked systems [1, 7], concurrency [9, 11, 12], probabilistic programming [18, 19], relational verification [4], program schematology [2], and program incorrectness [21].

Many applications require extending Kleene algebra with other axioms. A popular extension is adding commutativity conditions $e_1 e_2 = e_2 e_1$, which state that e_1 and e_2 can be composed in any order. In terms of program analysis, e_1 and e_2 correspond to commands of a larger program, and commutativity ensures that their order does not affect the final output. Such properties have been proven useful for relational reasoning [4] and concurrency [6].

Unfortunately, such extensions can pose issues for decidability. In particular, even the addition of equations of the form $xy = yx$, where x and y are primitives, can make the equivalence of two *regular languages* given by Kleene algebra terms [14, 8, 5] undecidable – in fact, Π^0_1-complete [15], or equivalent to the complement of the halting problem.

[1] Work performed at Boston Unviersity

Despite this negative result, it was still unknown whether we could decide such equations in *arbitrary* Kleene algebras – or, equivalently, decide whether an equation can be derived solely from the Kleene-algebra axioms. Indeed, since the set of Kleene-algebra equations is generated by finitely many clauses, it is recursively enumerable, or Σ_1^0. Since a set cannot be simultaneously Σ_1^0 and Π_1^0-complete, the problem of deciding equations under commutativity conditions for all regular languages is not the same as the problem of deciding such equations for all Kleene algebras. There must be equations that are valid for all regular languages, but not for arbitrary Kleene algebras. Nevertheless, the question of decidability of Kleene-algebra equations with commutativity conditions remained open for almost 30 years [15].

This paper settles this question *negatively*, proving that this problem is undecidable. In fact, undecidability holds even for weaker notions of Kleene algebra that do not validate its *induction axioms*, which are needed to prove many identities involving the iteration operation. At a high level, our proof works as follows. Given a machine M and an input x, we define an inequality between Kleene algebra terms with the following two properties: (1) if M halts on x and accepts, the inequality holds, but (2) if M halts on x and rejects, the inequality does not hold. If such inequalities were decidable, we would be able to computationally distinguish these two scenarios, which is impossible by diagonalization.

On Kuznetsov's Undecidability Proof

As we were finishing this paper, we learned that the question of undecidability had also been settled independently by Kuznetsov in recent work [17]. Though our techniques overlap, there are two noteworthy differences between the two proofs. On the one hand, Kuznetsov's proof uses the induction axioms of Kleene algebra, so it applies to fewer settings. On the other hand, Kuznetsov was able to prove that the equational theory of Kleene algebra with commutativity conditions is, in fact Σ_1^0-complete, by leveraging *effective inseparability*, a standard notion of computability theory. After learning about Kuznetsov's work, we could adapt his argument to derive completeness in our more general setting as well, so this paper can be seen as a synthesis of Kuznetsov's work and our own.

Structure of the paper

In Section 2, we recall basic facts about Kleene algebra, and introduce a framework for stating the problem of equations modulo commutativity conditions using category theory.

In Section 3, we present the core of our undecidability proof. We use algebra terms to model the transition relation of an abstract machine, and construct a set of inequalities that allows us to tell whether a machine accepts a given input or not. If we could decide such inequalities, we would be able to distinguish two effectively inseparable sets, which would lead to a contradiction. This argument hinges on a *completeness result* (Theorem 16), which guarantees that, if a certain machine accepts an input, then a corresponding inequality holds.

In Section 4, we prove that an analog of the completeness result holds for a large class of relations that can be represented with terms, provided that they satisfy a technical condition that allows us to reason about the image of a set by a relation.

In Section 5, we develop techniques to prove that the machine transition relation satisfies the required technical conditions for completeness. In Section 5.1, we show how we can view Kleene algebra terms as automata, proving an *expansion lemma* (Lemma 27) that guarantees that most terms can be expanded so that all of its matched strings bounded by some maximum length can be identified. This framework generalizes the usual definitions of derivative on Kleene algebra terms, but does not rely on the induction axioms of Kleene algebra. In Section 5.2, we show how we can refine the expansion lemma when terms have

bounded-output, which, roughly speaking, means such terms represent relations that map a string to only finitely many next strings. We prove that the transition relation satisfies these technical conditions (Section 5.3), which concludes the undecidability proof.

We conclude in Section 6, providing a detailed comparison between our work and the independent work of Kuznetsov [17].

2 Kleene Algebra and Commutable Sets

To set the stage for our result, we recall some basic facts about Kleene algebra and establish some common notation that we will use throughout the paper. We also introduce a notion of *commutable set*, which we will use to define algebras with commutativity conditions.

A (left-biased) *pre-Kleene algebra* is an idempotent semiring X equipped with a star operation. Spelled out explicitly, this means that X has operations

$$0 : X \qquad 1 : X$$
$$(-) + (-) : X \times X \to X \qquad (-) \cdot (-) : X \times X \to X \qquad (-)^* : X \to X,$$

which are required to satisfy the following equations:

$$1 \cdot x = x \cdot 1 = x \qquad 0 \cdot x = x \cdot 0 = 0 \qquad x \cdot (y \cdot z) = (x \cdot y) \cdot z$$
$$0 + x = x \qquad x + y = y + x \qquad x + (y + z) = (x + y) + z$$
$$x \cdot (y + z) = x \cdot y + x \cdot z \qquad (x + y) \cdot z = x \cdot z + y \cdot z \qquad x^* = 1 + x \cdot x^*,$$

where the last rule $x^* = 1 + x \cdot x^*$ is named "left unfolding". A pre-Kleene algebra carries the usual ordering relation on idempotent monoids: $x \leq y$ means that $y + x = x$. A *Kleene algebra* is a pre-Kleene algebra that satisfies the following properties:

$$xy \leq y \Rightarrow x^* y \leq y \qquad xy \leq x \Rightarrow xy^* \leq x,$$

dubbed left and right induction. A $*$*-continuous Kleene algebra* is a pre-Kleene algebra where, for all p, q and r, $\sup_{n \geq 0} pq^n r$ exists and is equal to $pq^* r$. Every $*$-continuous algebra satisfies the induction axioms, so it is, in fact, a Kleene algebra.

▶ **Example 1.** Though many pre-Kleene algebras that we'll consider are actually proper Kleene algebras, the two theories do not coincide. The following algebra, adapted from Kozen [13], validates all the pre-Kleene algebra axioms, but not the induction ones. The carrier set of the algebra is $\mathbb{N} + \{\bot, \top\}$, ordered by posing $\bot \leq n \leq \top$ for all $n \in \mathbb{N}$. The addition operation computes the maximum of two elements. Multiplication is defined as:

$$x \cdot \bot \triangleq \bot \cdot x \triangleq \bot \qquad x \cdot \top \triangleq \top \cdot x \triangleq \top \text{ when } x \neq \bot \qquad x \cdot y \triangleq x +_{\mathbb{N}} y$$

where $+_{\mathbb{N}}$ is the usual addition operation on natural numbers. The neutral elements of addition and multiplication are respectively $\bot$ and 0. The star operation is defined as follows:

$$x^* \triangleq \begin{cases} 0 & \text{if } x = \bot \\ \top & \text{otherwise} \end{cases}$$

We can verify the unfolding rule by case analysis:

when $x = \bot$: $\bot^* = 0 = \max(0, \bot) = \max(0, \bot \cdot 0) = \max(0, \bot \cdot \bot^*)$;

when $x \neq \bot$: $x^* = \top = \max(0, \top) = \max(0, x \cdot \top) = \max(0, x \cdot x^*)$.

However, this algebra is not a proper Kleene algebra, because it doesn't satisfy $(x^*)^* = x^*$ (which must hold in any Kleene algebra). Indeed, $(\bot^*)^* = 0^* = \top \neq 0 = \bot^*$.

▶ Remark 2. Weak Kleene algebras [16] are algebraic structures that sit between proper Kleene algebras and pre-Kleene algebras. They need not validate the induction axioms of Kleene algebra, but satisfy more rules than just left unfolding – in particular, $(x^*)^* = x^*$. Thus, Example 1 also shows that the theory of pre-Kleene algebras is strictly weaker than that of weak Kleene algebras.

Let X and Y be pre-Kleene algebras. A *morphism* of type $X \to Y$ is a function $f : X \to Y$ that commutes with all the operations. This gives rise to a series of categories $\mathsf{KA}^* \subset \mathsf{KA} \subset \mathsf{preKA}$ of $*$-continuous algebras, Kleene algebras, and pre-Kleene algebras. Each category is a strict full subcategory of the next one – strict because some pre-Kleene algebras are not Kleene algebras (Example 1) and because some Kleene algebras are not $*$-continuous [13].

The prototypical example of Kleene algebra is given by the set $\mathcal{L}X$ of regular languages over some alphabet X. In program analysis applications, a regular language describes the possible traces of events performed by some system. We use the multiplication operation to represent the sequential composition of two systems: if two components produce traces t_1 and t_2, then their sequential composition produces the concatenated trace $t_1 t_2$, indicating that the actions of the first component happen first. Thus, by checking if two regular languages are equal, we can assert that the behaviors of two programs coincide. When X is empty, $\mathcal{L}X$ is isomorphic to the booleans $\mathbb{2} \triangleq \{0 \leq 1\}$. The addition operation is disjunction, the multiplication operation is conjunction, and the star operation always outputs 1. This Kleene algebra is the initial object in all three categories preKA, KA and KA^*.

The induction property of Kleene algebra allows us to derive several useful properties for terms involving the star operation. For example, they imply that the star operation is monotonic, a *right-unfolding rule* $x^* = 1 + x^* x$, and also that $x^* x^* = x^*$. This means that many of intuitions about regular languages carry over to Kleene algebra. Unfortunately, when working with pre-Kleene algebras, most of these results cannot be directly applied, making reasoning trickier. In practice, we can only reason about properties of the star operation that involve a finite number of uses of the left-unfolding rule. Dealing with this limitation is at the heart of the challenges we will face when proving our undecidability result.

2.1 Commuting conditions

Sometimes, we would like to reason about a system where two actions can be reordered without affecting its behavior. For example, we might want to say that a program can perform assignments to separate variables in any order, or that actions of separate threads can be executed concurrently. To model this, we can work with algebra terms where some elements can be composed in any order. As we will see, unfortunately, adding such hypotheses indiscriminately can lead to algebras with an undecidable equational theory. The notion of *commutable set*, which we introduce next, allows us to discuss such hypotheses in generality.

▶ **Definition 3.** *A* commuting relation *on a set X is a reflexive symmetric relation. A* commutable set *is a carrier set endowed with a commuting relation $\sim$. We say that two elements x and y commute if $x \sim y$. A commutable set is* commutative *if all elements commute; it is* discrete *if the commuting relation is equality. A morphism of commutable sets is a function that preserves the commuting relation, which leads to a category* Comm*. A* commutable subset *of a commutable set X is a commutable set Y whose carrier is a subset of X, and whose commuting relation is the restriction of $\sim$ to Y. We'll often abuse notation and treat a subobject $Y \hookrightarrow X$ as a commutable subset if its image in X is a commutable subset.*

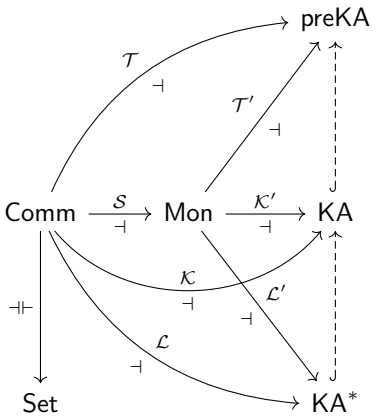
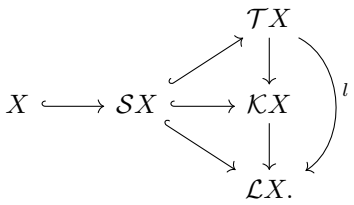


Figure 1 Algebraic constructions on commutable sets.

Given a commutable set, we have various ways of building algebraic structures, which can be summarized in the diagram of Figure 1 (which is commutative, except for the dashed arrows). The right-pointing arrows, marked with a $\dashv$, denote free constructions, in the sense that they have right adjoints that forget structure. The first construction, $\mathcal{S}$, is a functor from Comm to the category Mon of monoids and monoid morphisms. It maps a commutable set X to the monoid $\mathcal{S}X$ of strings over X, where we equate two strings if they can be obtained from each other by swapping adjacent elements that commute in X. The monoid operation is string concatenation, and the neutral element is the empty string. The corresponding right adjoint views a monoid Y as a commutable set where $x \sim y$ if and only if $xy = yx$.

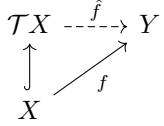
Another group of constructions extends a monoid X with the other Kleene algebra operations, and quotient the resulting terms by the equations we desire. For example, the elements of $\mathcal{T}'X$ are terms formed with Kleene algebra operations, where we identify the monoid operation with the multiplication operation of the pre-Kleene algebra, and where we identify two terms if they can be obtained from each other by applying the pre-Kleene algebra equations. The construction $\mathcal{K}'$ is obtained by imposing further equations on terms, while $\mathcal{L}'$ is given by the algebra of regular languages over a monoid [15], which we'll define soon. The right adjoints of these constructions view an algebra as a multiplicative monoid. By composing these constructions with $\mathcal{S}$, we obtain free constructions $\mathcal{T}$, $\mathcal{K}$ and $\mathcal{L}$ that turn any commutable set into some Kleene-algebra-like structure.

Being a free construction means, in particular, that we can embed the elements of a commutable set X into $\mathcal{S}X$, $\mathcal{T}X$, $\mathcal{K}X$ and $\mathcal{L}X$, as depicted in this commutative diagram:



By abuse of notation, we'll usually treat X as a proper subset of the free algebras. The vertical arrows take the elements of some algebra and impose the additional identities required by a stronger algebra. The composite l computes the *language interpretation* of a term, and will play an important role in our development, as we will see.

Free constructions, like $\mathcal{T}$, also allow us to define a morphism out of an algebra $\mathcal{T}X$ simply by specifying how the morphism acts on X. In other words, if $f : X \to Y$ is a morphism mapping a commutable set X to a pre-Kleene algebra Y, there exists a unique algebra morphism $\hat{f} : \mathcal{T}X \to Y$ such that the following diagram commutes:



Since f and $\hat{f}$ correspond uniquely to each other, we will not bother distinguishing between the two. We'll employ similar conventions for other left adjoints such as $\mathcal{S}$ or $\mathcal{L}$.

The last construction on Figure 1 allows us to turn any commutable set into a plain set by forgetting its commuting relation. This construction has both a left and a right adjoint: the right adjoint views a set as a commutative commutable set, by endowing it with the total relation; the left adjoint views a set as a discrete commutative set, by endowing it with the equality relation. By turning a set into a commutable set, discrete or commutative, and then building an algebra on top of that commutable set, we are able to express the usual notions of free algebra over a set, or of a free algebra where all symbols are allowed to commute.

▶ Remark 4 (Embedding algebras). We introduce some notation for embedding algebras into larger ones. Suppose that X is a commutable set and $Y \subseteq X$ is a commutable subset. By functoriality, this inclusion gives rise to morphisms of algebras of types $\mathcal{S}Y \to \mathcal{S}X$, $\mathcal{T}Y \to \mathcal{T}X$, etc. These morphisms are all injective, because they can be inverted: we can define a projection π_Y that maps $x \in X$ to itself, if $x \in Y$, or to 1, if $x \notin Y$. This definition is valid because, since Y inherits the commuting relation from X, and since 1 commutes with everything in $\mathcal{S}Y$, $\mathcal{T}Y$, etc., we can check that the commuting relation in X is preserved.

2.2 Regular Languages

If X is a monoid, we can view its power set $\mathcal{P}X$ as a $*$-continuous algebra equipped with the following operations:

$$0 \triangleq \emptyset \qquad 1 \triangleq \{1\}$$
$$A + B \triangleq A \cup B \qquad A \cdot B \triangleq \{xy \mid x \in A, y \in B\} \qquad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n.$$

The $*$-continuous algebra $\mathcal{L}'X$ of regular languages over X is the smallest subalgebra of $\mathcal{P}X$ that contains the singletons. The language interpretation $l : \mathcal{T}X \to \mathcal{L}X$ is the morphism that maps a symbol $x \in X$ to the singleton set $\{x\}$. This allows us to view a term as a set of strings over X, and we will often do this to simplify the notation; for example, if e is a term, we'll write $X \subseteq e$ to mean $X \subseteq l(e)$. Indeed, as the next few results show, it is often safe for us to view a term as a set of strings.

▶ **Theorem 5.** *If $s \in \mathcal{S}X$ is a string and $e \in \mathcal{T}X$ a term, then $s \leq e$ is equivalent to $s \in l(e)$.*

▶ **Theorem 6.** *We say that $e \in \mathcal{T}X$ is* finite *if its language $l(e)$ is. In this case, then $e = \sum l(e)$.*

▶ **Corollary 7.** *The language interpretation l is injective on finite terms: if $l(e_1) = l(e_2)$ and both e_1 and e_2 are finite, then $e_1 = e_2$.*

These results allow us to unambiguously view a finite set of strings over X as a finite term over X. We'll extend this convention to other sets: Y is a (pre-)Kleene algebra, we are going to view a finite set of elements $A \subseteq Y$ as the element $\sum_{a \in A} a \in Y$.

▶ **Corollary 8.** *For every term $e \neq 0$, there exists some string s such that $s \leq e$.*

One useful property of Kleene algebra is that, if X is finite, then $X^* \in \mathcal{K}X$ is the top element of the algebra. This result is generally not valid for $\mathcal{T}X$, but the following property will be good enough for our purposes.

▶ **Theorem 9.** *If X is finite and $e \in \mathcal{T}X$ is finite, then $eX^* \leq X^*$.*

To conclude our analogy between languages and terms, as far as $*$-continuous algebras are concerned, elements of $\mathcal{T}X$ are just as good as their corresponding languages – if Y is $*$-continuous, then every morphism of algebras $f : \mathcal{T}X \to Y$ can be factored through the language interpretation l:

$$\begin{array}{ccc} X & \hookrightarrow & \mathcal{L}X \\ \downarrow & \overset{l}{\nearrow} & \downarrow \\ \mathcal{T}X & \xrightarrow[f]{} & Y. \end{array}$$

This has some pleasant consequences. For example, let $[-]_0 : \mathcal{T}X \to \mathbb{2}$ be the morphism that maps every $x \in X$ to 0. Then $[e]_0 = 1$ if and only if $1 \leq e$. Indeed, this morphism must factor through $\mathcal{L}X$. The corresponding factoring $\mathcal{L}X$ must map any nonempty string to 0 and the empty string to 1. Thus, $[e]_0 = 1$ if and only if $1 \in l(e)$, which is equivalent to $1 \leq e$.

3 Undecidability via Effective Inseparability

Our undecidability result works by using pre-Kleene algebra equations to encode the execution of *two-counter machines*. Roughly speaking, a two-counter machine M is an automaton that has a control state and two counters. The machine can increment each counter, test if their values are zero, and halt. Two-counter and Turing machines are equivalent in expressive power: any two-counter machine can simulate the execution of a Turing machine, and vice versa; see Hopcroft et al. [10, §8.5.3,§8.5.4] for an idea of how this simulation works. In particular, given a Turing machine M, there exists a two-counter machine that halts on every input where M halts, and yields the same output for that input. For this reason, we'll tacitly use two-counter machines to implement computable functions in what follows.

▶ **Definition 10.** *A* two-counter *machine is a tuple $M = (Q_M, \dot{q}, \iota)$, where Q_M is a finite set of* control states*, $\dot{q} \in Q_M$ is an initial state, and $\iota : Q_M \to I_M$ is a transition function. The set I_M is the set of* instructions *of the machine, defined by the grammar*

$$I_M \ni i := \mathsf{Inc}(r, q) \mid \mathsf{If}(r, q, q) \mid \mathsf{Halt}(x) \qquad (r \in \{1, 2\}, q \in Q_M, x \in \{0, 1\}).$$

Two-counter machines act on configurations, which are strings of the form $a^n b^m q$, where q is a control state and a and b are counter symbols: the number of symbol occurrences determines which number is stored in a counter. When the machine halts, it outputs either 1 or 0 to indicate whether its input was accepted or rejected.

▶ **Definition 11.** *Let M be a two-counter machine. We define the following discrete commutable sets and terms:*

$$\begin{aligned} \Sigma_M &\triangleq Q_M + \{a, b, c_0, c_1\} && \textit{symbols} \\ \mathcal{T}\Sigma_M \ni C_M &\triangleq a^* b^* Q_M && \textit{running configurations} \\ \mathcal{T}\Sigma_M \ni T_M &\triangleq C_M + \{c_0, c_1\} && \textit{all configurations.} \end{aligned}$$

Normally, we would define the semantics of a two-counter machine directly, as a relation on configurations. However, it'll be more convenient to instead define the semantics through algebra terms that describe the graph of this relation, since we'll use these terms to analyze the execution of a machine with equations in pre-Kleene algebra. Our definition relies on the following construction.

▶ **Definition 12.** *Let X and Y be commutable sets. We define a commutable set*

$$X \oplus Y \triangleq \{x_l \mid x \in X\} \uplus \{y_r \mid y \in Y\},$$

where the commuting relation on $X \oplus Y$ is generated by the following rules:

$$\frac{}{x_l \sim y_r} \qquad \frac{x \sim x'}{x_l \sim x'_l} \qquad \frac{y \sim y'}{y_r \sim y'_r}$$

The canonical injections $(-)_l : X \to X \oplus Y$ and $(-)_r : Y \to X \oplus Y$ are morphisms in Comm *(and present commutable subsets). We abbreviate $X \oplus X$ as $\ddot{X}$.*

If X and Y are commutable sets, we abuse notation and view the functions $(-)_l : X \to X \oplus Y$ and $(-)_r : Y \to X \oplus Y$ as having types $\mathcal{T}X \to \mathcal{T}(X \oplus Y)$ and $\mathcal{T}Y \to \mathcal{T}(X \oplus Y)$. We have the corresponding projection functions $\pi_l : \mathcal{T}(X \oplus Y) \to \mathcal{T}X$ and $\pi_r : \mathcal{T}(X \oplus Y) \to \mathcal{T}Y$, where $\pi_l(y_r) = 1$ for $y \in Y$, and similarly for π_r (cf. Remark 4). If X is a commutable set, view a term $e \in \mathcal{T}X$ as an element $\mathcal{T}\ddot{X}$ by mapping each symbol $x \in X$ in e to $x_l x_r$. We'll use a similar convention for strings $\mathcal{S}$.

The idea behind this construction is that any string over $X \oplus Y$ can be seen as a pair of strings over X and Y. More precisely, the monoids $\mathcal{S}(X \oplus Y)$ and $\mathcal{S}X \times \mathcal{S}Y$ are isomorphic via the mappings

$$\mathcal{S}(X \oplus Y) \ni s \mapsto (\pi_l(s), \pi_r(s)) \qquad \mathcal{S}X \times \mathcal{S}Y \ni (s_1, s_2) \mapsto (s_1)_l (s_2)_r.$$

Since a term e over $X \oplus Y$ can be seen as a set of strings over $X \oplus Y$, we can also view it as a set of pairs of strings over X and Y – in other words, as a relation from $\mathcal{S}X$ to $\mathcal{S}Y$. We write $s \to_e s'$ if two strings are related in this way; that is, if $s_l s'_r \leq e$.

▶ **Definition 13** (Running a two-counter machine)**.** *We interpret each instruction $i \in I_M$ as an element $[\![i]\!] \in \mathcal{T}\ddot{\Sigma}_M$:*

$$\begin{aligned} [\![\mathsf{Inc}(1,q)]\!] &\triangleq a_r a^* b^* q_r & [\![\mathsf{If}(1,q_1,q_2)]\!] &\triangleq b^*(q_1)_r + a_l a^* b^* (q_2)_r \\ [\![\mathsf{Inc}(2,q)]\!] &\triangleq a^* b_r b^* q_r & [\![\mathsf{If}(2,q_1,q_2)]\!] &\triangleq a^*(q_1)_r + a^* b_l b^* (q_2)_r \\ [\![\mathsf{Halt}(x)]\!] &\triangleq (c_x)_r. \end{aligned}$$

The transition relation *of M, $R_M \in \mathcal{T}\ddot{\Sigma}_M$, is defined as*

$$R_M \triangleq \sum \{[\![\iota(q)]\!] q_l \mid q \in Q_M\}.$$

We say that M halts *on n if $a^n b^0 \dot{q} \to_{R_M}^* c_x$ for some $x \in \{0, 1\}$. We refer to x as the* output *of M on n.*

▶ **Lemma 14.** *The relation R_M satisfies the following property: for every $s \to_{R_M} s'$, s is of the form $a^n b^m q \leq C_M$. Moreover, for any s of this form, we have $s' = [\![\iota(q)]\!]_f(n,m)$, where the function $[\![i]\!]_f : \mathbb{N} \times \mathbb{N} \to T_M$ is defined as follows:*

$$[\![\mathsf{Inc}(1,q)]\!]_f(n,m) \triangleq a^{n+1}b^m q \qquad [\![\mathsf{If}(1,q_1,q_2)]\!]_f(n,m) \triangleq \begin{cases} a^n b^m q_1 & \text{if } n = 0 \\ a^p b^m q_2 & \text{if } n = p+1 \end{cases}$$

$$[\![\mathsf{Inc}(2,q)]\!]_f(n,m) \triangleq a^n b^{m+1} q \qquad [\![\mathsf{If}(2,q_1,q_2)]\!]_f(n,m) \triangleq \begin{cases} a^n b^m q_1 & \text{if } m = 0 \\ a^n b^p q_2 & \text{if } m = p+1 \end{cases}$$

$$[\![\mathsf{Halt}(x)]\!]_f(n,m) \triangleq c_x.$$

In particular, R_M defines a (partial) functional relation on T_M.

This means that Definition 13 accurately describes the standard semantics of two-counter machines [10], which allows us to analyze their properties algebraically. Combining this encoding with Theorem 17, we can show that KA inequalities over $\ddot{\Sigma}_M$ cannot be decided. More precisely, in the remainder of the paper, our aim is to prove the following results:

▶ **Theorem 15** (Soundness). *Given a two-counter machine M and a configuration $s \leq T_M$, suppose that the following inequality holds in $\mathcal{L}\ddot{\Sigma}_M$:*

$$s_r R_M^* \leq \Sigma^*(C_M + c_1)_r + \Sigma_M^* \Sigma_M^{\neq} \ddot{\Sigma}_M^*,$$

where $\Sigma_M^{\neq} \triangleq \sum_{\substack{x,y \in \Sigma \\ x \neq y}} x_l y_r$. If $s \to_{R_M}^* c_x$, then $x = 1$.

▶ **Theorem 16** (Completeness). *Given a two-counter machine M and some configuration $s \leq T_M$, we can compute a term ρ with the following property. If $s \to_{R_M}^* c_1$, then the following inequality is valid in* pre-*Kleene algebra: $sR_M^* \leq \Sigma_M^*(C_M + c_1)_r + \Sigma_M^* \Sigma_M^{\neq} \rho$.*

The soundness theorem can be shown by establishing a correspondence between traces of two-counter machines and the languages, then arguing that the language inequality implies that M halts and outputs 1. However, an inequality between terms is always stronger than the same inequality on languages. Thus, for completeness, we need to establish a stronger inequality between terms.

To obtain undecidability from soundness and completeness, we leverage *effective inseparability*, a notion from computability theory. In what follows, we use the notation $\langle x \rangle$ to refer to some effective encoding of the object x as a natural number.[2]

▶ **Theorem 17.** *The following two languages are effectively inseparable:*

$$A \triangleq \{\langle M, x\rangle \mid \textit{The two-counter machine } M \textit{ halts on } x \textit{ and outputs } 1\}$$

$$B \triangleq \{\langle M, x\rangle \mid \textit{The two-counter machine } M \textit{ halts on } x \textit{ and outputs } 0\}.$$

In other words, there is a partial computable function f with the following property. Given a machine M, let W_M be the set of inputs accepted by M. Suppose that M_1 and M_0 are such that $W_{M_1} \cap W_{M_0} = \emptyset$, $A \subseteq W_{M_1}$ and $B \subseteq W_{M_0}$. Then $f\langle M_1, M_0\rangle$ is defined and does not belong to $W_{M_1} \cup W_{M_0}$.

[2] Note that we do not assume that this encoding is a functional relation. For example, we will need to encode pre-Kleene algebra terms as numbers. Such a term is an equivalence class of syntax trees quotiented by provable equality. Thus, each term can be encoded as multiple natural numbers, one for each syntax tree in its equivalence class. Nevertheless, by abuse of notation, we'll use the encoding notation as if it denoted a unique number.

Effective inseparability is a strengthening of the notion of inseparability, which says that two sets cannot be distinguished by a total computable function. If we are just interested in the undecidability of the equational theory, then basic inseparability is enough, as the following argument shows:

▶ **Theorem 18** (Undecidability). *Let $\Sigma \triangleq \{0,1\}$ be a discrete commutable set. Suppose that we have a diagram of sets*

$$\begin{array}{ccc} \mathcal{T}\ddot{\Sigma} & \xrightarrow{\ l\ } & \mathcal{L}\ddot{\Sigma} \\ & \searrow^{l'} & \uparrow \\ & & X, \end{array}$$

where l' is computable. Then equality on X is undecidable. In particular, equality is undecidable on $\mathcal{T}\ddot{\Sigma}$, $\mathcal{K}\ddot{\Sigma}$ and $\mathcal{L}\ddot{\Sigma}$.

Proof. Let A and B be the sets of Theorem 17. Let's define a computable function $\eta : \Sigma^* \to \Sigma^*$ with the following properties:

- if $s \in A$, then $\eta(s) \in X_=$, where $X_= \triangleq \{\langle x, y\rangle \mid x \text{ and } y \text{ encode the same element of } X\}$.
- if $s \in B$, then $\eta(s) \notin X_=$.

Find a suitable encoding of the characters of Σ_M as binary strings, which leads to the following injective embeddings:

$$\begin{array}{ccc} \mathcal{T}\ddot{\Sigma}_M & \xrightarrow{\ l\ } & \mathcal{L}\ddot{\Sigma}_M \\ \downarrow & & \downarrow \\ \mathcal{T}\ddot{\Sigma} & \xrightarrow{\ l\ } & \mathcal{L}\ddot{\Sigma} \\ & \searrow^{l'} & \uparrow \\ & & X. \end{array}$$

In what follows, we'll treat $\mathcal{T}\ddot{\Sigma}_M$ and $\mathcal{L}\ddot{\Sigma}_M$ as subsets of $\mathcal{T}\ddot{\Sigma}$ and $\mathcal{L}\ddot{\Sigma}$, to simplify the notation.

Suppose that we are given some string $s \in \Sigma^*$. We define $\eta(s)$ as follows. We can assume that s is of the form $\langle M, n\rangle$, where M is a machine and $n \in \mathbb{N}$ (if s is not of this form, we define the output as $\eta(s) = \langle l'(0), l'(1)\rangle$). First, we compute the term ρ of Theorem 16, using $a^n b^0 \dot{q}$ as the initial configuration. Next, let e_L and e_R be the left- and right-hand sides of the inequality of Theorem 16. We pose $\eta\langle M, n\rangle \triangleq \langle l'(e_L + e_R), l'(e_R)\rangle$.

If $\langle M, n\rangle \in A$, the inequality of Theorem 16 is valid. Thus, $e_L \leq e_R$ holds, or, equivalently, $e_L + e_R = e_R$. Thus, $l'(e_L + e_R) = l'(e_R)$ is valid, which implies that $\eta(s) \in X_=$.

If, on the other hand, M outputs 0 on n (that is, $\langle M, n\rangle \in B$), we claim that $\eta(s) \notin X_=$. It suffices to prove $l'(e_L + e_R) \neq l'(e_R)$. Aiming for a contradiction, suppose that $l'(e_L + e_R) = l'(e_R)$. This implies $l(e_L + e_R) = l(e_L) + l(e_R) = l(e_R)$, which is equivalent to the inequality $l(e_L) \leq l(e_R)$. Let $e'_R \in \mathcal{T}\ddot{\Sigma}_M$ be the right-hand side of the inequality of Theorem 15. We have $l(e_R) \leq l(e'_R)$ because $l(\rho) \leq l(\Sigma^*_M)$. Thus, $l(e_L) \leq l(e'_R)$. However, by Theorem 15, this can only hold if M outputs 1, which contradicts our assumption.

To conclude, suppose that $d : \Sigma^* \to \{0,1\}$ is a decider for $X_=$ (that is, suppose that equality on X is decidable). Then $d \circ \eta$ can separate the sets A and B, which contradicts Theorem 17 because two effectively inseparable sets are also computationally inseparable. Therefore, such a d cannot exist. ◀

However, if we also want a more precise characterization of the complexity of this theory, the notion of effective inseparability is crucial. The following argument, which refines the previous proof, is based on Kuznetsov's work [17].

▶ **Theorem 19** (Complexity). *If equalities in X (from Theorem 18) are recursive enumerable, then equalities in X are Σ_1^0-complete. In particular, equalities in $\mathcal{T}\ddot{\Sigma}$ and $\mathcal{K}\ddot{\Sigma}$ are Σ_1^0-complete.*

Proof. Let $A' \triangleq \{\langle M, n\rangle \mid l'(e_L + e_R) = l'(e_R)\}$, where (M, n) is a machine-input pair and e_L and e_R are defined as in the proof of Theorem 18. By arguments in the proof of Theorem 18, if $\langle M, n\rangle \in A$, then $l'(e_L + e_R) = l'(e_R)$, which means $A' \supseteq A$. By similar arguments, $A' \cap B = \emptyset$.

By folklore [17, Proposition 9], A' and B are effectively inseparable because A and B are. Moreover, note that both A' and B are in Σ_1^0 – membership in A' is recursively enumerable because we can compute e_L and e_R from (M, n) and enumerate the possible proofs of $l'(e_L + e_R) = l'(e_R)$. This implies that A' is Σ_1^0-complete [17, Proposition 7], and therefore Σ_1^0-hard.

The function η in the proof of Theorem 18 has the property that $\eta(s) \in X_=$ if and only if $s \in A'$. (This relies on the fact that we defined $\eta(s) = \langle l'(0), l'(1)\rangle$ when s is not the encoding of a machine-input pair, and that $l'(0) \neq l'(1)$ because $l(0) \neq l(1)$). In other words, η is a reduction from A' to $X_=$, which proves $X_=$ is Σ_1^0-hard. We conclude because we assumed that equality on X is in Σ_1^0. ◀

Thus, to establish undecidability, we need to prove soundness and completeness. The easiest part is proving soundness: we just need to adapt the proof of undecidability of equations of $*$-continuous Kleene algebras with commutativity conditions [14]. For completeness, however, we need to do some more work. Roughly speaking, we first prove that R_M satisfies an analogue of the completeness theorem for a single transition, and then show that this version implies a more general one for an arbitrary number of transitions (Section 4).

The main challenge for proving the single-step version of completeness is that we can no longer leverage properties of regular languages, and must reason solely using the laws of pre-Kleene algebra. Our strategy is to show that R_M is just as good as its corresponding regular language if we want to reason about *prefixes* of matched strings. Given any string $s' \leq R_M$ and a current state s, we can tell whether s' encodes a valid sequence of transitions or not simply by looking at some finite prefix determined by s. This finite prefix can be extracted by unfolding R_M finitely many times, which can be done in the setting of preKA.

4 Representing Relations

In this section, we show that we can reduce the statement of completeness to a similar statement about single transitions. If $e \in \mathcal{T}\ddot{\Sigma}$ and $\Lambda \subseteq \mathcal{S}\Sigma$ is a set of strings, we write $\mathsf{Next}_e(\Lambda)$ to denote the image of Λ by $\to_e$; that is, the set $\bigcup_{s\in\Lambda}\{s' \mid s \to_e s'\}$.

▶ **Definition 20.** *Let $L \in \mathcal{T}\Sigma$ be term. We say that a term $e \in \mathcal{T}\ddot{\Sigma}$ is a* representable relation *on L if the following conditions hold:*

- $\pi_l(e) \leq L$;
- $\pi_r(e) \leq L$;
- $\mathsf{Next}_e(\Lambda)$ *is finite if* Λ *is (note that we must have* $\mathsf{Next}_e(\Lambda) \leq \pi_r(e) \leq L$*);*
- *there exists some* residue term ρ *such that* $\Lambda_r e \leq \Lambda\, \mathsf{Next}_e(\Lambda)_r + \Sigma^*\Sigma^{\neq}\rho$ *for every finite* Λ.

We write $e : \mathsf{Rel}(L)$ to denote the type of e.

Given a representable relation e, we can iterate the above inequality several times when reasoning about its reflexive transitive closure e^*:

▶ **Lemma 21.** *Suppose that* $e : \mathsf{Rel}(L)$*. There exists some* ρ *such that, for every* $n \in \mathbb{N}$ *and every finite* $\Lambda \leq L$*, we have the inequality* $\Lambda_r e^* \leq \Sigma^* \mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \mathsf{Next}_e^n(\Lambda)_r e^* + \Sigma^* \Sigma^{\neq} \rho$*, where* $\mathsf{Next}_e^{<n} = \bigcup_{i<n} \mathsf{Next}_e^i(\Lambda)$*.*

If we know that the number of transitions from a given set of initial states is bounded, we obtain the following result.

▶ **Theorem 22.** *If* $e : \mathsf{Rel}(L)$*, there exists* ρ *such that, given* $n \in \mathbb{N}$ *and a finite* $\Lambda \leq L$*, if* $\mathsf{Next}_e^n(\Lambda) = \emptyset$*, then* $\Lambda_r e^* \leq \Sigma^* \mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \Sigma^{\neq} \rho$*.*

5 Proving Representability

In this section, we prove that the transition relation R_M of a two-counter machine is a representable relation, which will allow us to derive completeness from Theorem 22. To do this, we need to show how we can use finite unfoldings of a relation to pinpoint certain terms that definitely match the "error" term $\Sigma_M^* \Sigma_M^{\neq} \rho$.

5.1 Automata theory

One of the pleasant consequences of working with Kleene algebra is that many intuitions about regular languages carry over. In particular, we can analyze terms by characterizing them as automata. This can be done algebraically by posing certain *derivative operations* δ_x on terms, which satisfy a *fundamental theorem* [20]: given a term $e \in \mathcal{K}X$, we have $e = e_0 + \sum_{x \in X} x \cdot \delta_x(e)$, where $e_0 \in \{0, 1\}$. Intuitively, each term in this equation corresponds to a state of some automaton. The term e corresponds to the starting state of the automaton, the null term e_0 states whether the starting state is accepting, and each $\delta_x(e)$ the state we transition too after observing the character $x \in X$. Derivatives can be iterated, describing the behavior of the automaton as it reads larger and larger strings, and which of those strings are accepted by it. This would be useful for our purposes, because such iterated derivatives would allow us to compute all prefixes up to a given length that can match an expression. Unfortunately, this theory of derivatives hinges on the induction properties of Kleene algebra, and it is unlikely that it can be adapted in all generality to the preKA setting. For example, the closest we can get to an expansion for 1^* is $1^* = 1 + 1 \cdot 1^* = 1 + 1^*$, which does not have the required form. Indeed, as demonstrated by Example 1, the star operation no longer preserves the multiplicative identity in preKA.

To remedy this issue, we are going to carve out a set of so-called *finite-state terms* of a pre-Kleene algebra, for which this type of reasoning is sound. Luckily, most regular operations preserve finite-state terms; we just need to be a little bit careful with the star operation. We start by defining *derivable* terms, which can be derived at least once. Finite-state terms will then allow us to iterate derivatives.

▶ **Definition 23.** *Let* $e \in \mathcal{T}X$ *be a term, where* X *is finite. We say that* e *is* derivable *if there exists a family of terms* $\{\delta_x(e)\}_{x \in X}$ *such that* $e = [e]_0 + \sum_x x\delta_x(e)$*. Recall* $[-]_0$ *is the homomorphism* $[-]_0 : \mathcal{T}X \to \mathbb{2}$ *such that* $[e]_0 = 1 \iff e \geq 1$*. We refer to the term* $\delta_x(e)$ *as the* derivative with respect to x*.*

The family $\delta_x(e)$ is not necessarily unique. Nevertheless, we'll use the notation $\delta_x(e)$ to refer to specific derivatives of x when it is clear from the context which one we mean.

▶ **Lemma 24.** *Derivable terms are closed under all the pre-Kleene algebra operations, with the following caveats: for e^*, we also require that $[e]_0 = 0$; for $e_1 e_2$, the term is also derivable if e_2 isn't, provided that $[e_1]_0 = 0$. We have the following choices of derivatives:*

$$\begin{aligned} \delta_x(0) &= 0 & \delta_x(1) &= 0 \\ \delta_x(x) &= 1 & \delta_x(y) &= 0 \quad \text{if } y \neq x \\ \delta_x(e_1 + e_2) &= \delta_x(e_1) + \delta_x(e_2) & \delta_x(e_1 e_2) &= [e_1]_0 \delta_x(e_2) + \delta_x(e_1) e_2 \\ \delta_x(e^*) &= \delta_x(e) e^*, \end{aligned}$$

where, by abuse of notation, we treat $[e_1]_0 \delta_x(e_2)$ as 0 when e_2 is not necessarily derivable (since, by assumption, $[e_1]_0 = 0$ in that case).

▶ **Definition 25.** *Suppose that X is finite. A* finite-state automaton *is a finite set S of elements of $\mathcal{T}X$ (the* states*) that contains 1, is closed under finite sums and under derivatives (that is, every $e \in S$ is derivable, and each $\delta_x(e)$ is a state). We say that a term e is* finite state *if it is a state of some finite-state automaton S.*

Requiring that the states of an automaton be closed under sums means, roughly speaking, that we are working with non-deterministic rather than deterministic automata, generalizing the notion of Antimirov's derivative [3]. This treatment is convenient for the commutative setting, since a given string could be matched by choosing different orderings of its characters.

Finite-state terms can, in fact, be inductively constructed from the operations of pre-Kleene algebra, thus making the identification of a finite-state term trivial.

▶ **Lemma 26.** *Let X be a finite commutable set. Finite-state terms are preserved by all the pre-Kleene algebra operations (for e^*, we additionally require that $[e]_0 = 0$). Moreover, the set of states of the corresponding automata can be effectively computed.*

Furthermore, since terms in a finite-state automaton are closed under derivatives, we can unfold them via derivatives k times. This unfolding will turn a term into a sum of some strings that are shorter than k; and some strings s with length exact k, followed the residual expressions e_s indexed by s. Formally, we can express this property as follows.

▶ **Lemma 27.** *Let $e \in \mathcal{T}X$ be a state of a finite-state automaton S, and $k \in \mathbb{N}$. We can write*

$$e = \sum\{s \mid s \in \mathcal{S}X, s \leq e, |s| < k\} + \sum\{s e_s \mid s \in \mathcal{S}X, |s| = k\},$$

where each $e_s \in S$ for all s, and the size $|s| \in \mathbb{N}$ of a string s is defined by mapping every symbol of s to $1 \in \mathbb{N}$.

5.2 Bounded-Output Terms

Lemma 27 gives us almost what we need to prove that the transition term R_M is a representable relation. It allows us to partition R_M into strings s of length bounded by k and terms of the form se_s, which match strings prefixed by s of length greater than k. The first component, the strings s, can be easily shown to satisfy the upper bound required for being representable. However, the prefixes s that appear in the terms se_s are arbitrary, and, since we are working with pre-Kleene algebra, there isn't much we can leverage to show that such prefixes will yield a similar bound. The issue is that, in principle, in order to tell whether $s'_r s e_s \leq \Sigma_M^* \Sigma_M^{\neq} \rho$, we might need to unfold e_s arbitrarily deep, which we cannot do in the preKA setting. To rule out these issues, we introduce a notion of *bounded-output terms*, which guarantee that only a finite amount of unfolding is necessary.

▶ **Definition 28.** *Let $e \in \mathcal{T}\ddot{\Sigma}$ be a term. We say that e has* bounded output *if there exists some $k \in \mathbb{N}$ (the* fanout*) such that, for every string $s \leq e$, $|\pi_r(s)| \leq (|\pi_l(s)| + 1)k$.*

▶ **Lemma 29.** *Let e have bounded output with fanout k and let Λ be finite. If $s \in \mathsf{Next}_e(\Lambda)$, then $|s| \leq (m+1)k$, where $m = \max\{|s'| \mid s' \in \Lambda\}$. Thus, since Σ is finite, $\mathsf{Next}_e(\Lambda)$ is finite.*

▶ **Lemma 30.** *Bounded-output terms are closed under all the pre-Kleene algebra operations. For e^*, we additionally require that $|\pi_l(s)| \geq 1$ for all strings $s \leq e$.*

For bounded-output terms, we can improve the expansion of Lemma 27.

▶ **Lemma 31.** *Let $e \in \mathcal{T}\ddot{\Sigma}$ be a bounded-output term that is the state of some automaton S. There exists some $k \in \mathbb{N}$ such that e has fanout k and such that, for every $n \in \mathbb{N}$, we can write*

$$e = \sum\{s \mid s \leq e, |s| < n\} + \sum\{se_s \mid s \in \mathcal{S}\ddot{\Sigma}, |s| = n, |\pi_r(s)| \leq (|\pi_l(s)| + 1)k\},$$

where $e_s \in S$ for every s.

▶ **Definition 32.** *A term L over Σ is* prefix free *if for all strings $s_1 \leq L$ and $s_2 \leq L$, if s_1 is a prefix of s_2, then $s_1 = s_2$.*

▶ **Lemma 33** (normal)**.** *Let s and s' be two strings over Σ such that one is not a prefix of the other, or vice versa. Then we can write $s = s_0 x s_1$ and $s' = s_0 x' s_1'$ with $x \neq x'$. Thus, $s_r s_l' \ddot{\Sigma}^* \leq \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^*$.*

▶ **Lemma 34.** *Suppose that $e \in \mathcal{T}\ddot{\Sigma}$ is such that $\pi_l(e) \leq L$ and $\pi_r(e) \leq L$, where L is prefix free. Suppose, moreover, that e is finite-state and has bounded output. Then $e : \mathsf{Rel}(L)$.*

5.3 Putting Everything Together

To derive completeness for two-counter machines (Theorem 16), it suffices to show that the hypotheses of Lemma 34 are satisfied.

▶ **Lemma 35.** *We have the following properties:*
- *T_M is prefix free.*
- *$\pi_l(R_M) \leq C_M \leq T_M$.*
- *$\pi_r(R_M) \leq T_M$.*
- *R_M is finite state (Definition 25).*
- *R_M has bounded output (Definition 28).*

Thus, by Lemma 34, the term R_M is a representable relation of type $\mathsf{Rel}(T_M)$.

Proof. To show that R_M is finite state and had bounded output, we just appeal to the closure properties of such terms Lemmas 26 and 30. The rest is routine. ◀

We can finally conclude with the proof of completeness, thus establishing undecidability (Theorem 18).

Proof of Theorem 16. If $s = s_0 \to_{R_M} \cdots \to_{R_M} s_n = c_1$, we can show that $\mathsf{Next}_e^i(s)$ is $\{s_i\}$ for $i \leq n$ and $\emptyset$ when $i > n$, because the transition relation is deterministic and because c_1 does not transition. Moreover, by Lemma 35, we have $s_i \leq C_M$ for every $i < n$ (since $(s_i)_l(s_{i+1})_r \leq R_M$).

Choose ρ as in Theorem 22. We have

$$\begin{aligned} sR_M^* &\leq \Sigma^* \,\mathsf{Next}_e^{<n+1}(s)_r + \Sigma^*\Sigma^{\neq}\rho \\ &= \Sigma^*(\mathsf{Next}_e^{<n}(s) + \mathsf{Next}_e^{n}(s))_r + \Sigma^*\Sigma^{\neq}\rho \\ &\leq \Sigma^*(C_M + c_1)_r + \Sigma^*\Sigma^{\neq}\rho. \end{aligned}$$

◀

6 Conclusion and Related Work

In his seminal work, Kozen [15] established several hardness and completeness results for variants of Kleene algebra. He noted that deciding equality in $*$-continuous Kleene algebras with commutativity conditions on primitives was not possible – more precisely, the problem is Π_1^0-complete, by reduction from the complement of the Post correspondence problem (PCP). However, at the time, it was unknown whether a similar result applied to the pure theory of Kleene algebra with commutativity conditions ($\mathcal{K}X$). The question had been left open since then. Our work provides a solution, proving that the problem is undecidable, even for a much weaker theory $\mathcal{T}X$, which omits the induction axioms of Kleene algebra.

As we were about to post publicly this work, we became aware of the work of Kuznetsov [17], who independently proved a similar result. There are two main differences between our results and his. Originally, our proof only established the undecidability of the theory of Kleene algebra with commutativity conditions, whereas Kuznetsov's work proved its Σ_1^0-completeness as well by leveraging the notion of *effective inseparability*. Since learning about his work, we managed to adapt his ideas to our setting, thus obtaining completeness as well. On the other hand, Kuznetsov's proof requires the induction axiom of Kleene algebra to simplify some of the inequalities involving starred terms – specifically, he needs the identity $A^*(A^*)^+ \leq A^*$ and the monotonicity of $(-)^*$, whereas our proof also applies to the weaker theory of pre-Kleene algebra. In this sense, we can view the results reported here as a synthesis of Kuznetsov's work and ours.

In terms of techniques, both of our works draw inspiration from the proof of Π_1^0-completeness of the equational theory of $*$-continuous KA. Leveraging the reduction of the halting problem to the PCP, Kuznetsov used Kleene-algebra inequalities to describe *self-looping* Turing machines – that is, Turing machines that run forever by reaching a designated configuration that steps to itself. He then showed that the set of machine-input pairs $\langle M, x\rangle$ where machines M that reach a self-looping state on input x is recursively inseparable from the set of such pairs where M halts on the input, which implies that such inequalities cannot decidable.

The inequalities used by Kuznetsov are similar to ours, and can be proved by unfolding finitely many times the starred term that defines the execution of Turing machines, and by applying standard Kleene algebra inequalities that follow from induction. One important difference is that, in Kuznetsov's work, this starred term contains only $*$-free terms, which arise from the reduction of the halting problem to the PCP. This requires some more work to establish that the inequality indeed encodes the execution of the Turing machine, but this work just replicates the ideas behind the standard reduction from the halting problem to the PCP, so it does not need to be belabored. On the other hand, we leverage the language of Kleene algebra to define an execution model for two-counter machines, which can be encoded more easily. The downside of our approach is that our relation R_M involves starred terms, which require our notion of bounded output to be analyzed effectively.

References

1 Carolyn Jane Anderson, Nate Foster, Arjun Guha, Jean-Baptiste Jeannin, Dexter Kozen, Cole Schlesinger, and David Walker. Netkat: semantic foundations for networks. *ACM SIGPLAN Notices*, 49(1):113–126, January 2014. doi:10.1145/2578855.2535862.

2 Allegra Angus and Dexter Kozen. Kleene Algebra with Tests and Program Schematology. Technical Report, Cornell University, USA, June 2001.

3 Valentin Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *Theoretical Computer Science*, 155(2):291–319, March 1996. doi:10.1016/0304-3975(95)00182-4.

4 Timos Antonopoulos, Eric Koskinen, Ton Chanh Le, Ramana Nagasamudram, David A. Naumann, and Minh Ngo. An Algebra of Alignment for Relational Verification. *Proceedings of the ACM on Programming Languages*, 7(POPL):20:573–20:603, January 2023. doi:10.1145/3571213.

5 Jean Berstel. *Transductions and Context-Free Languages*. Vieweg+Teubner Verlag, Wiesbaden, 1979. doi:10.1007/978-3-663-09367-1.

6 Volker Diekert and Yves Métivier. Partial Commutation and Traces. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages: Volume 3 Beyond Words*, pages 457–533. Springer, Berlin, Heidelberg, 1997. doi:10.1007/978-3-642-59126-6_8.

7 Nate Foster, Dexter Kozen, Mae Milano, Alexandra Silva, and Laure Thompson. A coalgebraic decision procedure for netkat. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '15, pages 343–355, New York, NY, USA, January 2015. Association for Computing Machinery. doi:10.1145/2676726.2677011.

8 Alan Gibbons and Wojciech Rytter. On the decidability of some problems about rational subsets of free partially commutative monoids. *Theoretical Computer Science*, 48:329–337, January 1986. doi:10.1016/0304-3975(86)90101-5.

9 C. A. R. Tony Hoare, Bernhard Möller, Georg Struth, and Ian Wehrman. Concurrent Kleene Algebra. In Mario Bravetti and Gianluigi Zavattaro, editors, *CONCUR 2009 - Concurrency Theory*, pages 399–414, Berlin, Heidelberg, 2009. Springer. doi:10.1007/978-3-642-04081-8_27.

10 J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley series in computer science. Addison-Wesley, 2001. URL: https://books.google.com/books?id=omIPAQAAMAAJ.

11 Tobias Kappé, Paul Brunet, Alexandra Silva, Jana Wagemaker, and Fabio Zanasi. *Concurrent Kleene Algebra with Observations: From Hypotheses to Completeness*, volume 12077 of *Lecture Notes in Computer Science*, pages 381–400. Springer International Publishing, Cham, 2020. doi:10.1007/978-3-030-45231-5_20.

12 Tobias Kappé, Paul Brunet, Alexandra Silva, and Fabio Zanasi. Concurrent kleene algebra: Free model and completeness. In Amal Ahmed, editor, *Programming Languages and Systems*, Lecture Notes in Computer Science, pages 856–882, Cham, 2018. Springer International Publishing. doi:10.1007/978-3-319-89884-1_30.

13 Dexter Kozen. On kleene algebras and closed semirings. In Branislav Rovan, editor, *Mathematical Foundations of Computer Science 1990*, volume 452, pages 26–47. Springer-Verlag, Berlin/Heidelberg, 1990. doi:10.1007/BFb0029594.

14 Dexter Kozen. *Kleene algebra with tests and commutativity conditions*, volume 1055 of *Lecture Notes in Computer Science*, pages 14–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996. doi:10.1007/3-540-61042-1_35.

15 Dexter Kozen. On the complexity of reasoning in kleene algebra. In *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland, June 29 - July 2, 1997*, pages 195–202. IEEE Computer Society, 1997. doi:10.1109/LICS.1997.614947.

16 Dexter Kozen and Alexandra Silva. Left-handed completeness. *Theoretical Computer Science*, 807:220–233, February 2020. doi:10.1016/j.tcs.2019.10.040.

17 Stepan L. Kuznetsov. On the complexity of reasoning in kleene algebra with commutativity conditions. In Erika Ábrahám, Clemens Dubslaff, and Silvia Lizeth Tapia Tarifa, editors, *Theoretical Aspects of Computing - ICTAC 2023 - 20th International Colloquium, Lima, Peru, December 4-8, 2023, Proceedings*, volume 14446 of *Lecture Notes in Computer Science*, pages 83–99. Springer, 2023. doi:10.1007/978-3-031-47963-2_7.

18 A. K. McIver, E. Cohen, and C. C. Morgan. *Using Probabilistic Kleene Algebra for Protocol Verification*, volume 4136 of *Lecture Notes in Computer Science*, pages 296–310. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. doi:10.1007/11828563_20.

19 Annabelle McIver, Tahiry M. Rabehaja, and Georg Struth. On probabilistic kleene algebras, automata and simulations. In *Proceedings of the 12th International Conference on Relational and Algebraic Methods in Computer Science*, RAMICS'11, pages 264–279, Berlin, Heidelberg, May 2011. Springer-Verlag. doi:10.1007/978-3-642-21070-9_20.

20 A.M Silva. *Kleene coalgebra*. s.n.; UB Nijmegen host, S.l.; Nijmegen, 2010. URL: http://hdl.handle.net/2066/83205.

21 Cheng Zhang, Arthur Azevedo de Amorim, and Marco Gaboardi. On incorrectness logic and kleene algebra with top and tests. *arxiv preprint*, August 2022. arXiv:2108.07707, doi:10.48550/arXiv.2108.07707.

A Detailed Proofs

▶ **Theorem 5.** *If $s \in \mathcal{S}X$ is a string and $e \in \mathcal{T}X$ a term, then $s \leq e$ is equivalent to $s \in l(e)$.*

Proof of Theorem 5. Suppose that $s \leq e$. Then $s \in \{s\} = l_X(s) \subseteq l_X(e)$ by monotonicity. Conversely, suppose that $s \in l_X(e)$. We proceed by induction on e.

- If $e = x \in X$, then $s \in l_X(x)$ means that $s = x$. Thus, we get $s \leq e$.
- If $e = 0$, we get a contradiction.
- If $e = 1$, we must have $s = 1$, thus $s \leq e$.
- If $e = e_1 e_2$, we must have $s = s_1 s_2$, with $s_i \in l_X(e_i)$. By the induction hypotheses, $s_i \leq e_i$, and thus $s \leq e$.
- If $e = e_1 + e_2$, then there is some i such that $s \in l_X(e_i)$. By the induction hypothesis, $s \leq e_i$, and thus $s \leq e_1 + e_2$.
- Finally, suppose that $e = e_1^*$. Thus, there exists some n such that $s \in l_x(e_1)^n$. This means that we can find a family $(s_i)_{i \in \{1,\ldots,n\}}$ such that $s = \prod_i s_i$ and $s_i \in l_x(e_1)$ for every i. By the induction hypothesis, $s_i \leq e_1$ for every i. Therefore, $s = \prod_i s_i \leq e_1^n \leq e_1^* = e$. ◀

▶ **Theorem 6.** *We say that $e \in \mathcal{T}X$ is* finite *if its language $l(e)$ is. In this case, then $e = \sum l(e)$.*

Proof of Theorem 6. By induction on e. We note that, if $l(e)$ is finite, then $l(e')$ is also finite for every immediate subterm e', which allows us to apply the relevant induction hypotheses. If e is of the form $e_1 e_2$ and $l(e) = \emptyset$, this need not be the case, but at least one of the factors e_i satisfies $l(e_i) = \emptyset$, which is good enough. ◀

▶ **Corollary 7.** *The language interpretation l is injective on finite terms: if $l(e_1) = l(e_2)$ and both e_1 and e_2 are finite, then $e_1 = e_2$.*

Proof of Corollary 7. We have $e_1 = \sum l(e_1) = \sum l(e_2) = e_2$. ◀

▶ **Corollary 8.** *For every term $e \neq 0$, there exists some string s such that $s \leq e$.*

Proof of Corollary 8. Note that $l(e) \neq \emptyset$. Indeed, if $l(e) = \emptyset = l(0)$, then $e = 0$ by Corollary 7, which contradicts our hypothesis. Therefore, we can find some s such that $s \in l(e)$. But this is equivalent to $s \leq e$ by Theorem 5. ◀

▶ **Lemma 14.** *The relation R_M satisfies the following property: for every $s \to_{R_M} s'$, s is of the form $a^n b^m q \leq C_M$. Moreover, for any s of this form, we have $s' = [\![\iota(q)]\!]_f(n,m)$, where the function $[\![i]\!]_f : \mathbb{N} \times \mathbb{N} \to T_M$ is defined as follows:*

$$[\![\mathsf{Inc}(1,q)]\!]_f(n,m) \triangleq a^{n+1}b^m q \qquad [\![\mathsf{If}(1,q_1,q_2)]\!]_f(n,m) \triangleq \begin{cases} a^n b^m q_1 & \text{if } n = 0 \\ a^p b^m q_2 & \text{if } n = p+1 \end{cases}$$

$$[\![\mathsf{Inc}(2,q)]\!]_f(n,m) \triangleq a^n b^{m+1} q \qquad [\![\mathsf{If}(2,q_1,q_2)]\!]_f(n,m) \triangleq \begin{cases} a^n b^m q_1 & \text{if } m = 0 \\ a^n b^p q_2 & \text{if } m = p+1 \end{cases}$$

$$[\![\mathsf{Halt}(x)]\!]_f(n,m) \triangleq c_x.$$

In particular, R_M defines a (partial) functional relation on T_M.

▶ **Theorem 15** (Soundness)**.** *Given a two-counter machine M and a configuration $s \leq T_M$, suppose that the following inequality holds in $\mathcal{L}\ddot{\Sigma}_M$:*

$$s_r R_M^* \leq \Sigma^*(C_M + c_1)_r + \Sigma_M^* \Sigma_M^{\neq} \ddot{\Sigma}_M^*,$$

where $\Sigma_M^{\neq} \triangleq \sum_{\substack{x,y \in \Sigma \\ x \neq y}} x_l y_r$. If $s \to_{R_M}^* c_x$, then $x = 1$.

Proof of Theorem 15. Suppose that we have some finite sequence of transitions $s = s_0 \to \cdots \to s_n = c_x$. By definition, $(s_i)_l (s_{i+1})_r \leq R_M$ for every $i \in \{0, \ldots, n-1\}$. Thus, we have the following inequality on languages:

$$\begin{aligned} p &\triangleq (s_0)_r \cdot (s_0)_l (s_1)_r \cdots (s_{n-1})_l (s_n)_r \\ &\leq (s_0)_r \cdot R_M \cdot \cdots \cdot R_M \\ &\leq (s_0)_r R_M^* \\ &\leq \Sigma_M^* (C_M + c_1)_r + \Sigma_M^* \Sigma_M^{\neq} \ddot{\Sigma}_M^*. \end{aligned}$$

On the other hand, by shuffling left and right characters,

$$\begin{aligned} p &= (s_0)_r \cdot (s_0)_l (s_1)_r \cdots (s_{n-1})_l (s_n)_r \\ &= (s_0)_r (s_0)_l \cdot (s_1)_r (s_1)_l \cdots (s_{n-1})_r (s_{n-1})_l \cdot (s_n)_r \\ &= s_0 \cdots s_{n-1} (s_n)_r \\ &\leq \Sigma_M^* (\Sigma_M)_r^+. \end{aligned}$$

We can check that the languages $\Sigma_M^*(\Sigma_M)_r^+$ and $\Sigma_M^* \Sigma_M^{\neq} \ddot{\Sigma}_M^*$ are disjoint. Therefore, it must be the case that $p \leq \Sigma_M^*(C_M + c_1)_r$. By projecting out the right components, we find that $\pi_r(p) = s_0 \cdots s_n \leq \Sigma_M^*(C_M + c_1)_r$. We cannot have $\pi_r(p) \leq \Sigma_M^* C_M$, since the last character c_x cannot appear in a string in C_M. Therefore, $\pi_r(p) \leq \Sigma_M^* c_1$, from which we conclude. ◀

▶ **Theorem 17.** *The following two languages are effectively inseparable:*

$$A \triangleq \{\langle M, x\rangle \mid \textit{The two-counter machine } M \textit{ halts on } x \textit{ and outputs } 1\}$$
$$B \triangleq \{\langle M, x\rangle \mid \textit{The two-counter machine } M \textit{ halts on } x \textit{ and outputs } 0\}.$$

In other words, there is a partial computable function f with the following property. Given a machine M, let W_M be the set of inputs accepted by M. Suppose that M_1 and M_0 are such that $W_{M_1} \cap W_{M_0} = \emptyset$, $A \subseteq W_{M_1}$ and $B \subseteq W_{M_0}$. Then $f\langle M_1, M_0\rangle$ is defined and does not belong to $W_{M_1} \cup W_{M_0}$.

Proof of Theorem 17. We implement f as follows. Given an input x, if x does not encode a pair of machines, then the output is undefined. Otherwise, suppose that $x = \langle M_1, M_0 \rangle$. Construct a machine M_η as follows. On an input x, run M_1 and M_0 on $\langle x, x \rangle$ in parallel. If M_i accepts first, then halt and output $1 - i$. If neither accept, then just run forever. We pose $f(x) = \langle M_\eta, M_\eta \rangle$.

We need to show that $f(x) \notin W_{M_1} \cup W_{M_0}$ when $x = \langle M_1, M_0 \rangle$ and the two machines satisfy the above hypotheses. Suppose that $f(x) = \langle M_\eta, M_\eta \rangle \in W_{M_1}$. By the definition of M_η, this means that M_η outputs 0 on $\langle M_\eta \rangle$. Thus $\langle M_\eta, M_\eta \rangle \in B \subseteq W_{M_0}$. This contradicts the hypothesis that $W_{M_1} \cap W_{M_0} = \emptyset$. Thus, $f(x) \notin W_{M_1}$. An analogous reasoning shows that $f(x) \notin W_{M_0}$, which allows us to conclude. ◀

▶ **Lemma 21.** *Suppose that $e : \mathsf{Rel}(L)$. There exists some ρ such that, for every $n \in \mathbb{N}$ and every finite $\Lambda \leq L$, we have the inequality $\Lambda_r e^* \leq \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \,\mathsf{Next}_e^n(\Lambda)_r e^* + \Sigma^* \Sigma^{\neq} \rho$, where $\mathsf{Next}_e^{<n} = \bigcup_{i<n} \mathsf{Next}_e^i(\Lambda)$.*

Proof of Lemma 21. Let $\rho \triangleq \rho' e^*$, where ρ' is the residue of e. Abbreviate $\Sigma^* \Sigma^{\neq} \rho$ as ε. We proceed by induction on n. If $n = 0$, then the goal becomes $\Lambda_r e^* \leq \Sigma^* \,\mathsf{Next}_e^0(\Lambda)_r e^* + \varepsilon$, which holds because $\mathsf{Next}_e^0(\Lambda) = \Lambda$.

Otherwise, for the inductive step, suppose that the goal is valid for n. We need to prove that it is valid for $n + 1$. Recall that $\Lambda' \triangleq \mathsf{Next}_e(\Lambda) \leq L$. We have

$$
\begin{aligned}
&\Lambda_r e^* \\
&= \Lambda_r + \Lambda_r e e^* \\
&\leq \Lambda_r + \Lambda \,\mathsf{Next}_e(\Lambda)_r e^* + \varepsilon && (e \text{ is representable}) \\
&= \Lambda_r + \Lambda \Lambda'_r e^* + \varepsilon \\
&\leq \Lambda_r + \Lambda \left(\Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda')_r + \Sigma^* \,\mathsf{Next}_e^n(\Lambda')_r e^* + \varepsilon \right) + \varepsilon && \text{I.H.} \\
&= \Lambda_r + \Lambda \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda')_r + \Lambda \Sigma^* \,\mathsf{Next}_e^n(\Lambda')_r e^* + \Lambda \varepsilon + \varepsilon \\
&\leq \Sigma^* \Lambda_r + \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda')_r + \Sigma^* \,\mathsf{Next}_e^n(\Lambda')_r e^* + \varepsilon + \varepsilon && (\Lambda \text{ is finite}) \\
&= \Sigma^* \,\mathsf{Next}_e^0(\Lambda)_r + \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda')_r + \Sigma^* \,\mathsf{Next}_e^n(\Lambda')_r e^* + \varepsilon \\
&= \Sigma^* \,\mathsf{Next}_e^{<n+1}(\Lambda)_r + \Sigma^* \,\mathsf{Next}_e^{n+1}(\Lambda)_r e^* + \varepsilon.
\end{aligned}
$$
◀

▶ **Theorem 22.** *If $e : \mathsf{Rel}(L)$, there exists ρ such that, given $n \in \mathbb{N}$ and a finite $\Lambda \leq L$, if $\mathsf{Next}_e^n(\Lambda) = \emptyset$, then $\Lambda_r e^* \leq \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \Sigma^{\neq} \rho$.*

Proof of Theorem 22. Choose the same ρ as in Lemma 21. Then

$$
\begin{aligned}
&\Lambda_r e^* \\
&\leq \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \,\mathsf{Next}_e^n(\Lambda)_r e^* + \Sigma^* \Sigma^{\neq} \rho && \text{by Lemma 21} \\
&= \Sigma^* \,\mathsf{Next}_e^{<n}(\Lambda)_r + \Sigma^* \Sigma^{\neq} \rho.
\end{aligned}
$$
◀

▶ **Lemma 24.** *Derivable terms are closed under all the pre-Kleene algebra operations, with the following caveats: for e^*, we also require that $[e]_0 = 0$; for $e_1 e_2$, the term is also derivable if e_2 isn't, provided that $[e_1]_0 = 0$. We have the following choices of derivatives:*

$$
\begin{aligned}
\delta_x(0) &= 0 & \delta_x(1) &= 0 \\
\delta_x(x) &= 1 & \delta_x(y) &= 0 \quad \text{if } y \neq x \\
\delta_x(e_1 + e_2) &= \delta_x(e_1) + \delta_x(e_2) & \delta_x(e_1 e_2) &= [e_1]_0 \delta_x(e_2) + \delta_x(e_1) e_2 \\
\delta_x(e^*) &= \delta_x(e) e^*,
\end{aligned}
$$

where, by abuse of notation, we treat $[e_1]_0 \delta_x(e_2)$ as 0 when e_2 is not necessarily derivable (since, by assumption, $[e_1]_0 = 0$ in that case).

Proof of Lemma 24. We prove the closure property for products and star. For products, we start by expanding e_1:

$$\begin{aligned} e_1 e_2 &= \left([e_1]_0 + \sum_x x\delta_x(e_1) \right) e_2 \\ &= [e_1]_0 e_2 + \sum_x x\delta_x(e_1)e_2. \end{aligned}$$

If $[e_1]_0 = 0$, the first term gets canceled out, and we obtain $\sum_x x\delta_x(e_1)e_2 = [e_1]_0[e_2]_0 + \sum_x x\delta_x(e_1)e_2$. Otherwise, we know that e_2 is derivable, and we proceed as follows:

$$\begin{aligned} e_1 e_2 &= [e_1]_0 \left([e_2]_0 + \sum_x x\delta_x(e_2) \right) + \sum_x x\delta_x(e_1)e_2 \\ &= [e_1]_0[e_2]_0 + \sum_x [e_1]_0 x\delta_x(e_2) + \sum_x x\delta_x(e_1)e_2 \\ &= [e_1]_0[e_2]_0 + \sum_x x([e_1]_0\delta_x(e_2) + \delta_x(e_1)e_2) \qquad \text{(because } [e_1]_0 x = x[e_1]_0\text{)}, \end{aligned}$$

which allows us to conclude.

For star, assuming that $[e]_0 = 0$, we note that $e^* = 1 + ee^*$, and we apply the closure properties for the other operations. ◀

▶ **Lemma 26.** *Let X be a finite commutable set. Finite-state terms are preserved by all the pre-Kleene algebra operations (for e^*, we additionally require that $[e]_0 = 0$). Moreover, the set of states of the corresponding automata can be effectively computed.*

Proof of Lemma 26. Let's consider all the cases.

- The set $\{0, 1\}$ is an automaton by Lemma 24. Therefore, 0 and 1 are finite state.
- By Lemma 24, if x is a symbol, the set $S = \{x\}$ is a pre-automaton. Therefore, x is finite state because it belongs to the automaton $\bar{S}$.
- Suppose that S_1 and S_2 are finite automata. By Lemma 24, the set $S = \{e_1 + e_2 \mid e_1 \in S_1, e_2 \in S_2\}$ is a pre-automaton. Therefore, if we have finite-state terms e_1 and e_2 of S_1 and S_2, their sum $e_1 + e_2$ is finite state because it belongs to the automaton $\bar{S}$.
- Suppose that S_1 and S_2 are finite automata. By Lemma 24, the set $S = \{e_1 e_2 \mid e_1 \in S_1, e_2 \in S_2\}$ is a pre-automaton. Indeed, $\delta_x(e_1 e_2) = [e_1]_0\delta_x(e_2) + \delta_x(e_1)e_2$ is a sum of elements of S, since

  $$\begin{aligned} [e_1]_0 &\in S_1 \\ \delta_x(e_2) &\in S_2 \\ \delta_x(e_1) &\in S_1 \\ e_2 &\in S_2. \end{aligned}$$

  Therefore, if we have finite-state terms e_1 and e_2 of S_1 and S_2, their product $e_1 e_2$ is finite state because it belongs to the automaton $\bar{S}$.
- Suppose that e is a state of some automaton S such that $[e]_0 = 0$. Define $S' = \{e'e^* \mid e' \in S\}$. By Lemma 24, this set is a pre-automaton. Indeed,

  $$\begin{aligned} \delta_x(e'e^*) &= [e']_0\delta_x(e^*) + \delta_x(e')e^* \\ &= [e']_0\delta_x(e)e^* + \delta_x(e')e^* \\ &= ([e']_0\delta_x(e) + \delta_x(e'))e^*. \end{aligned}$$

  The terms $\delta_x(e)$ and $\delta_x(e')$ are in S. Thus, $[e']_0\delta_x(e) \in S$ and $\delta_x(e'e^*)$ is a sum of terms of S'. Since $e^* = 1e^*$ is an element of S', then it is a state of $\bar{S}'$, and e^* is finite state. ◀

▶ **Lemma 27.** *Let $e \in \mathcal{T}X$ be a state of a finite-state automaton S, and $k \in \mathbb{N}$. We can write*

$$e = \sum\{s \mid s \in \mathcal{S}X, s \leq e, |s| < k\} + \sum\{se_s \mid s \in \mathcal{S}X, |s| = k\},$$

where each $e_s \in S$ for all s, and the size $|s| \in \mathbb{N}$ of a string s is defined by mapping every symbol of s to $1 \in \mathbb{N}$.

Proof of Lemma 27. By induction on k. When $k = 0$, the equation is equivalent to $e = e$, and we are done. Otherwise, suppose that the result is valid for k. We need to prove that it is also valid for $k + 1$. Write

$$e = \sum_{\substack{s \in \mathcal{S}X \\ s \leq e \\ |s| < k}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k}} se_s.$$

By deriving each e_s, we can rewrite this as

$$\begin{aligned} e &= \sum_{\substack{s \in \mathcal{S}X \\ s \leq e \\ |s| < k}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k}} s\left([e_s]_0 + \sum_{x \in X} x\delta_x(e_s)\right) \\ &= \sum_{\substack{s \in \mathcal{S}X \\ s \leq e \\ |s| < k}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k}} s[e_s]_0 + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k}} \sum_{x \in X} sx\delta_x(e_s). \end{aligned} \tag{1}$$

We can see that $[e_s]_0 = 1$ if and only if $s \leq e$: by taking the language interpretation of (1), we can see that a string of size k can only belong to the middle term, since the left and right terms can only account for strings of strictly smaller or larger size, respectively. Thus, we can rewrite (1) as

$$\begin{aligned} e &= \sum_{\substack{s \in \mathcal{S}X \\ s \leq e \\ |s| < k}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k \\ s \leq e}} s[e_s]_0 + \sum_{s, |s| = k} \sum_{x \in X} sx\delta_x(e_s) \\ &= \sum_{\substack{s \in \mathcal{S}X \\ s \leq e \\ |s| < k+1}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s| = k}} \sum_{x \in X} sx\delta_x(e_s). \end{aligned} \tag{2}$$

Given some string s with $|s| = k + 1$, define

$$e'_s \triangleq \sum_{\substack{(s', x) \in \mathcal{S}X \times X \\ s = s'x}} \delta_x(e_{s'}).$$

This sum is well defined because there are only finitely many s' and $x \in X$ such that $s = s'x$: s' must be of size k, and there are only finitely many such strings. Moreover, e'_s is an element of S, since S is closed under taking derivatives and finite sums. We have

$$\begin{aligned} se'_s &= \sum_{\substack{(s', x) \\ |s'| = k \\ s = s'x}} s\delta_x(e_{s'}) \\ &= \sum_{\substack{(s', x) \\ |s'| = k \\ s = s'x}} s'x\delta_x(e_{s'}). \end{aligned}$$

Therefore,

$$\begin{aligned}\sum_{\substack{s\\|s|=k+1}} se'_s &= \sum_{\substack{s\\|s|=k+1}} \sum_{\substack{(s',x)\\|s'|=k\\s=s'x}} s'x\delta_x(e_{s'}) \\ &= \sum_{\substack{(s',x)\\|s'|=k}} s'x\delta_x(e_{s'}) \\ &= \sum_{\substack{s'\\|s'|=k}} \sum_{x\in X} s'x\delta_x(e_{s'}).\end{aligned}$$

Putting everything together, (2) becomes

$$e = \sum_{s\le e,|s|<k+1} s + \sum_{\substack{s\\|s|=k+1}} se'_s, \tag{3}$$

which completes the inductive case. ◀

▶ **Lemma 29.** *Let e have bounded output with fanout k and let Λ be finite. If $s \in \mathsf{Next}_e(\Lambda)$, then $|s| \le (m+1)k$, where $m = \max\{|s'| \mid s' \in \Lambda\}$. Thus, since Σ is finite, $\mathsf{Next}_e(\Lambda)$ is finite.*

Proof of Lemma 29. If $s \in \mathsf{Next}_e(\Lambda)$, by definition, there exists $s' \in \Lambda$ such that $s'_l s_r \le e$. Since e has fanout k, we have

$$|s| = |\pi_r(s'_l s_r)| \le (|\pi_l(s'_l s_r)| + 1)k = (|s| + 1)k \le (n+1)k.$$ ◀

▶ **Lemma 30.** *Bounded-output terms are closed under all the pre-Kleene algebra operations. For e^*, we additionally require that $|\pi_l(s)| \ge 1$ for all strings $s \le e$.*

Proof of Lemma 30. Let's focus on the last point. Suppose that e has fanout k and that $|\pi_l(s)| \ge 1$ for every $s \le e$. We are going to show that e^* has bounded output with fanout $2k$.

Suppose that $s \le e^*$. We can write $s = s_1 \cdots s_n$ such that $s_i \le e$ for every $i \in \{1, \ldots, n\}$. We have, for every $i \in \{1, \ldots, n\}$, $|\pi_r(s_i)| \le (|\pi_l(s_i)| + 1)k$. Thus,

$$\begin{aligned}|\pi_r(s)| &= \sum_{i=1}^{n} |\pi_r(s_i)| \\ &\le \sum_{i=1}^{n} (|\pi_l(s_i)| + 1)k \\ &\le \sum_{i=1}^{n} 2|\pi_l(s_i)|k && \text{(because } |\pi_l(s_i)| \ge 1\text{)} \\ &= \left(\sum_{i=1}^{n} |\pi_l(s_i)|\right) 2k \\ &= |\pi_l(s_0) \cdots \pi_l(s_n)|2k \\ &= |\pi_l(s_0 \cdots s_n)|2k \\ &= |\pi_l(s)|2k \\ &\le (|\pi_l(s)| + 1)2k.\end{aligned}$$ ◀

▶ **Lemma 31.** *Let $e \in \mathcal{T}\ddot{\Sigma}$ be a bounded-output term that is the state of some automaton S. There exists some $k \in \mathbb{N}$ such that e has fanout k and such that, for every $n \in \mathbb{N}$, we can write*

$$e = \sum\{s \mid s \leq e, |s| < n\} + \sum\{se_s \mid s \in \mathcal{S}\ddot{\Sigma}, |s| = n, |\pi_r(s)| \leq (|\pi_l(s)| + 1)k\},$$

where $e_s \in S$ for every s.

Proof of Lemma 31. Let k_0 be the fanout of e. For each $e' \in S$ such that $e' \neq 0$, choose some string $w_{e'} \leq e'$. Define $m \triangleq \max\{|\pi_l(w_{e'})| \mid e' \in S, e' \neq 0\}$ and $k \triangleq (m+1)k_0$. Since $k \geq k_0$, we know that e has fanout k. Moreover, by Lemma 27, we have

$$\begin{aligned} e &= \sum_{\substack{s \leq e \\ |s|<n}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s|=n}} se_s \\ &= \sum_{\substack{s \leq e \\ |s|<n}} s + \sum_{\substack{s \in \mathcal{S}X \\ |s|=n \\ e_s \neq 0}} se_s, \end{aligned}$$

where each e_s is a state of S. If s is such that $|s| = n$ and $e_s \neq 0$, we have $sw_{e_s} \leq e$. Therefore,

$$\begin{aligned} |\pi_r(s)| &\leq |\pi_r(sw_{e_s})| \\ &\leq (|\pi_l(sw_{e_s})| + 1)k_0 \\ &= (|\pi_l(s)| + |\pi_l(w_{e_s})| + 1)k_0 \\ &\leq (|\pi_l(s)| + m + 1)k_0 \\ &\leq (|\pi_l(s)| + 1)(m+1)k_0 \\ &= (|\pi_l(s)| + 1)k. \end{aligned}$$

Thus,

$$\begin{aligned} e &= \sum_{\substack{s \leq e \\ |s|<n}} s + \sum_{\substack{s \\ |s|=n \\ e_s \neq 0 \\ |\pi_r(s)| \leq (|\pi_l(s)|+1)k}} se_s \\ &= \sum_{\substack{s \leq e \\ |s|<n}} s + \sum_{\substack{s \\ |s|=n \\ |\pi_r(s)| \leq (|\pi_l(s)|+1)k}} se_s. \end{aligned}$$

◀

▶ **Lemma 33** (normal)**.** *Let s and s' be two strings over Σ such that one is not a prefix of the other, or vice versa. Then we can write $s = s_0 x s_1$ and $s' = s_0 x' s_1'$ with $x \neq x'$. Thus, $s_r s_l' \ddot{\Sigma}^* \leq \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^*$.*

Proof of Lemma 33. By induction on the length of s. ◀

▶ **Lemma 34.** *Suppose that $e \in \mathcal{T}\ddot{\Sigma}$ is such that $\pi_l(e) \leq L$ and $\pi_r(e) \leq L$, where L is prefix free. Suppose, moreover, that e is finite-state and has bounded output. Then $e : \mathsf{Rel}(L)$.*

Proof of Lemma 34. We have already seen that $\mathsf{Next}_e(\Lambda)$ is finite when Λ is (Lemma 29). Thus, we need to find some ρ such that, for every finite Λ,

$$\Lambda_r e \leq \Lambda\, \mathsf{Next}_e(\Lambda)_r + \Sigma^* \Sigma^{\neq} \rho.$$

Define $\rho \triangleq \ddot{\Sigma}^* \rho_e$, where ρ_e is the greatest element of the automaton of e. It suffices to prove the result for the case $\Lambda = \{s\}$. Indeed, if the result holds for singletons, we have

$$\begin{aligned}
\Lambda_r e &= \sum_{s \in \Lambda} s_r e \\
&\leq \sum_{s \in \Lambda} s\, \mathsf{Next}_e(s)_r + \Sigma^* \Sigma^{\neq} \rho && \text{by assumption} \\
&\leq \sum_{s \in \Lambda} \Lambda\, \mathsf{Next}_e(s)_r + \Sigma^* \Sigma^{\neq} \rho \\
&= \Lambda \sum_{s \in \Lambda} \mathsf{Next}_e(s) + \Sigma^* \Sigma^{\neq} \rho \\
&= \Lambda\, \mathsf{Next}_e(\Lambda) + \Sigma^* \Sigma^{\neq} \rho. && \blacktriangleleft
\end{aligned}$$

Let k be the constant of Lemma 31 for e, $n = |s|$, and let $p = (k+1)(n+1)$. Let

$$\ddot{\Lambda} \triangleq \{s' \in \mathcal{S}\ddot{\Sigma} \mid |s'| = p+1, |\pi_r(s')| \leq (|\pi_l(s')|+1)k\}.$$

By applying Lemma 31 to e, we can write

$$\begin{aligned}
e &= \sum_{\substack{s' \leq e \\ |s'| < p+1}} s' + \sum_{s' \in \ddot{\Lambda}} s' e_{s'} \\
&= \sum_{s' \leq e, |s'| \leq p} s' + \sum_{s' \in \ddot{\Lambda}} s' e_{s'} \\
&= \sum_{\substack{s' \leq e \\ |s'| \leq p \\ \pi_l(s') = s}} s' + \sum_{\substack{s' \leq e \\ |s'| \leq p \\ \pi_l(s') \neq s}} s' + \sum_{s' \in \ddot{\Lambda}} s' e_{s'},
\end{aligned}$$

Thus, to prove the inequality, it suffices to prove

$$s_r \sum_{\substack{s' \leq e \\ |s'| \leq p \\ \pi_l(s') = s}} s' = s\, \mathsf{Next}_e(s)_r \tag{4}$$

$$s_r \sum_{\substack{s' \leq e \\ |s'| \leq p \\ \pi_l(s') \neq s}} s' \leq \Sigma^* \Sigma^{\neq} \rho \tag{5}$$

$$s_r \sum_{s' \in \ddot{\Lambda}} s' e'_s \leq \Sigma^* \Sigma^{\neq} \rho. \tag{6}$$

Let us start with (4). Notice that, for any string s' over $\ddot{\Sigma}$, we have $s' = \pi_l(s')_l \pi_r(s')_r$. Therefore, there is a bijection between the set of indices s' of the sum and the set of strings $\mathsf{Next}_e(s)$. The bijection is given by

$$\begin{aligned}
s' &\mapsto \pi_r(s') \in \mathsf{Next}_e(s) \\
\mathsf{Next}_e(s) \ni s' &\mapsto s_l s'_r.
\end{aligned}$$

To prove that this is a bijection, we must show that the inverse produces indeed a valid index. Notice that, if $s' \in \mathsf{Next}_e(s)$, by Lemma 29, we have $|s'| \leq (n+1)k$, and thus $|s_l s'_r| = |s| + |s'| \leq (n+1)(k+1) = p$.

By reindexing the sum in (4) with this bijection, we have

$$\begin{aligned} s_r \sum_{\substack{s' \le e \\ |s'| \le p \\ \pi_l(s') = s}} s' &= s_r \sum_{s' \in \mathsf{Next}_e(s)} s_l s'_r \\ &= s_r s_l \sum_{s' \in \mathsf{Next}_e(s)} s'_r \\ &= s_r s_l \left(\sum_{s' \in \mathsf{Next}_e(s)} s' \right)_r \\ &= s\, \mathsf{Next}_e(s)_r. \end{aligned}$$

Next, let us look at (5). Suppose that s' is such that $s' \le e$ and $\pi_l(s') \neq s$. Since L is prefix free, and $\pi_l(s') \le L$, Lemma 33 applied to s and s' yields

$$s_l s' \le \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^* \le \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^* \rho_e = \Sigma^* \Sigma^{\neq} \rho,$$

where we use the fact that $\rho_e \ge 1$ because 1 is a state of the automaton of e. Summing over all such s', we get the desired inequality.

To conclude, we must show (6). By distributivity, this is equivalent to showing that, for every $s' \in \Lambda$,

$$s_r s' e_{s'} \le \Sigma^* \Sigma^{\neq} \rho.$$

If $e_{s'} = 0$, we are done. Otherwise, by Corollary 8, we can find some string $s'' \le e_{s'}$. We have $s's'' \le s' e_{s'} \le e$.

Note that we must have $|\pi_l(s')| > n$. Indeed, suppose that $|\pi_l(s')| \le n$. Since $s' \in \Lambda$, we have

$$\begin{aligned} |s'| &= |\pi_l(s')| + |\pi_r(s')| \\ &\le |\pi_l(s')| + (|\pi_l(s')| + 1)k \\ &\le (|\pi_l(s')| + 1)(k+1) \\ &\le (n+1)(k+1) \\ &< p + 1 \\ &= |s'|, \end{aligned}$$

which is a contradiction.

Since $\pi_l(s's'') \le \pi_l(e) \le L$ and L is prefix free, by Lemma 33, we can write $s = s_0 x s_1$ and $\pi_l(s's'') = \pi_l(s')\pi_l(s'') = s_0 x' s'_1$, with $x \neq x'$. But $|\pi_l(s')| > n = |s|$ and $|s_0| < |s|$, thus $\pi_l(s')$ must be of the form $s_0 x' s'_2$. We find that $s_r s' = s_r \pi_l(s')\pi_r(s') \le \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^*$, and thus

$$s_r s' e_{s'} \le \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^* e_{s'} \le \Sigma^* \Sigma^{\neq} \ddot{\Sigma}^* \rho_e = \Sigma^* \Sigma^{\neq} \rho.$$

Finite Relational Semantics for Language Kleene Algebra with Complement

Yoshiki Nakamura
Institute of Science Tokyo, Japan

Abstract

We study the equational theory of Kleene algebra (KA) w.r.t. languages (here, meaning the equational theory of regular expressions where each letter maps to any language) by extending the algebraic signature with the language complement. This extension significantly enhances the expressive power of KA. In this paper, we present a *finite relational semantics* completely characterizing the equational theory w.r.t. languages, which extends the relational characterizations known for KA and for KA with top. Based on this relational semantics, we show that the equational theory w.r.t. languages is Π^0_1-complete for KA with complement (with or without Kleene-star) and is PSPACE-complete if the complement only applies to variables or constants.

2012 ACM Subject Classification Theory of computation → Equational logic and rewriting; Theory of computation → Logic and verification; Theory of computation → Formal languages and automata theory

Keywords and phrases Kleene algebra, Language model, Relational model, Complexity

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.37

Related Version *Full Version*: `https://hal.science/hal-04455882` [35]

Funding This work was supported by JSPS KAKENHI Grant Number JP21K13828.

Acknowledgements We thank anonymous reviewers for useful comments.

1 Introduction

Kleene algebra (KA) [24, 11, 25] is an algebraic system for regular expressions consisting of identity (1), empty (0), composition ($;$), union ($+$), and iteration ($\_^*$). As iteration frequently appears in computer science, KA has many applications, e.g., the semantics of programs [46], relation algebra [40], graph query language [12, 21], program verification [29, 23, 48], and program logics [26, 41, 53]. In practice, we often consider extensions of KA. One direction of extensions is to extend equations to formulas, e.g., Horn formulas ($t_1 = s_1 \to \cdots \to t_n = s_n \to t = s$) for considering hypotheses [9, 28, 14, 44]. Another direction is to extend terms by adding some operators. For example, Kleene algebra with tests (KAT) applies to model Hoare logic [26] and KAT with top ($\top$) applies to model incorrectness logic [41, 53, 45]. It is also natural to extend KA with language operators, e.g., reverse [3], residual [8], intersection ($\cap$) [2], top (universality) [53, 45], variable complements ($\overline{x}$) [38, 39], and combinations of some of them [4, 5]. Note that, whereas the class of regular languages is closed under these operators, such extensions strictly enhance the expressive power of KA w.r.t. languages (here, meaning regular expressions where each letter maps to any language); see [38, 39] and Section 2.2 for complement.

In this paper, we study KA w.r.t. languages by extending the algebraic signature with the *language complement* ($\_^-$). Extending with complement and considering its fragments is a natural, comprehensive approach, e.g., in logic, formal language [10, 42], and relation

algebra [50, 40] (see also [1, 6, 32, 43, 34]). The language complement[1] in KA w.r.t. languages significantly enhances the expressive power. For instance, we can define $\top$ and $\cap$ using complement: $\top = 0^-$ and $t \cap s = (t^- + s^-)^-$. Additionally, we can encode positive quantifier-free formulas by equations of KA terms with complement (Remark 3.3 and Section A).

Our main contribution is to present *a finite relational semantics* for KA with complement w.r.t. languages: relational subword models RSUB (Section 3). As KA with complement has a high expressive power, our relational semantics can apply to a more broad class of extensions of KA (including KA with $\top$ and $\cap$) than known relational semantics, e.g., REL (for KA) [46, third page] and GREL (for KA with $\top$) [53, 45] (see Remark 3.5). A good point of RSUB is its form; each model is finite and totally ordered (with minimal and maximal vertices). For instance, the Π^0_1 upper bound result of the equational theory of KA with complement w.r.t. languages is immediate from the finiteness of RSUB. Another good point is that we can naturally consider lifting techniques known in REL to LANG. For instance, by using the techniques in [34] w.r.t. REL, we can show the following complexity results: the equational theory w.r.t. languages is Π^0_1-complete for KA with intersection and variable complements (Theorem 4.12) and for KA with complement and without Kleene-star (i.e., star-free regular expressions w.r.t. LANG) (Theorem 4.15); and PSPACE-complete for KA with variable and constant complements (Theorem 6.10). The PSPACE decidability result above positively settles the open problem posed in [38, Sect. 7].

This paper is structured as follows. In Section 2, we give basic definitions, including language models (LANG) and generalized relational models (GREL). In Section 3, we introduce RSUB (a subclass of GREL) and show that the equational theory w.r.t. LANG coincides with that w.r.t. RSUB. In Section 4, by using RSUB, we give a reduction from the quantifier-free theory w.r.t. LANG into the equational theory w.r.t. LANG. Using this reduction, we show that the equational theory w.r.t. LANG is Π^0_1-complete for KA with complement (moreover, for KA with intersection and variable complements and for KA with complement and without Kleene-star). In Section 5, by using RSUB, we give a graph characterization for KA terms with variable and constant complements. In Section 6, by using this characterization, we show that the equational theory for KA terms with variable and constant complements is PSPACE-complete. In Section 7, we conclude this paper.

2 Preliminaries

We write $\mathbb{N}$ for the set of non-negative integers. For $l, r \in \mathbb{N}$, we write $[l, r]$ for the set $\{i \in \mathbb{N} \mid l \leq i \leq r\}$. For a set X, we write $\wp(X)$ for the power set of X.

For a set X (of letters), we write X^* for the set of words over X. A *language* over X is a subset of X^*. We use w, v to denote words and use L, K to denote languages, respectively. We write $\|w\|$ for the *length* of a word w. We write ε for the empty word. We write wv for the concatenation of words w and v. For languages $L, K \subseteq X^*$, the concatenation $L\,;K$ and the Kleene-star L^* is defined by:

$$L\,;K \triangleq \{wv \mid w \in L \ \wedge \ v \in K\}, \qquad L^* \triangleq \bigcup_{n \geq 0} \{\varepsilon\}\,;\underbrace{L\,;\cdots;L}_{n \text{ times}}.$$

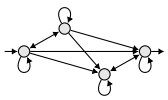
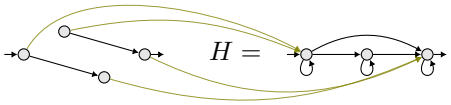
A *(2-pointed) graph* G over a set A is a tuple $\langle |G|, \{a^G\}_{a \in A}, 1^G, 2^G \rangle$, where $|G|$ is a non-empty set (of vertices), each $a^G \subseteq |G|^2$ is a binary relation, and $1^G, 2^G \in |G|$ are vertices. Let G, H be graphs over a set A. For a map $f\colon |G| \to |H|$, we say that f is a *graph*

[1] KAT [29] is also an extension of KA with complement, but this complement is not the language complement.

homomorphism from G to H, written $f\colon G \longrightarrow H$, if for all x, y, and a, $\langle x, y\rangle \in a^G$ implies $\langle f(x), f(y)\rangle \in a^H$, $f(1^G) = 1^H$, and $f(2^G) = 2^H$. We say that f is a *graph isomorphism* from G to H if f is a bijective graph homomorphism and for all x, y, and a, $\langle x, y\rangle \in a^G$ iff $\langle f(x), f(y)\rangle \in a^H$. We say that H is a (canonical) *edge-extension* of G if $|H| = |G|$ and the identity map is a graph homomorphism from G to H. For a set $\{1^G, 2^G\} \subseteq X \subseteq |G|$, the *induced subgraph* of G on X is the graph $\langle X, \{a^G \cap X^2\}_{a\in A}, 1^G, 2^G\rangle$. For an equivalence relation E on $|G|$, the *quotient graph* of G w.r.t. E is the graph $G/E \triangleq \langle |G|/E, \{\langle X, Y\rangle \mid \exists x \in X, y \in Y, \langle x, y\rangle \in a^G\}_{a\in A}, [1^G]_E, [2^G]_E\rangle$ where X/E denotes the set of equivalence classes of X by E and $[x]_E$ denotes the equivalence class of x. Additionally, we use the following operation:

▶ **Definition 2.1.** *For a graph homomorphism $h\colon G \longrightarrow H$ where G, H are graphs over a set A, the* edge-saturation *of G w.r.t. h is the graph $\mathcal{S}(h) \triangleq \langle |G|, \{\{\langle x, y\rangle \in |G|^2 \mid \langle h(x), h(y)\rangle \in a^H\}\}_{a\in A}, 1^G, 2^G\rangle$.*

▶ **Example 2.2.** Let $h\colon G \longrightarrow H$ be the graph homomorphism indicated by green colored arrows (graphs are depicted as unlabeled graphs for simplicity). Then $\mathcal{S}(h)$ is the following graph in the left-hand side, which is an edge-extension of G where the extended edges are derived from edges of H:

$\mathcal{S}(h) =$  , $G =$  .

2.1 Syntax: terms of KA with complement

We consider *terms* over the signature $\mathrm{S} \triangleq \{1_{(0)}, 0_{(0)}, ;_{(2)}, +_{(2)}, {}^*_{(1)}, {}^-_{(1)}\}$. Let $\mathbf{V}$ be a countably infinite set of variables. For a term t over S, let $\bar{t}$ be s if $t = s^-$ for some s and be t^- otherwise. We use the abbreviations: $\top \triangleq 0^-$ and $t \cap s \triangleq (t^- + s^-)^-$.

For $X \subseteq \{\overline{x}, \overline{1}, \top, \cap, -\}$, let KA_X be the minimal set A of terms over S satisfying:

$$\frac{y \in \mathbf{V}}{y \in A} \qquad \frac{}{1 \in A} \qquad \frac{}{0 \in A} \qquad \frac{t \in A \quad s \in A}{t\,;s \in A} \qquad \frac{t \in A \quad s \in A}{t + s \in A} \qquad \frac{t \in A}{t^* \in A}$$

$$\frac{\overline{x} \in X \quad y \in \mathbf{V}}{\overline{y} \in A} \qquad \frac{\overline{1} \in X}{\overline{1} \in A} \qquad \frac{\top \in X}{\top \in A} \qquad \frac{\cap \in X \quad t \in A \quad s \in A}{t \cap s \in A} \qquad \frac{- \in X \quad t \in A}{t^- \in A}.$$

We often abbreviate $t\,;s$ to ts. We use parentheses in ambiguous situations (where $+$ and $;$ are left-associative). We write $\sum_{i=1}^n t_i$ for the term $0 + t_1 + \cdots + t_n$.

An *equation* $t = s$ is a pair of terms. An *inequation* $t \leq s$ abbreviates the equation $t + s = s$. The set of *quantifier-free formulas* of KA_X is defined by the following grammar:

$$\varphi, \psi \quad ::= \quad t = s \mid \varphi \wedge \psi \mid \neg\varphi. \qquad (t, s \in \mathrm{KA}_X)$$

We use the following abbreviations, as usual: $\varphi \vee \psi \triangleq \neg(\neg\varphi \wedge \neg\psi)$, $\varphi \to \psi \triangleq \neg\varphi \vee \psi$, $\varphi \leftrightarrow \psi \triangleq (\varphi \to \psi) \wedge (\psi \to \varphi)$, $\mathsf{f} \triangleq \neg\varphi \wedge \varphi$, and $\mathsf{t} \triangleq \neg\mathsf{f}$. We use parentheses in ambiguous situations (where $\vee$ and $\wedge$ are left-associative). We write $\bigwedge_{i=1}^n \varphi_i$ for $\mathsf{t} \wedge \varphi_1 \wedge \cdots \wedge \varphi_n$ and $\bigvee_{i=1}^n \varphi_i$ for $\mathsf{f} \vee \varphi_1 \vee \cdots \vee \varphi_n$.

We say that a quantifier-free formula is *positive* if the formula in the following set A:

$$\varphi, \psi \in A \quad ::= \quad t = s \mid \varphi \wedge \psi \mid \varphi \vee \psi \qquad (t, s \in \mathrm{KA}_X)$$

where $\varphi \vee \psi$ expresses $\neg(\neg\varphi \wedge \neg\psi)$ in the above. We say that a quantifier-free formula is a *Horn formula* if the formula is of the form $(\bigwedge_{i=1}^n \varphi_i) \to \psi$ where $n \geq 0$.

2.2 Semantics: language models

An S-*algebra* $\mathcal{A}$ is a tuple $\langle |\mathcal{A}|, \{f^{\mathcal{A}}\}_{f_{(k)} \in \mathrm{S}} \rangle$, where $|\mathcal{A}|$ is a non-empty set and $f^{\mathcal{A}} \colon |\mathcal{A}|^k \to |\mathcal{A}|$ is a k-ary map for each $f_{(k)} \in \mathrm{S}$. A *valuation* $\mathfrak{v}$ of an S-algebra $\mathcal{A}$ is a map $\mathfrak{v} \colon \mathbf{V} \to |\mathcal{A}|$. For a valuation $\mathfrak{v}$, we write $\hat{\mathfrak{v}} \colon \mathrm{KA}_{\{-\}} \to |\mathcal{A}|$ for the unique homomorphism extending $\mathfrak{v}$. Moreover, for a quantifier-free formula φ, we define $\hat{\mathfrak{v}}(\varphi) \in \{\mathsf{true}, \mathsf{false}\}$ by:

$$\hat{\mathfrak{v}}(t = s) \triangleq (\hat{\mathfrak{v}}(t) = \hat{\mathfrak{v}}(s)), \qquad \hat{\mathfrak{v}}(\varphi \wedge \psi) \triangleq (\hat{\mathfrak{v}}(\varphi) \text{ and } \hat{\mathfrak{v}}(\psi)), \qquad \hat{\mathfrak{v}}(\neg\varphi) \triangleq (\text{not } \hat{\mathfrak{v}}(\varphi)).$$

For a quantifier-free formula φ and a class of valuations (of S-algebra) $\mathcal{C}$,[2] we write

$$\mathcal{C} \models \varphi \iff \hat{\mathfrak{v}}(\varphi) \text{ holds for all valuations } \mathfrak{v} \in \mathcal{C}.$$

We abbreviate $\{\mathfrak{v}\} \models \varphi$ to $\mathfrak{v} \models \varphi$. The *equational theory w.r.t.* $\mathcal{C}$ is the set of all equations $t = s$ such that $\mathcal{C} \models t = s$. The *quantifier-free theory w.r.t.* $\mathcal{C}$ is the set of all quantifier-free formulas φ such that $\mathcal{C} \models \varphi$.

The *language model* $\mathcal{A}$ over a set X, written lang_X, is the S-algebra defined by $|\mathcal{A}| = \wp(X^*)$, $1^{\mathcal{A}} = \{\varepsilon\}$, $0^{\mathcal{A}} = \emptyset$, and for all $L, K \subseteq X^*$,

$$L \mathbin{;^{\mathcal{A}}} K = L \,;\, K, \qquad L \mathbin{+^{\mathcal{A}}} K = L \cup K, \qquad L^{*^{\mathcal{A}}} = L^*, \qquad L^{-^{\mathcal{A}}} = X^* \setminus L.$$

We write LANG_X for the class of all valuations of lang_X and write LANG for $\bigcup_X \mathsf{LANG}_X$. The *equational theory (resp. quantifier-free theory) w.r.t. languages* expresses that w.r.t. LANG.

The language $[t] \subseteq \mathbf{V}^*$ of a term t is $\hat{\mathfrak{v}}_{\mathrm{st}}(t)$ where $\mathfrak{v}_{\mathrm{st}}$ is the standard language valuation on the language model over the set $\mathbf{V}$, which is defined by $\mathfrak{v}_{\mathrm{st}}(x) = \{x\}$ for $x \in \mathbf{V}$. Since $\mathfrak{v}_{\mathrm{st}} \in \mathsf{LANG}$, we have

$$\mathsf{LANG} \models t = s \quad \Rightarrow \quad [t] = [s] \tag{$\dagger$}$$

The converse direction fails; e.g., when $x \neq y$, we have $[y] \subseteq [\overline{x}]$ and $\mathsf{LANG} \not\models y \leq \overline{x}$, because $[y] = \{y\} \subseteq \mathbf{V}^* \setminus \{x\} = [\overline{x}]$ and $\hat{\mathfrak{v}}(y) = \{\varepsilon\} \not\subseteq \mathbf{V}^* \setminus \{\varepsilon\} = \hat{\mathfrak{v}}(\overline{x})$ where $\mathfrak{v}$ is a valuation of lang_X s.t. $\mathfrak{v}(x) = \mathfrak{v}(y) = \{\varepsilon\}$. See [38] for more counter-examples.

▶ Remark 2.3. For (non-extended) KA, the equational theory w.r.t. languages coincides with the language equivalence [25, 2] (i.e., the converse direction of Equation ($\dagger$) also holds). This is an easy consequence of the completeness theorem of KA [25] (see also [38, Appendix A] for a direct proof). From this, KA with complement (even with variable complements) has a strictly more expressive power than KA.

In the sequel, we consider the equational theory w.r.t. languages.

2.3 (Generalized) relational models

We write $\triangle_A$ for the identity relation on a set A: $\triangle_A \triangleq \{\langle x, x\rangle \mid x \in A\}$. For binary relations R, S on a set B, the composition $R \,;\, S$, and the reflexive transitive closure R^* are defined by:

$$R \,;\, S \triangleq \{\langle x, z\rangle \mid \exists y, \langle x, y\rangle \in R \wedge \langle y, z\rangle \in S\}, \qquad R^* \triangleq \bigcup_{n \geq 0} \triangle_B \,;\, \underbrace{R \,;\, \cdots \,;\, R}_{n \text{ times}}.$$

[2] This paper considers classes of valuations rather than classes of S-algebras (cf. Theorem 3.6).

Let U be a binary relation on a non-empty set B. A *generalized relational model*[3] $\mathcal{A}$ on U is an S-algebra such that $|\mathcal{A}| \subseteq \wp(U)$, $1^{\mathcal{A}} = \triangle_B$, $0^{\mathcal{A}} = \emptyset$, and for all $R, S \subseteq U$,

$$R ;^{\mathcal{A}} S = R ; S, \qquad R +^{\mathcal{A}} S = R \cup S, \qquad R^{*^{\mathcal{A}}} = R^*, \qquad R^{-^{\mathcal{A}}} = U \setminus R.$$

We say that $\mathcal{A}$ is a *relational model* if $U = B^2$ and $|\mathcal{A}| = \wp(B^2)$. We write GREL (resp. REL) for the class of all valuations of generalized relational models (resp. relational models).[4]

Let $\mathcal{A}$ be a generalized relational model on a binary relation U on a set A. For a non-empty subset $B \subseteq A$, the (induced) *submodel* $\mathcal{A} \restriction B$ of $\mathcal{A}$ w.r.t. B is the generalized relational model on the binary relation $U \cap B^2$ on the set B with the universe $\{R \cap B^2 \mid R \in |\mathcal{A}|\}$. We say that a non-empty subset $B \subseteq A$ is *$\top$-closed* if for all $x, y, z \in A$, if $\langle x, z\rangle, \langle z, y\rangle \in U$ and $x, y \in B$, then $z \in B$. When B is $\top$-closed, it is easy to see that the map

$$\kappa_B \colon R \mapsto R \cap B^2$$

forms an S-homomorphism from $\mathcal{A}$ to $\mathcal{A} \restriction B$ (the condition is used for preserving ; and $*$). Similarly, for a valuation $\mathfrak{v}$ of $\mathcal{A}$, let $\mathfrak{v} \restriction B$ be the valuation of $\mathcal{A} \restriction B$ given by the map κ_B.

3 RSUB: finite relational models for language models

In this section, we define the class RSUB of *relational subword models*, for the equational theory w.r.t. languages of $\mathsf{KA}_{\{-\}}$. RSUB is a subclass of finite generalized relational models where the universe relation U is a total order.

▶ **Definition 3.1.** *Let $n \in \mathbb{N}$. The relational subword language model $\mathcal{A}$ of length n, written rsub_n, is the generalized relational model on the set $U = \{\langle i, j\rangle \in [0, n]^2 \mid i \leq j\}$ s.t.*

$$|\mathcal{A}| \quad = \quad \{R \in \wp(U) \mid R \supseteq \triangle_{[0,n]} \ \vee \ U \setminus R \supseteq \triangle_{[0,n]}\}.$$

We write RSUB_n for the class of all valuations of rsub_n and write RSUB for $\bigcup_{n\geq 0} \mathsf{RSUB}_n$. ⌟

Each rsub_n is based on the image of Pratt's embedding (or called Cayley map) [46][5]:

$$\iota_X \colon L \mapsto \{\langle w, wv\rangle \mid w \in X^* \wedge v \in L\}$$

where we restrict the universe X^* of words into the subwords of a word of length n with *pairwise distinct* letters (i.e., a subword of length i corresponds to the vertex i in rsub_n).

Let rlang_X be the generalized relational model on $\iota_X(X^*)$ with the universe $\{\iota_X(L) \mid L \subseteq X^*\}$. It is easy to see that the map ι_X forms an S-isomorphism from lang_X to rlang_X. For a word w, let $\mathrm{Subw}(w)$ be the set of subwords of w. By Definition 3.1, it is easily shown that

- for a word $w \in X^*$ of length n, the generalized relational model $\mathsf{rlang}_X \restriction \mathrm{Subw}(w)$ is isomorphic to a subalgebra of rsub_n,
- for a word $w = a_1 \dots a_n \in X$ where $a_1, \dots, a_n$ are pairwise distinct letters, the generalized relational model $\mathsf{rlang}_X \restriction \mathrm{Subw}(w)$ is isomorphic to rsub_n,

[3] By definition, for each generalized relational model, U is a preorder: (Reflexivity): By $\triangle_B = 1^{\mathcal{A}} \in |\mathcal{A}| \subseteq \wp(U)$, we have $\triangle_B \subseteq U$; (Transitivity): By $\emptyset = 0^{\mathcal{A}} \in |\mathcal{A}|$, $U = \emptyset^{-^{\mathcal{A}}} \in |\mathcal{A}|$, and $U ; U = U ;^{\mathcal{A}} U \in |\mathcal{A}| \subseteq \wp(U)$, we have $U ; U \subseteq U$.

[4] Generalized relational models and relational models are variants of proper relation algebras and full proper relation algebras (see, e.g., [22]), respectively, where B is non-empty set and the converse operator is not introduced (due to this, U is possibly not symmetric, cf. [22, Lem. 3.4]) here.

[5] This trick itself is already used to prove equivalences between relational and language models, e.g., for KAT [29] and for $\mathsf{KA}_{\{\top\}}$ [53, 45].

by the map

$$\theta\colon R \mapsto \{\langle \|w\|, \|v\| \rangle \mid \langle w, v \rangle \in R\}.$$

We then have that the equational theory w.r.t. languages coincides with that w.r.t. RSUB.

▶ **Theorem 3.2.** *For all* $\mathsf{KA}_{\{-\}}$ *terms* t *and* s*, we have:* $\mathsf{LANG} \models t \leq s \Leftrightarrow \mathsf{RSUB} \models t \leq s$.

Proof. ($\Rightarrow$): For each $n \in \mathbb{N}$, by the *surjective* S-homomorphism given by:

$$\mathsf{lang}_X \xrightarrow{\iota_X} \mathsf{rlang}_X \xrightarrow{\kappa_{\mathrm{Subw}(a_1 \dots a_n)}} \mathsf{rlang}_X \restriction \mathrm{Subw}(a_1 \dots a_n) \xrightarrow{\theta} \mathsf{rsub}_n$$

where $a_1, \dots, a_n$ are any pairwise distinct letters and $X = \{a_1, \dots, a_n\}$. (As $\mathrm{Subw}(a_1 \dots a_n)$ is $\top$-closed, $\kappa_{\mathrm{Subw}(a_1 \dots a_n)}$ is indeed an S-homomorphism.) ($\Leftarrow$): We prove the contraposition. By $\mathsf{LANG} \not\models t \leq s$, there are X, $\mathfrak{v} \in \mathsf{LANG}_X$, and $w_0 \in X^*$ such that $w_0 \in \hat{\mathfrak{v}}(t) \setminus \hat{\mathfrak{v}}(s)$. We then consider the S-homomorphism given by:

$$\mathsf{lang}_X \xrightarrow{\iota_X} \mathsf{rlang}_X \xrightarrow{\kappa_{\mathrm{Subw}(w_0)}} \mathsf{rlang}_X \restriction \mathrm{Subw}(w_0) \xrightarrow{\theta} \mathsf{rsub}_{\|w_0\|}$$

Let $\mathfrak{v}'$, $\mathfrak{v}''$, and $\mathfrak{v}'''$ be the valuations of rlang_X, $\mathsf{rlang}_X \restriction \mathrm{Subw}(w_0)$, and rsub_n, given by $\iota_X \circ \mathfrak{v}$, $\kappa_{\mathrm{Subw}(w_0)} \circ \mathfrak{v}'$, and $\theta \circ \mathfrak{v}''$, respectively. We then have:

$$\begin{aligned}
& w_0 \in \hat{\mathfrak{v}}(t) \setminus \hat{\mathfrak{v}}(s) && \Rightarrow \quad \langle \varepsilon, w_0 \rangle \in \hat{\mathfrak{v}}'(t) \setminus \hat{\mathfrak{v}}'(s) && \text{(By } w_0 \in L \text{ iff } \langle \varepsilon, w_0 \rangle \in \iota_X(L)\text{)} \\
& \Rightarrow && \langle \varepsilon, w_0 \rangle \in \hat{\mathfrak{v}}''(t) \setminus \hat{\mathfrak{v}}''(s) && \text{(By } \varepsilon, w_0 \in \mathrm{Subw}(w_0)\text{)} \\
& \Rightarrow && \langle 0, \|w_0\| \rangle \in \hat{\mathfrak{v}}'''(t) \setminus \hat{\mathfrak{v}}'''(s). && \text{(By } \langle \varepsilon, w_0 \rangle \in R \text{ iff } \langle 0, \|w_0\| \rangle \in \theta(R)\text{)}
\end{aligned}$$

Hence, $\mathsf{RSUB} \not\models t \leq s$. ◀

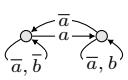
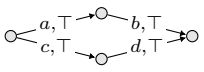
▶ Remark 3.3. By almost the same argument as Theorem 3.2, we can extend the coincidence between LANG and RSUB from the equational theory to the *positive* quantifier-free theory (see Section A for more details). However, this coincidence is broken (only $\mathsf{LANG} \models \varphi \Leftarrow \mathsf{RSUB} \models \varphi$ holds) for the quantifier-free theory and even for Horn theory. For instance, $\varphi \triangleq xx \leq 0 \to x \leq 0$ is a counter-example ($\mathsf{LANG} \models \varphi$ holds because, if $w \in \hat{\mathfrak{v}}(x)$, then $ww \in \hat{\mathfrak{v}}(xx)$; however, $\mathsf{RSUB}_1 \not\models \varphi$ under the valuation $x \mapsto \{\langle 0, 1 \rangle\}$). ⌟

▶ **Corollary 3.4.** *The equational theory w.r.t. languages is in* Π^0_1 *for* $\mathrm{KA}_{\{-\}}$ *terms.*

Proof. By the finite model property of RSUB (the universe $|\mathsf{rsub}_n|$ is finite for each n). ◀

Comparison to other semantics

▶ Remark 3.5 (RSUB and GREL). For $\mathrm{KA}_{\{\top\}}$, the equational theory of LANG coincides with that of GREL [45, REL′ in Sect. 5][53]. However for $\mathrm{KA}_{\{-\}}$, this coincidence is broken. For instance, the following equations are valid w.r.t. LANG but not valid w.r.t. GREL (the first equation is not valid also w.r.t. REL):

$$a \leq \overline{b}a\overline{b} + bab \qquad \text{  } \qquad | \qquad ab \cap cd \leq a\top d + c\top b \qquad \text{  }$$

(Each figure expresses a valuation for (G)REL $\not\models$ _ where some edges are omitted.) Here, LANG $\models a \leq \overline{b}a\overline{b} + bab$ is shown by distinguishing the cases based on $\mathsf{LANG} \models 1 \leq b \vee 1 \leq \overline{b}$. The inequation $ab \cap cd \leq a\top d + c\top b$ is Levi's inequation [30][5, Example 26]. ⌟

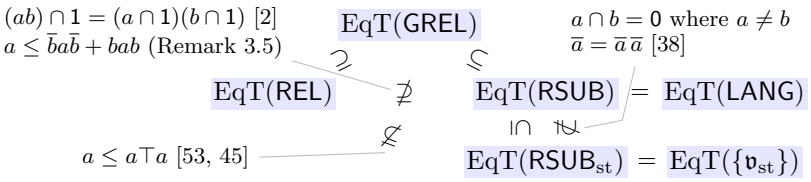


Figure 1 Equational theories for $\mathrm{KA}_{\{-\}}$ under GREL.

Additionally, the standard language valuation can also be given as a subclass of RSUB (cf. Theorem 3.2), based on the following correspondence between words and relations:

$$a_1 a_2 \dots a_n \qquad | \qquad \rightarrow\!\circ\!\xrightarrow{a_1}\!\circ\!\xrightarrow{a_2}\!\circ\!\cdots\!\xrightarrow{a_n}\!\circ\!\rightarrow \ .$$

▶ **Theorem 3.6.** *For all terms t and s, $[t] = [s]$ iff $\mathsf{RSUB}_{\mathrm{st}} \models t = s$ where*

$$\mathsf{RSUB}_{\mathrm{st}} \triangleq \bigcup_{n \geq 0} \left\{ \mathfrak{v} \in \mathsf{RSUB}_n \;\middle|\; \begin{array}{r} \bigcup_{a \in \mathbf{V}} \mathfrak{v}(a) = \{\langle i-1, i\rangle \mid i \in [1,n]\} \\ \mathfrak{v}(a) \text{ (where } a \text{ ranges over } \mathbf{V}\text{) are disjoint sets} \end{array} \right\}.$$

Proof. By the same construction in the proof of Theorem 3.2, as $\mathsf{RSUB}_{\mathrm{st}}$ is the subclass of RSUB obtained by restricting valuations to the standard language valuation $\{\mathfrak{v}_{\mathrm{st}}\}$. ◀

Figure 1 summarizes the equational theories above where the inclusions are shown by $\mathsf{REL} \subseteq \mathsf{GREL} \supseteq \mathsf{RSUB} \supseteq \mathsf{RSUB}_{\mathrm{st}}$ (and Theorem 3.2) and the non-inclusions are shown by counter-examples. Additionally, note that $\mathrm{EqT}(\{\mathfrak{v}_{\mathrm{st}}\}) = \mathrm{EqT}(\mathsf{GREL})$ for KA [25] and $\mathrm{EqT}(\mathsf{LANG}) = \mathrm{EqT}(\mathsf{GREL})$ for $\mathrm{KA}_{\{\top\}}$ [53, 45].

4 From quantifier-free formulas to equations

In this section, we show that there is a (polynomial-time) reduction from the quantifier-free theory into the equational theory, w.r.t. RSUB. Slightly more generally, we show this characterization for *submodel-closed* classes. We say that a class $\mathcal{C} \subseteq \mathsf{GREL}$ is *submodel-closed* if for all $\mathfrak{v} \in \mathcal{C}$ (on a binary relation U on a set A) and all non-empty subsets $B \subseteq A$, we have $(\mathfrak{v} \restriction B) \in \mathcal{C}$. By definition, RSUB is a submodel-closed class up to isomorphism. Also, REL and GREL are submodel-closed. Additionally, for $\mathfrak{v} \in \mathsf{GREL}$ (on a binary relation U on a set A), we say that a vertex $x \in A$ is *minimal* on $\mathfrak{v}$ if $\langle x, y\rangle \in \hat{\mathfrak{v}}(\top)$ for all $y \in A$ and that a vertex $x \in A$ is *maximal* on $\mathfrak{v}$ if $\langle y, x\rangle \in \hat{\mathfrak{v}}(\top)$ for all $y \in A$. In the following lemma, we have that, to check whether a given equation is valid, it suffices to check for minimal and maximal pairs of vertices.

▶ **Lemma 4.1.** *Let $\mathcal{C} \subseteq \mathsf{GREL}$ be submodel-closed. For all terms t, s, we have: $\mathcal{C} \models t \leq s \ \Leftrightarrow$ $\forall \mathfrak{v} \in \mathcal{C}, \forall l, r$ s.t. l is minimal and r is maximal on $\mathfrak{v}$, $\langle l, r\rangle \notin \hat{\mathfrak{v}}(t) \setminus \hat{\mathfrak{v}}(s)$.*

Proof. ($\Rightarrow$): Trivial. ($\Leftarrow$): We prove the contraposition. Let $\mathfrak{v} \in \mathcal{C}$ (on a binary relation U on a set A), l, and r be s.t. $\langle l, r\rangle \in \hat{\mathfrak{v}}(t) \setminus \hat{\mathfrak{v}}(s)$. Let $B \triangleq \{z \in A \mid \langle l, z\rangle, \langle z, r\rangle \in U\}$. By letting $\mathfrak{v}' \triangleq \mathfrak{v} \restriction B$, we have $\langle l, r\rangle \in \hat{\mathfrak{v}}'(t) \setminus \hat{\mathfrak{v}}'(s)$ $(= (\hat{\mathfrak{v}}(t) \cap B^2) \setminus (\hat{\mathfrak{v}}(s) \cap B^2))$. Hence, this completes the proof. ◀

Next, using minimal vertex l and maximal vertex r, we consider replacing each inequation $u \leq 0$ with $\top u \top \leq 0$, based on that $\mathfrak{v} \models u \leq 0$ iff $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\top u \top)$. More generally, for a quantifier-free formula φ, let $\mathrm{Tr}(\varphi)$ be the $\mathsf{KA}_{\{-\}}$ term defined by:[6]

[6] $\mathrm{Tr}(t = s)$ can be simplified for specific cases, e.g., $\mathrm{Tr}(t \leq s) = \top(t \cap s^{-})\top$ and $\mathrm{Tr}(t \leq 0) = \top t \top$.

$$\mathrm{Tr}(t = s) \triangleq \top((t \cap s^-) + (t^- \cap s))\top, \quad \mathrm{Tr}(\varphi \wedge \psi) \triangleq \mathrm{Tr}(\varphi) + \mathrm{Tr}(\psi), \quad \mathrm{Tr}(\neg\varphi) \triangleq \mathrm{Tr}(\varphi)^-.$$

(For the case of $t = s$, we use the fact $\mathsf{GREL} \models t = s \leftrightarrow (t \cap s^-) + (t^- \cap s) \leq 0$.) We then have the following.

▶ **Lemma 4.2.** *Let $\mathfrak{v} \in \mathsf{GREL}$, l be a minimal vertex on $\mathfrak{v}$, and r be a maximal vertex on $\mathfrak{v}$. For all quantifier-free formulas φ (of $\mathsf{KA}_{\{-\}}$ terms), we have:*

$$\mathfrak{v} \models \varphi \quad \Leftrightarrow \quad \langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\varphi)).$$

Proof. By easy induction on φ. Case $(t = s)$: Let $u = (t \cap s^-) + (t^- \cap s)$. Then $\mathfrak{v} \models t = s$ iff $\hat{\mathfrak{v}}(u) = \emptyset$ iff $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\top u \top)$ iff $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(t = s))$. Case $\psi \wedge \rho$: By $(\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\psi))$ and $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\rho)))$ iff $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\psi) + \mathrm{Tr}(\rho))$. Case $\neg\psi$: By (not $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\psi))$) iff $\langle l, r\rangle \notin \hat{\mathfrak{v}}(\mathrm{Tr}(\psi)^-)$. ◀

▶ **Theorem 4.3.** *Let $\mathcal{C} \subseteq \mathsf{GREL}$ be submodel-closed. For all quantifier-free formulas φ,*

$$\mathcal{C} \models \varphi \quad \Leftrightarrow \quad \mathcal{C} \models \mathrm{Tr}(\varphi) \leq 0.$$

Proof. By Lemmas 4.1 and 4.2. ◀

By the reduction of Theorem 4.3, we have the following complexity results.

▶ **Corollary 4.4.** *The quantifier-free theory w.r.t. RSUB for $\mathsf{KA}_{\{-\}}$ terms is in Π^0_1.*

Proof. By Theorem 4.3 with Corollary 3.4. (The Π^0_1-hardness will be derived from Theorem 4.12.) ◀

▶ **Corollary 4.5.** *The equational theory w.r.t. $\mathsf{REL}/\mathsf{GREL}$ for $\mathsf{KA}_{\{-\}}$ terms is Π^1_1-complete.*

Proof. (Π^1_1-hard): By Theorem 4.3 with that the Horn theory of KA w.r.t. $\mathsf{REL}/\mathsf{GREL}$ is Π^1_1-complete [20]. (In Π^1_1): By the same argument as [20]. ◀

▶ Remark 4.6. In cotrast to Corollary 4.4, the authors do not know the complexity of the quantifier-free theory (resp. Horn theory) w.r.t. LANG for $\mathsf{KA}/\mathsf{KA}_{\{-\}}$ terms, cf. the Horn theory is Π^1_1-complete for $*$-*continuous* KA [27] and for KA w.r.t. $\mathsf{REL}/\mathsf{GREL}$ [20]. (E.g., in the proof of [27], *quotient models* of the standard language valuation are used, but they are not in LANG in general.) ⌟

Also, as a special case of Theorem 4.3, we have the following Hoare hypothesis elimination.

▶ **Corollary 4.7** (Hoare hypothesis elimination)**.** *Let $\mathcal{C} \subseteq \mathsf{GREL}$ be submodel-closed. For all terms t, s, u, we have:*

$$\mathcal{C} \models u \leq 0 \to t \leq s \quad \Leftrightarrow \quad \mathcal{C} \models t \leq s + \top u \top.$$

Proof. By Theorem 4.3 with easy inequations, we have:

$$\begin{aligned}
\mathcal{C} \models u \leq 0 \to t \leq s \quad &\Leftrightarrow \quad \mathcal{C} \models \top(t \cap s^-)\top \leq \top u \top && \text{(By Theorem 4.3)}\\
&\Leftrightarrow \quad \mathcal{C} \models t \cap s^- \leq \top u \top && (\Rightarrow\text{: By } 1 \leq \top \Leftarrow\text{: By } \top\top \leq \top)\\
&\Leftrightarrow \quad \mathcal{C} \models t \leq s + \top u \top. && \blacktriangleleft
\end{aligned}$$

▶ Remark 4.8. Theorem 4.3 and Corollary 4.7 fail w.r.t. LANG; for Corollary 4.7, for instance, we have:

$$\mathsf{LANG} \models xx \leq 0 \to x \leq 0, \qquad \mathsf{LANG} \not\models x \leq \top x x \top.$$

Hence, to use Hoare hypothesis elimination, it is essential to use RSUB instead of LANG. ⌟

▶ Remark 4.9. When $\mathcal{C} = \mathsf{REL}$, we have $\mathcal{C} \models \varphi \leftrightarrow \mathrm{Tr}(\varphi) \leq 0$ (cf. Theorem 4.3) and $\mathcal{C} \models (u \leq 0 \to t \leq s) \leftrightarrow (t \leq s + \top u \top)$ (cf. Corollary 4.7) by the Schröder-Tarski translation [50, XXXII.][19, p. 390, 391]. However, they fail in general when $\mathcal{C}$ is RSUB or GREL. For instance, when $\mathcal{C} = \mathsf{RSUB}$, $t = \top$, $s = 0$, and $u = x$, the second above is equivalent to "$\mathsf{RSUB} \not\models (\neg x \leq 0) \leftrightarrow \top \leq \top x \top$", but this fails; when $\mathfrak{v} \in \mathsf{RSUB}_1$ satisfies $\mathfrak{v}(x) = \{\langle 0,1\rangle\}$, we have $\hat{\mathfrak{v}}(\top) = \{\langle 0,0\rangle, \langle 0,1\rangle, \langle 1,1\rangle\}$ but $\hat{\mathfrak{v}}(\top x \top) = \{\langle 0,1\rangle\}$. This is why we go via "$\langle l, r\rangle \notin \hat{\mathfrak{v}}(\_)$". ⌟

▶ Remark 4.10. We say that a class $\mathcal{C} \subseteq \mathsf{GREL}$ is *$\top$-submodel-closed* if, for all $\mathfrak{v} \in \mathcal{C}$ (on a binary relation U on a set A) and all $\top$-closed non-empty subsets $B \subseteq A$, we have $(\mathfrak{v} \restriction B) \in \mathcal{C}$. By definition, if $\mathcal{C}$ is submodel-closed, thne $\mathcal{C}$ is $\top$-submodel-closed. Lemma 4.1, Theorem 4.3, and Corollary 4.7 can be straight-forwardly generalized for $\top$-submodel-closed classes. ⌟

4.1 Undecidability via Hoare hypothesis elimination

Using Hoare hypothesis elimination w.r.t. RSUB (Corollary 4.7) (see also Remark 4.8), we show the undecidability of the equational theory w.r.t. LANG. The proof can be obtained by the same argument as [34, Lem. 47] by replacing REL with RSUB.

A *context-free grammar* (CFG) $\mathfrak{C}$ over a finite set A is a tuple $\langle X, \mathcal{R}, s\rangle$, where

- X is a finite set of non-terminal labels s.t. $A \cap X = \emptyset$,
- $\mathcal{R}$ is a finite set of rewriting rules $x \leftarrow w$ of $x \in X$ and $w \in (A \cup X)^*$,
- $s \in X$ is the start label.

The relation $x \vdash_{\mathfrak{C}} w$, where $x \in X$ and $w \in A^*$, is defined as the minimal relation closed under the following rule: for all $n \in \mathbb{N}$, $x, x_1, \ldots, x_n \in X$ and $w_0, \ldots, w_n, v_1, \ldots, v_n \in A^*$, if $x \leftarrow w_0 x_1 w_1 \ldots x_n w_n \in \mathcal{R}$, then $\dfrac{x_1 \vdash_{\mathfrak{C}} v_1 \quad \ldots \quad x_n \vdash_{\mathfrak{C}} v_n}{x \vdash_{\mathfrak{C}} w_0 v_1 w_1 \ldots v_n w_n}$. The *language* $[\mathfrak{C}]$ is defined by $[\mathfrak{C}] \triangleq \{w \in A^* \mid s \vdash_{\mathfrak{C}} w\}$. It is well-known that the *universality problem* for CFGs – given a CFG $\mathfrak{C}$, does $[\mathfrak{C}] = A^*$ hold? – is Π^0_1-complete. We can naturally encode this problem by the quantifier-free theory w.r.t. RSUB as follows.

▶ **Lemma 4.11.** *Let $\mathfrak{C} = \langle X, \mathcal{R}, s\rangle$ be a CFG over a finite set $A = \{a_1, \ldots, a_n\}$. Then,*

$$[\mathfrak{C}] = A^* \quad \Leftrightarrow \quad \mathsf{RSUB} \models \Big(\bigwedge_{(x \leftarrow w) \in \mathcal{R}} w \leq x\Big) \to \Big(\Big(\sum_{i=1}^{n} a_i\Big)^* \leq s\Big).$$

Proof. By [34, Lem. 47] with replacing REL with RSUB, because the valuations used in the proof are of the form of RSUB and the operators $\top$ and $\_^-$ do not occur in the formula. (See a full version [35] for an explicit proof.) ◀

▶ **Theorem 4.12.** *The equational theory w.r.t. languages is Π^0_1-complete for $\mathrm{KA}_{\{\overline{x},\cap\}}$.*

Proof. (in Π^0_1): By Corollary 3.4. (Π^0_1-hard): Let $\mathfrak{C} = \langle X, \{x_i \leftarrow w_i \mid i \in [1,m]\}, s\rangle$ be a CFG over a finite set $A = \{a_1, \ldots, a_n\}$. Based on $(\bigwedge_{i=1}^m w_i \leq x_i) \leftrightarrow (\sum_{i=1}^m w_i \cap \overline{x}_i \leq 0)$, by applying the Hoare hypothesis elimination (Corollary 4.7) to Lemma 4.11, we have: $[\mathfrak{C}] = A^*$ iff $\mathsf{RSUB} \models (\sum_{i=1}^n a_i)^* \leq s + \top(\sum_{i=1}^m w_i \cap \overline{x}_i)\top$. Thus, we can give a reduction from the universality problem of CFGs. ◀

Moreover, by the following fact, we can eliminate Kleene-star from Lemma 4.11.

▶ **Proposition 4.13.** $\mathsf{RSUB} \models \overline{1} = x\top \to x^* = \top$.

Proof. Let $n \in \mathbb{N}$ and $\mathfrak{v} \in \mathsf{RSUB}_n$. Let $i \in [1, n]$ be arbitrary. By $\langle i-1, i-1\rangle \notin \hat{\mathfrak{v}}(\overline{1}) = \hat{\mathfrak{v}}(x\top)$, we have $\langle i-1, i-1\rangle \notin \hat{\mathfrak{v}}(x)$. By $\langle i-1, i\rangle \in \hat{\mathfrak{v}}(\overline{1}) = \hat{\mathfrak{v}}(x\top)$, we have $\langle i-1, i\rangle \in \hat{\mathfrak{v}}(x)$ (by $\langle i-1, i-1\rangle \notin \hat{\mathfrak{v}}(x)$). Thus, we have $\hat{\mathfrak{v}}(x^*) = \{\langle i, j\rangle \mid 0 \le i \le j \le n\} = \hat{\mathfrak{v}}(\top)$. ◀

▶ **Lemma 4.14.** *Let $\mathfrak{C} = \langle X, \mathcal{R}, s\rangle$ be a CFG over a finite set $A = \{a_1, \dots, a_n\}$. Then,*

$$[\mathfrak{C}] = A^* \quad \Leftrightarrow \quad \mathsf{RSUB} \models (\overline{1} = (\sum_{i=1}^{n} a_i)\top \wedge \bigwedge_{(x \leftarrow w) \in \mathcal{R}} w \le x) \to (\top \le s).$$

Proof Sketch. By the same argument as Lemma 4.11 with replacing $(\sum_{i=1}^{n} a_i)^*$ with $\top$ using Proposition 4.13 (see [35], for a detail). ◀

Hence, the undecidability above still holds even without Kleene-star.

▶ **Theorem 4.15.** *The equational theory w.r.t. languages is Π^0_1-complete for $\mathrm{KA}_{\{-\}}$ without Kleene-star.*

Proof. By the same way as Theorem 4.12 using Lemma 4.14 instead of with Lemma 4.11. ◀

▶ Remark 4.16. Theorem 4.15 is close to Trakhtenbrot's theorem [52] in first-order logic. By a similar Kleene-star elimination via an encoding of connectivity in finite models [15, p. 30], we can also give a reduction from the universality problem of CFGs into the theory of the finite validity problem of first-order logic (resp. the calculus of relations). (See [35], for a detail.) ⌟

5 Graph characterization for $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$ terms

In Sections 5 and 6, we show that the equational theory w.r.t. languages for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ is decidable and PSPACE-complete. We recall Section 2 for graphs. In this section, we give a graph characterization of the equational theory of RSUB for $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$, by generalizing the graph characterization of REL [34, Thm. 18] (and also [1, 6, 7]). Slightly more generally, we show this characterization for *submodel-closed* classes (Section 4).

5.1 Graph languages for $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$

Let $\tilde{\mathbf{V}} \triangleq \{x, \overline{x} \mid x \in \mathbf{V}\} \cup \{\overline{1}, \top\}$ and $\tilde{\mathbf{V}}_1 \triangleq \tilde{\mathbf{V}} \cup \{1\}$. For a $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$ term t, the graph language $\mathcal{G}(t)$ [1, 7, 34] is a set of graphs over $\tilde{\mathbf{V}}_1$ defined by:[7]

$$\mathcal{G}(x) \triangleq \{ \ \circ\!\!\xrightarrow{x}\!\!\circ \ \} \text{ where } x \in \tilde{\mathbf{V}}, \qquad \mathcal{G}(0) \triangleq \emptyset, \quad \mathcal{G}(1) \triangleq \{ \ \circ\!\!\to\!\!\circ \ \},$$

$$\mathcal{G}(t \cap s) \triangleq \{ \ \circ \overset{G}{\underset{H}{\rightrightarrows}} \circ \ \mid G \in \mathcal{G}(t) \wedge H \in \mathcal{G}(s)\}, \qquad \mathcal{G}(t + s) \triangleq \mathcal{G}(t) \cup \mathcal{G}(s),$$

$$\mathcal{G}(t\,;s) \triangleq \{ \ \circ\!\!\to\!\!G\!\!\to\!\!\circ\!\!\to\!\!H\!\!\to\!\!\circ \ \mid G \in \mathcal{G}(t) \wedge H \in \mathcal{G}(s)\}, \qquad \mathcal{G}(t^*) \triangleq \bigcup_{n \ge 0} \mathcal{G}(t^n).$$

For a valuation $\mathfrak{v} \in \mathsf{GREL}$ on a binary relation on a set B and $\langle x, y\rangle \in \hat{\mathfrak{v}}(\top)$, let $\mathsf{G}(\mathfrak{v}, x, y)$ be the graph defined by: $\mathsf{G}(\mathfrak{v}, x, y) \triangleq \langle B, \{\hat{\mathfrak{v}}(a)\}_{a \in \tilde{\mathbf{V}}_1}, x, y\rangle$. For a class $\mathcal{C} \subseteq \mathsf{GREL}$, let $\mathrm{GR}_{\mathcal{C}}$

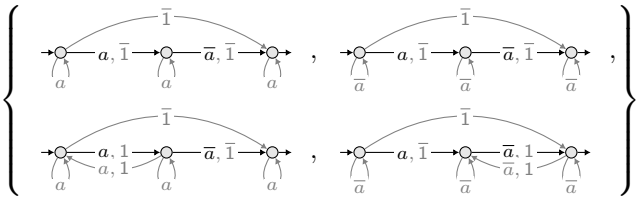
[7] We introduce $\top$-labeled edges, cf. [34, Def. 6], because $\top$ is not fixed to the full relation.

be the graph language $\{\mathsf{G}(\mathfrak{v}, x, y) \mid \mathfrak{v} \in \mathcal{C} \text{ and } \langle x, y\rangle \in \hat{\mathfrak{v}}(\top)\}$. Note that if $\mathcal{C} \subseteq \mathsf{GREL}$ is submodel-closed, then $\mathrm{GR}_{\mathcal{C}}$ is induced subgraph-closed (i.e., every induced subgraph of every $G \in \mathcal{G}$ is isomorphic to a member of $\mathcal{G}$).

We recall edge-saturations $\mathcal{S}(h)$ of Definition 2.1. For a graph G and graph language $\mathcal{G}$, let

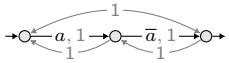
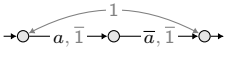
$$\mathcal{S}_{\mathcal{C}}(G) \triangleq \{\mathcal{S}(h) \mid \exists H \in \mathrm{GR}_{\mathcal{C}}, h\colon G \longrightarrow H\}, \qquad \mathcal{S}_{\mathcal{C}}(\mathcal{G}) \triangleq \bigcup_{H \in \mathcal{G}} \mathcal{S}_{\mathcal{C}}(H).$$

▶ **Example 5.1.** The following is an instance of $\mathcal{S}_{\mathsf{RSUB}}(G)$ where $\mathbf{V} = \{a\}$:

$\mathcal{S}_{\mathsf{RSUB}}($ $) =$ 

(Here, gray-colored edges are the edges extended by edge-saturations $\mathcal{S}_{\mathsf{RSUB}}$. We omit unimportant edges.)

For instance, the below right graph above can be obtained from the following map:

Note that, for instance, $\mathcal{S}_{\mathsf{RSUB}}($ $)$ does not contain graphs of the forms  nor  because we cannot make a graph homomorphism into graphs of RSUB. ⌟

By the form of $\mathrm{GR}_{\mathsf{RSUB}}$, each graph $H \in \mathcal{S}_{\mathsf{RSUB}}(\_)$ satisfies the following:

- $\top^H$ is a total preorder (possibly not a total order);
- $a^H \supseteq 1^H$ or $\overline{a}^H \supseteq 1^H$ holds for each $a \in \mathbf{V}$.

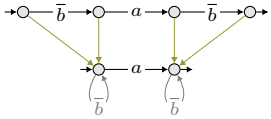
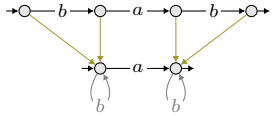
Let $H^{\mathcal{Q}} \triangleq H/(1^H)^=$ and $\mathcal{G}^{\mathcal{Q}} \triangleq \{H^{\mathcal{Q}} \mid H \in \mathcal{G}\}$ where $R^=$ denotes the equivalence closure of R. We then have the following graph language characterization, which is an analog of [34, Thm. 18], but is slightly generalized for including RSUB (see [35], for an explicit proof).

▶ **Theorem 5.2.** *Let $\mathcal{C} \subseteq \mathsf{GREL}$ be submodel-closed. For all $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$ terms t, s,*

$$\mathcal{C} \models t \leq s \quad \Leftrightarrow \quad \forall H \in \mathcal{S}_{\mathcal{C}}(\mathcal{G}(t))^{\mathcal{Q}}, \exists G \in \mathcal{G}(s), G \longrightarrow H.$$

▶ **Example 5.3.** (We recall the inequations in Remark 3.5.) Here are examples to show $\mathrm{KA}_{\{\overline{x},\overline{1},\top,\cap\}}$ equations on RSUB using Theorem 5.2. (Gray-colored edges are the edges extended by edge-saturations $\mathcal{S}_{\mathsf{RSUB}}$. We omit unimportant edges.)

- $\mathsf{LANG} \models a \leq \overline{b}a\overline{b} + bab$: This equation is shown by the following graph homomorphisms:

$\mathcal{G}(\overline{b}a\overline{b} + bab) = \{$  ,  $\}$

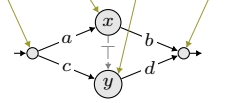
$\mathcal{S}_{\mathsf{RSUB}}(\mathcal{G}(a))^{\mathcal{Q}} \ni H$:

(Case $\overline{b}^H \supseteq 1^H$) (Case $b^H \supseteq 1^H$)

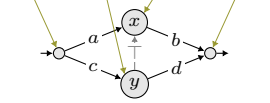
- $\mathsf{LANG} \models ab \cap cd \leq a\top d + c\top b$: For each graph $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathcal{G}(ab \cap cd))^{\mathcal{Q}}$, we can give a graph homomorphism from some graph in $\mathcal{G}(a\top d + c\top b)$ as follows:

$\mathcal{G}(a\top d + c\top b) = \{$  ,  $\}$

$\mathcal{S}_{\mathsf{RSUB}}(\mathcal{G}(ab \cap cd))^{\mathcal{Q}} \ni H$:

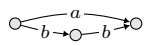


(Case $\langle x, y \rangle \in \top^H$)



(Case $\langle x, y \rangle \in \top^H$)

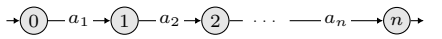
Additionally, note that $\_^{\mathcal{Q}}$ is necessary in general, e.g., for $\top \leq 1 + \overline{1}$ [34, Remark 19]. ⌟

▶ Remark 5.4. Theorem 5.2 fails when $\mathcal{C}$ is not submodel-closed. E.g., if $\mathcal{C}$ consists of the one valuation given by  , $t = a$, and $s = bb$, then $\mathcal{C} \models t \leq s$ holds but the right-hand side does not hold. ⌟

5.2 Word languages for $\mathbf{KA}_{\{\overline{x},\overline{1},\top\}}$

Particularly for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$, Theorem 5.2 can be rephrased by *word* languages.

For a word $w = a_1 \dots a_n$ over $\tilde{\mathbf{V}}$, let $\mathsf{G}(w)$ be the following graph where $|\mathsf{G}(w)| = [0, n]$:



$\mathsf{G}(w)$ is the unique graph in $\mathcal{G}(w)$ up to graph isomorphisms.

For a $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ term t, we write $[t]_{\tilde{\mathbf{V}}}$ for the word language $[t]$ over $\tilde{\mathbf{V}}$ (namely, $\overline{x}$, $\overline{1}$, and $\top$ are also viewed as letters); e.g., $[\overline{a}]_{\tilde{\mathbf{V}}} = \{\overline{a}\}$ and $[\overline{a}] = \mathbf{V}^* \setminus \{a\}$ for $a \in \mathbf{V}$. Note that $\mathcal{G}(t) = \{\mathsf{G}(w) \mid w \in [t]_{\tilde{\mathbf{V}}}\}$; thus, for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ terms, graph languages are expressible by using word languages.

Additionally, we introduce nondeterministic finite word automata with epsilon transitions (*NFAs*). NFAs are (2-pointed) graphs over $\tilde{\mathbf{V}}_1$ where the source and target vertices denote the initial and (single) accepting states, respectively, and 1-labeled edges denote epsilon transitions. For a graph H and a word $w = a_1 \dots a_n$, we write δ_w^H for the binary relation $(1^H)^*; a_1^H; (1^H)^*; \dots; a_n^H; (1^H)^*$. For $q \in |H|$, we let $\delta_w^H(q) \triangleq \{q' \mid \langle q, q' \rangle \in \delta_w^H\}$. For $Q \subseteq |H|$, we let $\delta_w^H(Q) \triangleq \bigcup_{q \in Q} \delta_w^H(q)$. The word language $[H]_{\tilde{\mathbf{V}}}$ is defined as $\{w \in \tilde{\mathbf{V}}^* \mid \langle 1^H, 2^H \rangle \in \delta_w^H\}$. Note that $[H]_{\tilde{\mathbf{V}}} = \{w \in \tilde{\mathbf{V}}^* \mid \mathsf{G}(w) \longrightarrow H^{\mathcal{Q}}\}$ if 1^H is an equivalence relation. We then have the following, which a rephrasing of Theorem 5.2 (see Section B for an explicit proof). This shows that $\mathsf{RSUB} \models t \leq s$ is equivalent to that every NFA obtained from a word w of t by an edge-saturation w.r.t. RSUB has an intersection with $[s]_{\tilde{\mathbf{V}}}$.

▶ **Corollary 5.5.** *Let $\mathcal{C} \subseteq \mathsf{GREL}$ be submodel-closed. For all $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ terms t and s,*

$$\mathcal{C} \models t \leq s \quad \Leftrightarrow \quad [t]_{\tilde{\mathbf{V}}} \subseteq \{w \in \tilde{\mathbf{V}}^* \mid \forall H \in \mathcal{S}_{\mathcal{C}}(\mathsf{G}(w)), [s]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \neq \emptyset\}.$$

▶ **Example 5.6.** Here are examples to show $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ equations on RSUB using Corollary 5.5.

- $\mathsf{LANG} \models a \leq \overline{b}a\overline{b} + bab$ (the first example in Example 5.3): For all NFAs $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(a))$, we have $[\overline{b}a\overline{b} + bab]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \ni \begin{cases} \overline{b}a\overline{b} & (\overline{b}^H \supseteq 1^H) \\ bab & (b^H \supseteq 1^H) \end{cases}$ by the following paths:

(Case $\overline{b}^H \supseteq 1^H$) (Case $b^H \supseteq 1^H$)

- $\mathsf{LANG} \models a \leq \overline{1} + aa$ [34, (3)]: For all NFAs $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(a))$, we have $[\overline{1} + aa]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \ni \begin{cases} \overline{1} & (1^H = \Delta_{|H|}) \\ aa & (1^H = \top^H) \end{cases}$ by the following paths:

(Case $1^H = \Delta_{|H|}$) (Case $1^H = \top^H$)

- $\mathsf{LANG} \models \overline{1}a\overline{a}\overline{1} \leq \overline{1}\overline{a}a\overline{1}$ [39]: For all NFAs $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(\overline{1}a\overline{a}\overline{1}))$, we have $[\overline{1}\overline{a}a\overline{1}]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \ni \overline{1}\overline{a}a\overline{1}$ in either $\overline{a}^H \supseteq 1^H$ or $a^H \supseteq 1^H$ by the following paths:

(Case $\overline{a}^H \supseteq 1^H$) (Case $a^H \supseteq 1^H$)

Next, we use the NFA characterization of Corollary 5.5 for an automata construction.

6 PSPACE decidability for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ terms

In this section, based on the graph characterization (Section 5), we present an NFA construction for deciding the equational theory for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ terms. Here, we will use NFAs (graphs over $\tilde{\mathbf{V}}_1$) instead of $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ terms (regular expressions over the alphabet $\tilde{\mathbf{V}}$). To be more precise, relying on the graph characterization (Corollary 5.5), we consider the following: given an NFA J (having the same language as the term s in Corollary 5.5), we construct an NFA recognizing the following word language:

$$\mathsf{L}_J \triangleq \{w \in \tilde{\mathbf{V}}^* \mid \exists H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(w)), [J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset\}.$$

Note that $\mathsf{RSUB} \models t \leq s \Leftrightarrow [t]_{\tilde{\mathbf{V}}} \cap \mathsf{L}_J = \emptyset$ when $[s]_{\tilde{\mathbf{V}}} = [J]_{\tilde{\mathbf{V}}}$. We first present an equivalent notion of "$w \in \mathsf{L}_J$" in Section 6.1, and then we give an NFA construction in Section 6.2. Our approach in this section is based on [34] where we consider RSUB instead of REL.

6.1 Saturable paths for RSUB

We first give an equivalent notion of $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$ in the definition of L_J.

▶ **Definition 6.1.** *Let J and H be NFAs. A map $U\colon |H| \to \wp(|J|)$ is an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$ if the following hold where $U_x \triangleq U(x)$:*

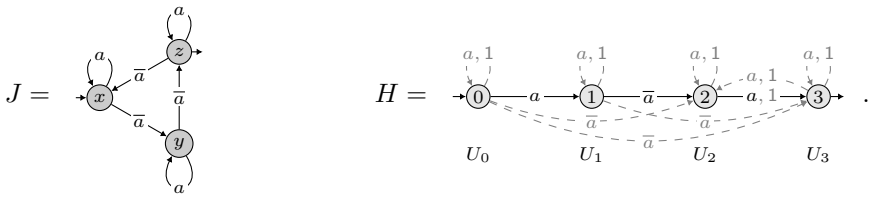
- $1^J \in U_{1^H}$ *and* $\forall a \in \tilde{\mathbf{V}}_1, \forall \langle x, y\rangle \in a^H$, $\delta_a^J(U_x) \subseteq U_y$,
- $2^J \notin U_{2^H}$. ⌟

Intuitively, the first condition denotes that U is a cover of the reachable states from the pair "$1^J \in U_{1^H}$". If the second condition holds, we can see that the pair "$2^J \in U_{2^H}$" is unreachable. As expected, we have the following (see Section C, for a proof).

▶ **Proposition 6.2.** *Let J and H be NFAs where 1^H is reflexive. Then, we have:*

$$[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset \quad \Leftrightarrow \quad \exists U\colon |H| \to \wp(|J|),\ U \text{ is an emptiness-witness for } [J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset.$$

▶ **Example 6.3.** We consider the following NFAs J and H. The NFA J satisfies $[J]_{\tilde{\mathbf{V}}} = \{w \in \{a, \overline{a}\}^* \mid \exists n \in \mathbb{N}, \overline{a} \text{ occurs } 3n+2 \text{ times in } w\}$ and the NFA H is a graph in $\mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(a\overline{a}a))$, where $\top$- or $\overline{1}$-labeled edges are omitted, and gray-colored edges are the edges edge-saturated from the graph $\mathsf{G}(a\overline{a}a)$. From the form of H, one can see that $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$.



If $U_0 = U_1 = \{\textcircled{x}\}$ and $U_2 = U_3 = \{\textcircled{y}\}$, then this U is an emptiness-witness; e.g., for $\langle 1,2\rangle \in \overline{a}^H$, $\delta_{\overline{a}}^J(U_1) = \{\textcircled{y}\} \subseteq U_2$. By the witnesses, we have $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$. Besides this, if $U_0 = U_1 = \{\textcircled{x}\}$ and $U_2 = U_3 = \{\textcircled{x}, \textcircled{y}\}$, then this U is also an emptiness-witness; so, U may not coincide with the reachable states from the pair "$1^J \in U_{1^H}$". ⌟

Next, we give an equivalent notion of "$w \in \mathsf{L}_J$", by forgetting saturated edges (gray-colored edges in Example 6.3) using "U" of Proposition 6.2.

▶ **Definition 6.4.** *Let J be a NFA and w be a word. A pair $P = \langle H, U\rangle$ is a saturable path for $w \in \mathsf{L}_J$ if the following hold:*

(P-Ext) *H is an edge-extension[8] of $\mathsf{G}(w)$ such that*
- *$\top^H$ is a total preorder and $\top^H \supseteq \{\langle i-1, i\rangle \mid i \in [1,n]\}$ where $w = a_1 \dots a_n$,*
- *$1^H = \top^H \cap \{\langle j, i\rangle \mid \langle i, j\rangle \in \top^H\}$ and $\overline{1}^H = \top^H \setminus 1^H$,*
- *$\forall a \in \mathbf{V}$, $\langle a^H, \overline{a}^H\rangle$ is either $\langle a^{\mathsf{G}(w)} \cup 1^H, \overline{a}^{\mathsf{G}(w)}\rangle$ or $\langle a^{\mathsf{G}(w)}, \overline{a}^{\mathsf{G}(w)} \cup 1^H\rangle$.*

(P-Con) *H is consistent: $\forall a \in \mathbf{V}$, $a^{H^{\mathcal{Q}}} \cap \overline{a}^{H^{\mathcal{Q}}} = \emptyset$.*

(P-Wit) *$U\colon |H| \to \wp(|J|)$ is an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$.*

(P-Sat) *H is saturable: $\forall a \in \mathbf{V}, \forall \langle i, j\rangle \in \overline{1}^H$, $\delta_a^J(U_i) \subseteq U_j$ or $\delta_{\overline{a}}^J(U_i) \subseteq U_j$.* ⌟

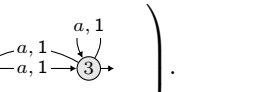
Then, as expected, the existence of saturable path can characterize "$w \in \mathsf{L}_J$".

▶ **Lemma 6.5** (Section D). *Let J be a NFA and w be a word. Then,*

$$w \in \mathsf{L}_J \quad \Leftrightarrow \quad \textit{there is a saturable path for } w \in \mathsf{L}_J.$$

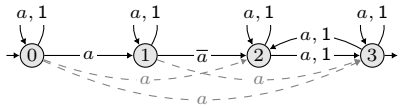
▶ **Example 6.6.** We recall the NFAs J and $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(a\overline{a}a))$ in Example 6.3. The following P is a saturable path for $a\overline{a}a \in \mathsf{L}_J$ where $\top$- or $\overline{1}$-labeled edges are omitted:

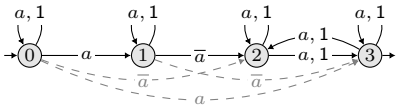
$$P \quad = \quad \left(\begin{array}{c} \text{0} \xrightarrow{a} \text{1} \xrightarrow{\overline{a}} \text{2} \\ \{\textcircled{x}\} \quad \{\textcircled{x}\} \quad \{\textcircled{x}, \textcircled{y}\} \end{array} \right.$$



$$\left. \begin{array}{c} \{\textcircled{x}, \textcircled{y}\} \\ \text{3} \end{array} \right).$$

(P is of the form of a *path graph* by taking the quotient graph w.r.t. 1-labeled edges.) P is an abstraction of edge-saturated graphs. From P, we can construct a graph $H \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(a\overline{a}a))$ s.t. $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$. Because both $\delta_{\overline{a}}^J(\{\textcircled{x}\}) \subseteq \{\textcircled{x}, \textcircled{y}\}$ and $\delta_a^J(\{\textcircled{x}\}) \subseteq \{\textcircled{x}, \textcircled{y}\}$ hold, in addition to the graph H in Example 6.3, for instance, the following are also possible edge-saturated graphs:





. ⌟

By using saturable paths, we can replace the existence of such gray-colored edges connecting distant vertices with a "locally" defined witness U. This rephrasing will be useful for our automata construction.

To give an NFA construction, let

$$\varphi^J(\mathcal{U}, U) \triangleq \forall a \in \mathbf{V}, \forall \langle u, u'\rangle \in \mathcal{U}, \delta_a^J(u) \subseteq U \vee \delta_{\overline{a}}^J(u') \subseteq U$$

and we also replace (P-Sat) with a "local" condition.

---

[8] In this definition, $\top^H$-, 1^H-, and $\overline{1}^H$-edges are edge-saturated and a- and $\overline{a}$-edges in 1^H (for $a \in \mathbf{V}$) are also edge-saturated. This is for preserving (P-Con) easily.

▶ **Proposition 6.7.** *Let J and H be graphs. Let $i \in |H|$. Then we have:*

$$(\forall a \in \mathbf{V}, \forall j \text{ s.t. } \langle j,i\rangle \in \overline{1}^H, \delta_a^J(U_j) \subseteq U_i \vee \delta_{\overline{a}}^J(U_j) \subseteq U_i) \quad \Leftrightarrow \quad \varphi^J(\bigcup_{j;\langle j,i\rangle\in\overline{1}^H} U_j^2, U_i).$$

Proof. For each i,j, we have: $(\forall a \in \mathbf{V}, \delta_a^J(U_j) \subseteq U_i \vee \delta_{\overline{a}}^J(U_j) \subseteq U_i)$ iff $(\forall a \in \mathbf{V}, (\forall u \in U_j, \delta_a^J(u) \subseteq U_i) \vee (\forall u' \in U_j, \delta_{\overline{a}}^J(u') \subseteq U_i))$ iff $\varphi^J(U_j^2, U_i)$ (by taking the prenex normal form). By $(\forall j \text{ s.t. } \langle j,i\rangle \in \overline{1}^H, \varphi^J(U_j^2, U_i))$ iff $\varphi^J(\bigcup_{j;\langle j,i\rangle\in\overline{1}^H} U_j^2, U_i)$, this completes the proof. ◀

6.2 Automata from saturable paths

Let $\mathcal{X} \triangleq \{X \in \wp(\tilde{\mathbf{V}}_1) \mid 1, \top \in X,\ \overline{1} \notin X, \text{ and } \forall x \in \mathbf{V}, x \in X \leftrightarrow \overline{x} \notin X\}$. (This set is equivalent to the set $\{\{x \in \tilde{\mathbf{V}}_1 \mid 1^H \subseteq x^H\} \mid H \in \mathrm{GR}_{\mathsf{RSUB}}\}$.)

▶ **Definition 6.8** (NFA construction). *Let ▶ and ◀ be two fresh symbols. For a graph J and a set $X \in \mathcal{X}$, let $J^{\mathcal{S}_X}$ be the graph G defined as follows:*

- $|G| \triangleq \{\blacktriangleright, \blacktriangleleft\} \cup Q$ *where* $Q \triangleq \{\langle \mathcal{U}, U\rangle \in \wp(|J|^2) \times \wp(|J|) \mid \varphi^J(\mathcal{U}, U) \wedge \forall x \in X, \delta_x^J(U) \subseteq U\}$,
- $1^G \triangleq (\{\blacktriangleright\} \times \{\langle \mathcal{U}, U\rangle \in Q \mid 1^J \in U \wedge \mathcal{U} = \emptyset\}) \cup (\{\langle \mathcal{U}, U\rangle \in Q \mid 2^J \notin U\} \times \{\blacktriangleleft\})$,
- $x^G \triangleq \{\langle \langle \mathcal{U}, U\rangle, \langle \mathcal{U}', U'\rangle\rangle \in Q^2 \mid \psi^X_{x,\overline{1}}(\mathcal{U}, U, \mathcal{U}', U') \vee \psi^X_{x,1}(\mathcal{U}, U, \mathcal{U}', U')\}$ *for* $x \in \tilde{\mathbf{V}}$,
- $1^G \triangleq \blacktriangleright$,
- $2^G \triangleq \blacktriangleleft$.

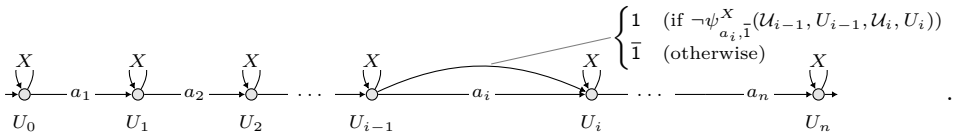
Here, $\psi^X_{x,\overline{1}}(\mathcal{U}, U, \mathcal{U}', U')$ and $\psi^X_{x,1}(\mathcal{U}, U, \mathcal{U}', U')$ are defined as follows:

- $\psi^X_{x,\overline{1}}(\mathcal{U}, U, \mathcal{U}', U') \quad \triangleq \quad \left(\mathcal{U}' = \mathcal{U} \cup U^2 \wedge \bigwedge \left\{ \begin{array}{r} \delta_x^J(U) \subseteq U', \\ \delta_\top^J(\{u \mid \langle u,u\rangle \in \mathcal{U}'\}) \subseteq U', \\ \delta_{\overline{1}}^J(\{u \mid \langle u,u\rangle \in \mathcal{U}'\}) \subseteq U' \end{array} \right\}\right)$,
- $\psi^X_{x,1}(\mathcal{U}, U, \mathcal{U}', U') \quad \triangleq \quad (\mathcal{U}' = \mathcal{U} \wedge U' = U \wedge x \in X)$. ⌟

By the form of $J^{\mathcal{S}_X}$, if $a_1 \dots a_n \in [J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}$, then its run is of the following form:

$$\to \blacktriangleright \xrightarrow{1} \langle \mathcal{U}_0, U_0\rangle \xrightarrow{a_1} \langle \mathcal{U}_1, U_1\rangle \xrightarrow{a_2} \langle \mathcal{U}_2, U_2\rangle \longrightarrow \cdots \longrightarrow \xrightarrow{a_n} \langle \mathcal{U}_n, U_n\rangle \xrightarrow{1} \blacktriangleleft \to .$$

Intuitively, this run corresponds to the following saturable path where some $\top$-, $\overline{1}$-, or 1-labeled edges are omitted and $\overset{X}{\circlearrowleft}$ denotes x-labeled edges for $x \in X$:



Here, $\mathcal{U}_i$ is used to denote the set $\bigcup_{j;\langle j,i\rangle\in\overline{1}^H} U_j^2$ (cf. Proposition 6.7) where H is the graph of the saturable path above. Additionally, we have $\psi^X_{a_i,\overline{1}}(\mathcal{U}_{i-1}, U_{i-1}, \mathcal{U}_i, U_i)$ if $\langle i-1, i\rangle \in \overline{1}^H$ and we have $\psi^X_{a_i,1}(\mathcal{U}_{i-1}, U_{i-1}, \mathcal{U}_i, U_i)$ if $\langle i-1, i\rangle \in 1^H$ by construction. Based on this correspondence, from a word $w \in \bigcup_{X\in\mathcal{X}}[J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}$, we can construct a saturable path for $w \in \mathsf{L}_J$, and conversely, from a saturable path for $w \in \mathsf{L}_J$, we can show $w \in \bigcup_{X\in\mathcal{X}}[J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}$ (see Section E, for details). Thus we have the following.

▶ **Lemma 6.9** (Section E). *Let J be a graph. Then we have $\mathsf{L}_J = \bigcup_{X\in\mathcal{X}}[J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}$.*

▶ **Theorem 6.10.** *The equational theory w.r.t. languages for $\mathrm{KA}_{\{\overline{x},\overline{1},\top\}}$ is PSPACE-complete.*

Table 1 Summary of our complexity results for equational theories w.r.t. languages, with comparison to other semantics.

	KA	$\mathsf{KA}_{\{\overline{x}\}}$	$\mathsf{KA}_{\{\overline{1}\}}$	$\mathsf{KA}_{\{\overline{x},\overline{1}\}}$	$\mathsf{KA}_{\{\cap\}}$
LANG RSUB	PSPACE-c [25]	PSPACE-c (Theorem 6.10)			EXPSPACE-c [4]
$\{\mathfrak{v}_{\mathrm{st}}\}$	PSPACE-c [31]				EXPSPACE-c [18]
REL	PSPACE-c [25]	PSPACE-c [34]		in coNEXP [34]	EXPSPACE-c [6, 7, 32, 37]

	$\mathsf{KA}_{\{\overline{1},\cap\}}$	$\mathsf{KA}_{\{\overline{x},\cap\}}$	$\mathsf{KA}_{\{\overline{x},\overline{1},\cap\}}$	$\mathsf{KA}_{\{-\}}$
LANG RSUB	(open)	Π^0_1-c (Theorem 4.12 and Corollary 3.4)		
$\{\mathfrak{v}_{\mathrm{st}}\}$	EXPSPACE-c [18]			TOWER-c [49, 47]
REL	Π^0_1-c [36]	Π^0_1-c [34]		Π^1_1-c [20] (Corollary 4.5)

Proof. (in PSPACE): Let t and s be $\mathsf{KA}_{\{\overline{x},\overline{1},\top\}}$ terms. Let G and J be NFAs s.t. $[G]_{\tilde{\mathbf{V}}} = [t]_{\tilde{\mathbf{V}}}$ and $[J]_{\tilde{\mathbf{V}}} = [s]_{\tilde{\mathbf{V}}}$. By Corollary 5.5 and Lemma 6.9, we have: $\mathsf{RSUB} \models t \leq s \Leftrightarrow [G]_{\tilde{\mathbf{V}}} \cap \mathsf{L}_J = \emptyset \Leftrightarrow [G]_{\tilde{\mathbf{V}}} \cap (\bigcup_{X \in \mathcal{X}} [J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}) = \emptyset$. Thus we can reduce the equational theory into the emptiness problem of NFAs of size exponential to the size of the input inequation, where we use the union construction for $\cup$ and the product construction for $\cap$ in NFAs. In this reduction, using a standard on-the-fly algorithm for the non-emptiness problem of NFAs (essentially the graph reachability problem), we can give a non-deterministic polynomial space algorithm. (Note that the membership of "$a \in |J^{\mathcal{S}_X}|$" and "$\langle a, b\rangle \in x^{J^{\mathcal{S}_X}}$" for each $x \in \tilde{\mathbf{V}}_1$ can be easily determined in polynomial space; so, we can construct such an on-the-fly algorithm indeed.) (Hardness): The equational theory of KA w.r.t. languages coincides with the language equivalence problem of regular expressions (Remark 2.3), which is PSPACE-complete [31]. Hence, the equational theory of $\mathsf{KA}_{\{\overline{x},\overline{1},\top\}}$ is PSPACE-hard. ◀

▶ Remark 6.11. W.r.t. REL, it is open the complexity of the equational theory for $\mathsf{KA}_{\{\overline{x},\overline{1},\top\}}$ [34, Remark 45]. W.r.t. RSUB, each equivalence class induced from 1-labeled edges is always an interval; so, the problematic case of [34, Remark 45] (w.r.t. REL) does not appear in Theorem 6.10 (w.r.t. RSUB). ⌟

7 Conclusion and Future directions

We have introduced RSUB for the equational theory w.r.t. languages for $\mathsf{KA}_{\{-\}}$ terms. Using RSUB, we have shown some complexity results for the equational theory w.r.t. languages for fragments of $\mathsf{KA}_{\{-\}}$ terms (Table 1). We leave open the decidability and complexity of the equational theory w.r.t. languages for $\mathsf{KA}_{\{\overline{1},\cap\}}$ (cf. Remark 6.11). A natural interest is to consider variants or fragments of $\mathrm{KA}_{\{-\}}$, e.g., with reverse [3], with tests [29] (by considering guarded strings) or with (anti-)domain [13]. It would also be interesting to consider the combination of variables and letters (cf. Theorems 3.2 and 3.6) in the context of language/string constraints.

Additionally, to separate the expressive power w.r.t. languages, it would also be interesting to consider games like Ehrenfeucht-Fraïssé games [16, 17] on RSUB, cf., e.g., on REL for the calculus of relations [33] and on languages for star-free expressions [51].

References

1 Hajnal Andréka and D. A. Bredikhin. The equational theory of union-free algebras of relations. *Algebra Universalis*, 33(4):516–532, 1995. doi:10.1007/BF01225472.

2 Hajnal Andréka, Szabolcs Mikulás, and István Németi. The equational theory of Kleene lattices. *Theoretical Computer Science*, 412(52):7099–7108, 2011. doi:10.1016/J.TCS.2011.09.024.

3 S. L. Bloom, Z. Ésik, and Gh. Stefanescu. Notes on equational theories of relations. *algebra universalis*, 33(1):98–126, 1995. doi:10.1007/BF01190768.

4 Paul Brunet. Reversible Kleene lattices. In *MFCS*, volume 83 of *LIPIcs*, pages 66:1–66:14. Schloss Dagstuhl, 2017. doi:10.4230/LIPICS.MFCS.2017.66.

5 Paul Brunet. A complete axiomatisation of a fragment of language algebra. In *CSL*, volume 152 of *LIPIcs*, pages 11:1–11:15. Schloss Dagstuhl, 2020. doi:10.4230/LIPIcs.CSL.2020.11.

6 Paul Brunet and Damien Pous. Petri automata for Kleene allegories. In *LICS*, pages 68–79. IEEE, 2015. doi:10.1109/LICS.2015.17.

7 Paul Brunet and Damien Pous. Petri automata. *Logical Methods in Computer Science*, 13(3), 2017. doi:10.23638/LMCS-13(3:33)2017.

8 Wojciech Buszkowski. On the complexity of the equational theory of relational action algebras. In *RAMICS*, volume 4136 of *LNTCS*, pages 106–119. Springer, 2006. doi:10.1007/11828563_7.

9 Ernie Cohen. Hypotheses in Kleene algebra. *Unpublished manuscript*, 1994.

10 Rina S. Cohen and J. A. Brzozowski. Dot-depth of star-free events. *Journal of Computer and System Sciences*, 5(1):1–16, 1971. doi:10.1016/S0022-0000(71)80003-X.

11 John H. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.

12 Isabel F. Cruz, Alberto O. Mendelzon, and Peter T. Wood. A graphical query language supporting recursion. *ACM SIGMOD Record*, 16(3):323–330, 1987. doi:10.1145/38714.38749.

13 Jules Desharnais, Bernhard Möller, and Georg Struth. Kleene algebra with domain. *ACM Transactions on Computational Logic*, 7(4):798–833, 2006. doi:10.1145/1183278.1183285.

14 Amina Doumane, Denis Kuperberg, Damien Pous, and Pierre Pradic. Kleene algebra with hypotheses. In *FoSSaCS*, volume 11425 of *LNTCS*, pages 207–223. Springer, 2019. doi:10.1007/978-3-030-17127-8_12.

15 Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite Model Theory*. Springer, 1995. doi:10.1007/3-540-28788-4.

16 Andrzej Ehrenfeucht. An application of games to the completeness problem for formalized theories. *Fundamenta Mathematicae*, 49(2):129–141, 1961. doi:10.4064/fm-49-2-129-141.

17 Roland Fraïssé. Sur les classifications des systems de relations. *Publ. Sci. Univ. Alger I.*, 1954.

18 Martin Fürer. The complexity of the inequivalence problem for regular expressions with intersection. In *ICALP*, volume 85 of *LNCS*, pages 234–245. Springer, 1980. doi:10.1007/3-540-10003-2_74.

19 Steven Givant. The calculus of relations. In *Introduction to Relation Algebras*, volume 1, pages 1–34. Springer International Publishing, 2017. doi:10.1007/978-3-319-65235-1_1.

20 Chris Hardin and Dexter Kozen. On the complexity of the Horn theory of REL. Technical report, Cornell University, 2003. URL: https://hdl.handle.net/1813/5612.

21 Jelle Hellings, Catherine L. Pilachowski, Dirk Van Gucht, Marc Gyssens, and Yuqing Wu. From relation algebra to semi-join algebra: An approach to graph query optimization. *The Computer Journal*, 64(5):789–811, 2021. doi:10.1093/comjnl/bxaa031.

22 Robin Hirsch and Ian Hodkinson. *Relation Algebras by Games*, volume 147 of *Studies in logic and the foundations of mathematics*. Elsevier, 1 edition, 2002.

23 Tony Hoare, Bernhard Möller, Georg Struth, and Ian Wehrman. Concurrent Kleene algebra and its foundations. *The Journal of Logic and Algebraic Programming*, 80(6):266–296, 2011. doi:10.1016/j.jlap.2011.04.005.

24 Stephen C. Kleene. Representation of events in nerve nets and finite automata. In *Automata Studies. (AM-34)*, pages 3–42. Princeton University Press, 1956. doi:10.1515/9781400882618-002.

25 Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. In *LICS*, pages 214–225. IEEE, 1991. doi:10.1109/LICS.1991.151646.

26 Dexter Kozen. On Hoare logic and Kleene algebra with tests. *ACM Transactions on Computational Logic*, 1(1):60–76, 2000. doi:10.1145/343369.343378.

27 Dexter Kozen. On the complexity of reasoning in Kleene algebra. *Information and Computation*, 179(2):152–162, 2002. doi:10.1006/INCO.2001.2960.

28 Dexter Kozen and Konstantinos Mamouras. Kleene algebra with equations. In *ICALP*, volume 8573 of *LNCS*, pages 280–292. Springer, 2014. doi:10.1007/978-3-662-43951-7_24.

29 Dexter Kozen and Frederick Smith. Kleene algebra with tests: Completeness and decidability. In *CSL*, volume 1258 of *LNCS*, pages 244–259. Springer, 1996. doi:10.1007/3-540-63172-0_43.

30 F. W. Levi. On semigroups. *Bulletin of the Calcutta Mathematical Society*, 36(36):141–146, 1944.

31 A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *SWAT*, pages 125–129. IEEE, 1972. doi:10.1109/SWAT.1972.29.

32 Yoshiki Nakamura. Partial derivatives on graphs for Kleene allegories. In *LICS*, pages 1–12. IEEE, 2017. doi:10.1109/LICS.2017.8005132.

33 Yoshiki Nakamura. Expressive power and succinctness of the positive calculus of binary relations. *Journal of Logical and Algebraic Methods in Programming*, 127:100760, 2022. doi:10.1016/j.jlamp.2022.100760.

34 Yoshiki Nakamura. Existential calculi of relations with transitive closure: Complexity and edge saturations. In *LICS*, pages 1–13. IEEE, 2023. doi:10.1109/LICS56636.2023.10175811.

35 Yoshiki Nakamura. Finite relational semantics for language Kleene algebra with complement, 2024. URL: https://hal.science/hal-04455882.

36 Yoshiki Nakamura. Undecidability of the positive calculus of relations with transitive closure and difference: Hypothesis elimination using graph loops. In *RAMICS*, volume 14787 of *LNTCS*, pages 207–224. Springer, 2024. doi:10.1007/978-3-031-68279-7_13.

37 Yoshiki Nakamura. Derivatives on graphs for the positive calculus of relations with transitive closure, 2024 (submitted, journal version of [32]). doi:10.48550/arXiv.2408.08236.

38 Yoshiki Nakamura and Ryoma Sin'ya. Words-to-letters valuations for language Kleene algebras with variable complements. In *AFL*, volume 386 of *EPTCS*, pages 185–199. EPTCS, 2023. doi:10.4204/EPTCS.386.15.

39 Yoshiki Nakamura and Ryoma Sin'ya. Words-to-letters valuations for language Kleene algebras with variable and constant complements, 2024 (accepted, journal version of [38]). URL: https://arxiv.org/abs/2309.02760.

40 Kan Ching Ng. *Relation algebras with transitive closure*. PhD thesis, University of California, 1984.

41 Peter W. O'Hearn. Incorrectness logic. *Proceedings of the ACM on Programming Languages*, 4(POPL):10:1–10:32, 2019. doi:10.1145/3371078.

42 Jean-Éric Pin. The dot-depth hierarchy, 45 years later. In *The Role of Theory in Computer Science*, pages 177–201. WORLD SCIENTIFIC, 2016. doi:10.1142/9789813148208_0008.

43 Damien Pous. On the positive calculus of relations with transitive closure. In *STACS*, volume 96 of *LIPIcs*, pages 3:1–3:16. Schloss Dagstuhl, 2018. doi:10.4230/LIPICS.STACS.2018.3.

44 Damien Pous, Jurriaan Rot, and Jana Wagemaker. On tools for completeness of kleene algebra with hypotheses. *Logical Methods in Computer Science*, Volume 20, Issue 2, 2024. doi:10.46298/lmcs-20(2:8)2024.

45 Damien Pous and Jana Wagemaker. Completeness theorems for Kleene algebra with top. In *CONCUR*, volume 243 of *LIPIcs*, pages 26:1–26:18. Schloss Dagstuhl, 2022. doi:10.4230/LIPICS.CONCUR.2022.26.

46 V. R. Pratt. Dynamic algebras and the nature of induction. In *STOC*, pages 22–28. ACM, 1980. doi:10.1145/800141.804649.

47 Sylvain Schmitz. Complexity hierarchies beyond elementary. *ACM Transactions on Computation Theory*, 8(1):1–36, 2016. doi:10.1145/2858784.

48 Steffen Smolka, Nate Foster, Justin Hsu, Tobias Kappé, Dexter Kozen, and Alexandra Silva. Guarded Kleene algebra with tests: verification of uninterpreted programs in nearly linear time. *Proceedings of the ACM on Programming Languages*, 4(POPL):61:1–61:28, 2019. doi:10.1145/3371129.

49 Larry J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic*. PhD thesis, Massachusetts Institute of Technology, 1974. doi:1721.1/15540.

50 Alfred Tarski. On the calculus of relations. *The Journal of Symbolic Logic*, 6(3):73–89, 1941. doi:10.2307/2268577.

51 Wolfgang Thomas. A concatenation game and the dot-depth hierarchy. In *Computation Theory and Logic*, number 270 in LNCS, pages 415–426. Springer, 1987. doi:10.1007/3-540-18170-9_183.

52 B. A. Trakhtenbrot. The impossibility of an algorithm for the decision problem in finite classes. *Doklady Akademii Nauk SSSR*, 70(4):569–572, 1950.

53 Cheng Zhang, Arthur Azevedo de Amorim, and Marco Gaboardi. On incorrectness logic and Kleene algebra with top and tests. *Proceedings of the ACM on Programming Languages*, 6(POPL):29:1–29:30, 2022. doi:10.1145/3498690.

A Slight Extensions of Theorem 3.2

In this section, we note that we can extend Theorem 3.2 in the following two:

▶ **Theorem A.1.** *For all* positive *quantifier-free formulas* φ *of* $\mathsf{KA}_{\{-\}}$ *terms, we have:* $\mathsf{LANG} \models \varphi \Rightarrow \mathsf{RSUB} \models \varphi$.

Proof Sketch. By the same surjective S-homomorphism in the proof of Theorem 3.2(⇒). ◀

▶ **Theorem A.2.** *For all quantifier-free formulas* φ *of* $\mathsf{KA}_{\{-\}}$ *terms, we have:* $\mathsf{RSUB} \models \varphi \Rightarrow \mathsf{LANG} \models \varphi$.

Proof. Because the formulas $t = s \leftrightarrow (t \leq s \wedge s \leq t)$ and $t \leq s \leftrightarrow t \cap s^{-} \leq 0$ are valid on $\mathsf{LANG} \cup \mathsf{SUB}$, without loss of generality, we can assume that each equation in φ is of the form $u \leq 0$. By taking the conjunctive normal form, it suffices to prove when φ is of the form $(\bigvee_{i=1}^{n} t_i \leq 0) \vee (\bigvee_{j=1}^{m} \neg s_j \leq 0)$. We prove the contraposition. By $\mathsf{LANG} \not\models \varphi$, there are $X, \mathfrak{v} \in \mathsf{LANG}_X, w_1, \ldots, w_n \in X^*$ such that $w_i \in \hat{\mathfrak{v}}(t_i)$ for $i \in [1, n]$ and $\hat{\mathfrak{v}}(s_j) = \emptyset$ for $j \in [1, m]$. By letting $w_0 \triangleq w_1 \ldots w_n$ and considering the same S-homomorphism as Theorem 3.2(⇐), we have $\mathsf{RSUB} \not\models \varphi$. ◀

For Theorem A.2, note that the converse direction fails (Remark 3.3), cf. Theorem A.1.

B Proof of Corollary 5.5

Proof. We have:

$$\begin{aligned}
\mathcal{C} \models t \leq s \;&\Leftrightarrow\; \forall w \in [t]_{\tilde{\mathbf{V}}}, \forall H \in \mathcal{S}_{\mathcal{C}}(\mathsf{G}(w)), \exists v \in [s]_{\tilde{\mathbf{V}}}, \mathsf{G}(v) \longrightarrow H^{\mathcal{Q}} \\
&\qquad\qquad (\text{Theorem 5.2 and } \mathcal{G}(s) = \{\mathsf{G}(v) \mid v \in [s]_{\tilde{\mathbf{V}}}\}) \\
&\Leftrightarrow\quad \forall w \in [t]_{\tilde{\mathbf{V}}}, \forall H \in \mathcal{S}_{\mathcal{C}}(\mathsf{G}(w)), [s]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \neq \emptyset \qquad ([H]_{\tilde{\mathbf{V}}} = \{v \in \tilde{\mathbf{V}}^* \mid \mathsf{G}(v) \longrightarrow H^{\mathcal{Q}}\}) \\
&\Leftrightarrow\quad [t]_{\tilde{\mathbf{V}}} \subseteq \{w \in \tilde{\mathbf{V}}^* \mid \forall H \in \mathcal{S}_{\mathcal{C}}(\mathsf{G}(w)), [s]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \neq \emptyset\}.
\end{aligned}$$

◀

C Proof of Proposition 6.2

Let $R' \subseteq |H| \times |J|$ be the minimal set such that
- $\langle 1^H, 1^J \rangle \in R'$,
- $\forall a \in \tilde{\mathbf{V}}_1, \forall x, x' \in |H|, \forall y, y' \in |J|$, $(\langle x, y \rangle \in R' \wedge \langle x, x' \rangle \in \delta_a^H \wedge \langle y, y' \rangle \in \delta_a^J) \Rightarrow \langle x', y' \rangle \in R'$.

▷ Claim C.1. $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} \neq \emptyset \quad \Leftrightarrow \quad \langle 2^H, 2^J \rangle \in R'$.

Proof. By definition, R' coincides with the set of all reachable states of the product NFA of H and J. ◁

Let $R \subseteq |H| \times |J|$ be the minimal set such that
- $\langle 1^H, 1^J \rangle \in R$,
- $\forall a \in \tilde{\mathbf{V}}_1, \forall x, x' \in |H|, \forall y, y' \in |J|$, $(\langle x, y \rangle \in R \wedge \langle x, x' \rangle \in a^H \wedge \langle y, y' \rangle \in \delta_a^J) \Rightarrow \langle x', y' \rangle \in R$.

▷ Claim C.2. $R = R'$.

Proof. ($\subseteq$): Clear, by $a^H \subseteq \delta_a^H$. ($\supseteq$): By induction on derivations of R'.
- Case $\langle 1^H, 1^J \rangle \in R'$: Trivial, by $\langle 1^H, 1^J \rangle \in R$.
- Case $(\langle x, y \rangle \in R' \wedge \langle x, x' \rangle \in \delta_a^H \wedge \langle y, y' \rangle \in \delta_a^J) \Rightarrow \langle x', y' \rangle \in R'$: By IH, $\langle x, y \rangle \in R$.
  - Sub-Case $a \neq 1$: Let $x_0, \ldots, x_{n-1}, x_n, \ldots, x_m$ be s.t. $\langle x, x' \rangle = \langle x_0, x_m \rangle$ and
    - for all $i \in [1, n-1]$, $\langle x_{i-1}, x_i \rangle \in 1^H$,
    - $\langle x_{n-1}, x_n \rangle \in a^H$,
    - for all $i \in [n+1, m]$, $\langle x_{i-1}, x_i \rangle \in 1^H$.

    Let $y_0 = \cdots = y_{n-1} = y$ and $y_n = \cdots = y_m = y'$. Then by applying the second rule multiply, we have $\langle x', y' \rangle \in R$.
  - Sub-Case $a = 1$: By reflexivity of 1^H, $\langle x, x' \rangle \in (1^H)^+$. Let $x_0, \ldots, x_m$ $(m > 0)$ be s.t. $\langle x, x' \rangle = \langle x_0, x_m \rangle$ and
    - for all $i \in [1, m]$, $\langle x_{i-1}, x_i \rangle \in 1^H$.

    Let $y_0 = y$ and $y_1 = \cdots = y_m = y'$. Then by applying the second rule multiply, we have $\langle x', y' \rangle \in R$. ◁

Proof of Proposition 6.2. ($\Rightarrow$): By letting U as the map defined by $U(x) \triangleq \{y \mid \langle x, y \rangle \in R\}$. Here, $2^J \notin U_{2^H}$ is shown by $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$ with **??** C.1**??** C.2. ($\Leftarrow$): Let $R'' \triangleq \{\langle x, y \rangle \mid y \in U(x)\}$. By the minimality of R, we have $R \subseteq R''$. By $\langle 2^H, 2^J \rangle \notin R''$, we have $\langle 2^H, 2^J \rangle \notin R$. Hence by Claim C.1, Claim C.2, we have $[J]_{\tilde{\mathbf{V}}} \cap [H]_{\tilde{\mathbf{V}}} = \emptyset$. ◀

D Proof of Lemma 6.5

Proof. ($\Rightarrow$): By Proposition 6.2, let $H' \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(w))$ and let U be an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H']_{\tilde{\mathbf{V}}} = \emptyset$. We define the graph H as follows:
- $|H| = |H'|$,
- $a^H = a^{H'}$ for $a \in \{\top, 1, \overline{1}\}$,
- $a^H = a^{\mathsf{G}(w)} \cup (a^{H'} \cap 1^{H'})$ for $a \in \tilde{\mathbf{V}}_1 \setminus \{\top, 1, \overline{1}\}$.

We then have that the pair $P \triangleq \langle H, U \rangle$ is a saturable path for $w \in \mathsf{L}_J$, as follows:
- (P-Ext): By that H' is an edge-saturation w.r.t. RSUB.
- (P-Con): Because H' is consistent by $H' \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(w))$.
- (P-Wit): Because U is an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H']_{\tilde{\mathbf{V}}} = \emptyset$.
- (P-Sat): Because $a^{H'} \cup \overline{a}^{H'} = \top^{H'}$ and U is an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H']_{\tilde{\mathbf{V}}} = \emptyset$.

($\Leftarrow$): Let $P = \langle H, U \rangle$ be a saturable path for $w \in \mathsf{L}_J$. By (P-Ext), 1^H is an equivalence relation. We define the graph H' as follows:

- $|H'| = |H|$,
- $a^{H'} = a^H$ for $a \in \{\top, \mathbf{1}, \overline{\mathbf{1}}\}$,
- for $a \in \mathbf{V}$ and $\langle x, y\rangle \in \top^H$,
  - if $\langle [x]_{1^H}, [y]_{1^H}\rangle \in a^{H^{\Omega}}$, then $\langle x, y\rangle \in a^{H'} \setminus \overline{a}^{H'}$,
  - else if $\langle [x]_{1^H}, [y]_{1^H}\rangle \in \overline{a}^{H^{\Omega}}$, then $\langle x, y\rangle \in \overline{a}^{H'} \setminus a^{H'}$,
  - else if $U_y \subseteq \delta_a^J(U_x)$, then $\langle x, y\rangle \in a^{H'} \setminus \overline{a}^{H'}$,
  - else $\langle x, y\rangle \in \overline{a}^{H'} \setminus a^{H'}$.

By the construction of H', we have the following:

- H' is an edge-extension of H: By (P-Con), if $\langle [x]_{1^H}, [y]_{1^H}\rangle \in \overline{a}^{H^{\Omega}}$, then $\langle [x]_{1^H}, [y]_{1^H}\rangle \notin a^{H^{\Omega}}$.
- H' is consistent: If $[x]_{1^H} = [y]_{1^H}$ then $U_x = U_y$, because $U_x \subseteq \delta_1^J(U_x) \subseteq U_y \subseteq \delta_1^J(U_y) \subseteq U_x$ by (P-Wit); thus, if $[x]_{1^H} = [x']_{1^H}$ and $[y]_{1^H} = [y']_{1^H}$, then $\langle x, y\rangle \in a^{H'}$ iff $\langle x', y'\rangle \in a^{H'}$.
- for $a \in \mathbf{V}$, $\overline{a}^{H'} = \top^{H'} \setminus a^{H'}$: Because $a^{H'} \cup \overline{a}^{H'} = \top^{H'}$ and H' is consistent.

From them and (P-Ext), we have $H' \in \mathcal{S}_{\mathsf{RSUB}}(\mathsf{G}(w))$. Also, U is an emptiness-witness for $[J]_{\tilde{\mathbf{V}}} \cap [H']_{\tilde{\mathbf{V}}} = \emptyset$ as follows. For edges already in H, it is shown by (P-Wit). For extended edges from H, it is shown by the construction of H' (for the last case of the four cases above, by $U_y \not\subseteq \delta_a^J(U_x)$ and (P-Sat), we have $U_y \subseteq \delta_{\overline{a}}^J(U_x)$). Hence, this completes the proof. ◀

E Proof of Lemma 6.9

Proof. ($\subseteq$): Let $w = a_1 \dots a_n \in \mathsf{L}_J$. Let $P = \langle H, U\rangle$ be a saturable path for $w \in \mathsf{L}_J$. Let $X \triangleq \{a \in \tilde{\mathbf{V}}_1 \mid a^H \supseteq \mathbf{1}^H\}$ (note that $X \in \mathcal{X}$). For each i, let $\mathcal{U}_i \triangleq \bigcup_{j;\langle j,i\rangle \in \overline{\mathbf{1}}^H} U_j^2$. Then we have:

- $\varphi^J(\mathcal{U}_i, U_i)$: By (P-Sat) and Proposition 6.7.
- $\forall a \in X, \delta_a^J(U_i) \subseteq U_i$: By $a^H \supseteq \mathbf{1}^H \supseteq \Delta_{|H|}$ and (P-Wit).

Thus $\langle \mathcal{U}_i, U_i\rangle \in |J^{\mathcal{S}_X}|$. We consider the following run of the NFA $J^{\mathcal{S}_X}$ on w:

$$\to \blacktriangleright \xrightarrow{1} \langle \mathcal{U}_0, U_0\rangle \xrightarrow{a_1} \langle \mathcal{U}_1, U_1\rangle \xrightarrow{a_2} \langle \mathcal{U}_2, U_2\rangle \longrightarrow \cdots \xrightarrow{a_n} \langle \mathcal{U}_n, U_n\rangle \xrightarrow{1} \blacktriangleleft \to .$$

This is indeed a run of the NFA $J^{\mathcal{S}_X}$ as follows:

- $\langle \blacktriangleright, \langle \mathcal{U}_0, U_0\rangle\rangle \in 1^{J^{\mathcal{S}_X}}$: By $1^J \in U_0$ (P-Wit) and $\mathcal{U}_0 = \emptyset$.
- $\langle \langle \mathcal{U}_n, U_n\rangle, \blacktriangleleft\rangle \in 1^{J^{\mathcal{S}_X}}$: By $2^J \notin U_n$ (P-Wit).
- $\forall i \in [1, n], \langle \langle \mathcal{U}_{i-1}, U_{i-1}\rangle, \langle \mathcal{U}_i, U_i\rangle\rangle \in a_i^{J^{\mathcal{S}_X}}$: We distinguish the following cases:
  - Case $\langle i-1, i\rangle \in 1^H$:
    - $\mathcal{U}_i = \mathcal{U}_{i-1}$: By $\langle j, i\rangle \in \overline{\mathbf{1}}^H$ iff $\langle j, i-1\rangle \in \overline{\mathbf{1}}^H$, for all j.
    - $U_i = U_{i-1}$: By (P-Wit), we have $U_{i-1} \subseteq \delta_1^J(U_{i-1}) \subseteq U_i \subseteq \delta_1^J(U_i) \subseteq U_{i-1}$.
    - $a_i \in X$ ($a_i^H \supseteq \mathbf{1}^H$): By $a_i^H \cap \mathbf{1}^H \neq \emptyset$ and (P-Ext), we have $a_i^H = a_i^{\mathsf{G}(w)} \cup \mathbf{1}^H$ (if not, this contradicts to (P-Con)).

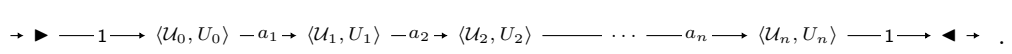
    Thus by $\psi_{a_i,1}^X(\mathcal{U}_{i-1}, U_{i-1}, \mathcal{U}_i, U_i)$, we have $\langle \langle \mathcal{U}_{i-1}, U_{i-1}\rangle, \langle \mathcal{U}_i, U_i\rangle\rangle \in a_i^{J^{\mathcal{S}_X}}$.
  - Case $\langle i-1, i\rangle \in \overline{\mathbf{1}}^H$:
    - $\mathcal{U}_i = \mathcal{U}_{i-1} \cup U_{i-1}^2$: By $\langle j, i\rangle \in \overline{\mathbf{1}}^H$ iff $j < i$ iff $\langle j, i-1\rangle \in \overline{\mathbf{1}}^H \vee \langle j, i-1\rangle \in \mathbf{1}^H$, for all j. (Intuitively, $\mathcal{U}_{i-1}$ corresponds to the case $\langle j, i-1\rangle \in \overline{\mathbf{1}}^H$ and U_{i-1}^2 corresponds to the case $\langle j, i-1\rangle \in \mathbf{1}^H$.)
    - $\delta_{a_i}^J(U_{i-1}) \subseteq U_i$: By (P-Wit).
    - $\delta_\top^J(\{u \mid \langle u, u\rangle \in \mathcal{U}_i\}) \subseteq U_i$: We have $\delta_\top^J(\{u \mid \langle u, u\rangle \in \mathcal{U}_i\}) = \delta_\top^J(\bigcup_{j;\langle j,i\rangle \in \overline{\mathbf{1}}^H} U_j) = \bigcup_{j<i} \delta_\top^J(U_j) \subseteq U_i$ by (P-Wit).

* $\delta_{\overline{1}}^J(\{u \mid \langle u,u\rangle \in \mathcal{U}_i\}) \subseteq U_i$: We have $\delta_{\overline{1}}^J(\{u \mid \langle u,u\rangle \in \mathcal{U}_i\}) = \delta_{\overline{1}}^J(\bigcup_{j;\langle j,i\rangle \in \overline{1}^H} U_j) = \bigcup_{j<i} \delta_{\overline{1}}^J(U_j) \subseteq U_i$ by (P-Wit).

Thus by $\psi^X_{a_i,\overline{1}}(\mathcal{U}_{i-1}, U_{i-1}, \mathcal{U}_i, U_i)$, we have $\langle\langle \mathcal{U}_{i-1}, U_{i-1}\rangle, \langle \mathcal{U}_i, U_i\rangle\rangle \in a_i^{J^{\mathcal{S}_X}}$.

Hence, $w \in [J^{\mathcal{S}_X}]$.

($\supseteq$): Let $X \subseteq \mathcal{X}$ and $w = a_1 \dots a_n \in [J^{\mathcal{S}_X}]_{\tilde{\mathbf{V}}}$. Let the run of $J^{\mathcal{S}_X}$ on w be as follows:



Let H be the edge-extension of $\mathsf{G}(w)$ defined as follows:

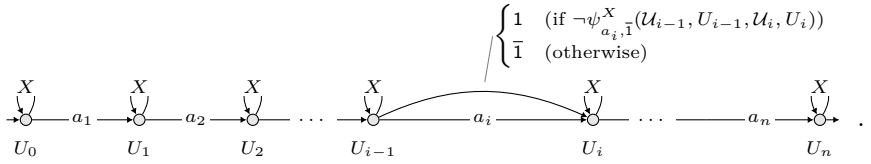
- $\top^H = \{\langle x,y\rangle \in [0,n]^2 \mid \forall i \in [y+1,x],\ \neg\psi^X_{a_i,\overline{1}}(\mathcal{U}_{i-1}, U_{i-1}, \mathcal{U}_i, U_i)\}$,
- $1^H = \top^H \cap \{\langle x,y\rangle \mid \langle y,x\rangle \in \top^H\}$ and $\overline{1}^H = \top^H \setminus 1^H$,
- $\forall a \in \mathbf{V} \cap X$, $\langle a^H, \overline{a}^H\rangle = \langle a^{\mathsf{G}(w)} \cup 1^H, \overline{a}^{\mathsf{G}(w)}\rangle$.

Note that by definition of $\top^H$, we have

- $\top^H \supseteq \{\langle x,y\rangle \mid x \le y\}$,
- $\top^H$ is transitive (by case analysis).

Hence, $\top^H$ is a total preorder and each equivalence class w.r.t. 1^H is an interval $[l,r]$.

Let $P \triangleq \langle H, U\rangle$ where U is defined as $i \mapsto U_i$ for $i \in [0,n]$. The following depicts P.



Then P is a saturable path for $w \in \mathsf{L}_J$ as follows:

- (P-Ext): By the definition of H.
- (P-Con): Assume that $a^{H^{\mathrm{Q}}} \cap \overline{a}^{H^{\mathrm{Q}}} \neq \emptyset$. Let x, x', y, y' be s.t. $[x]_{1^H} = [x']_{1^H}$, $[y]_{1^H} = [y']_{1^H}$, $\langle x,y\rangle \in a^H$, and $\langle x',y'\rangle \in \overline{a}^H$. WLOG, we can assume that $a \in X$ and $\overline{a} \notin X$. Then, we have the following:
  - $\langle x',y'\rangle \in \overline{a}^{\mathsf{G}(w)}$ (so, $x' = y' - 1$ and $a_{y'} = \overline{a}$): By $\overline{a}^H = \overline{a}^{\mathsf{G}(w)}$ (since $\overline{a} \notin X$).
  - $\langle x,y\rangle \in a^{\mathsf{G}(w)}$ (so, $x = y - 1$ and $a_y = a$): If not, then by $a^H = a^{\mathsf{G}(w)} \cup 1^H$, we have $[x]_{1^H} = [y]_{1^H}$. Thus, $\langle y', y'-1\rangle \in 1^H (\subseteq \top^H)$. By the definition of $\top^H$, we have $\neg\psi^X_{a_{y'},\overline{1}}(\mathcal{U}_{y'-1}, U_{y'-1}, \mathcal{U}_{y'}, U_{y'})$. By the definition of $a^{J^{\mathcal{S}_X}}$, we have $\psi^X_{a_{y'},1}(\mathcal{U}_{y'-1}, U_{y'-1}, \mathcal{U}_{y'}, U_{y'})$, so $\overline{a} \in X$. This contradicts $\overline{a} \notin X$.
  - $([x,x'] \cup [x',x]) \cap ([y,y'] \cup [y',y]) = \emptyset$ (so, $x = x'$ and $y = y'$): If not, then because the interval between x and x' and that between y and y' have an intersection, we have $[x]_{1^H} = [y]_{1^H}$. Then, in the same manner as above, we have $\overline{a} \in X$. This contradicts $\overline{a} \notin X$.

  Thus, we reach a contradiction, because $a = a_y = a_{y'} = \overline{a}$ (by $y = y'$). Hence, $a^{H^{\mathrm{Q}}} \cap \overline{a}^{H^{\mathrm{Q}}} = \emptyset$.
- (P-Sat): By the form of $J^{\mathcal{S}_X}$, we have $\mathcal{U}_x = \begin{cases} \mathcal{U}_{x-1} & (\langle x-1, x\rangle \in 1^H) \\ \mathcal{U}_{x-1} \cup U_{x-1}^2 & (\langle x-1, x\rangle \in \overline{1}^H) \end{cases}$. Thus, $\mathcal{U}_y = \bigcup_{x;\langle x,y\rangle \in \overline{1}^H} U_x^2$ ($\bigstar$). By Proposition 6.7, this completes the proof.
- (P-Wit): For $1^J \in U_0$ and $2^J \notin U_n$, they are shown by the form of $J^{\mathcal{S}_X}$. For $\forall a \in \tilde{\mathbf{V}}_1, \forall \langle x,y\rangle \in a^H, \delta_a^J(U_x) \subseteq U_y$, we distinguish the following cases:
  - Case $a = 1$: Then we have
    * $U_x = U_y$: By $\langle x,y\rangle \in 1^H$ and the form of $J^{\mathcal{S}_X}$, we have the following: $\forall z \in [y+1,x]$, $\psi^X_{a_z,1}(\mathcal{U}_{z-1}, U_{z-1}, \mathcal{U}_z, U_z)$. Thus, $U_y = U_{y+1} = \dots = U_x$.

  - $\delta^J_{\overline{1}}(U_x) \subseteq U_x$: By $\langle \mathcal{U}_x, U_x\rangle \in |J^{\mathcal{S}_X}|$.

  Hence, $\delta^J_{\overline{1}}(U_x) \subseteq U_y$.
- Case $a = \overline{1}$: Let $z \in [x+1, y]$ be such that $\psi^X_{a_z,\overline{1}}(\mathcal{U}_{z-1}, U_{z-1}, \mathcal{U}_z, U_z)$ and $\forall z' \in [z+1, y], \neg\psi^X_{a_{z'},\overline{1}}(\mathcal{U}_{z'-1}, U_{z'-1}, \mathcal{U}_{z'}, U_{z'})$. Then we have

$$\begin{aligned}
\delta^J_{\overline{1}}(U_x) &\subseteq \delta^J_{\overline{1}}(\{u \mid \langle u, u\rangle \in \mathcal{U}_z\}) && \text{(by } (\bigstar) \text{ and } \langle x, z\rangle \in \overline{1}^H \text{ (by } \langle z-1, z\rangle \in \overline{1}^H)) \\
&\subseteq U_z && \text{(by } \psi^X_{a_z,\overline{1}}(\mathcal{U}_{z-1}, U_{z-1}, \mathcal{U}_z, U_z)) \\
&\subseteq U_{z+1} = \cdots = U_y. && \text{(by the form of } J^{\mathcal{S}_X}, \psi^X_{a_{z'},1}(\mathcal{U}_{z'-1}, U_{z'-1}, \mathcal{U}_{z'}, U_{z'}))
\end{aligned}$$

- Case $a = \top$: We distinguish the following two sub-cases:
  - Case $\langle x, y\rangle \in \overline{1}^H$: By the similar argument as Case $a = \overline{1}$.
  - Case $\langle x, y\rangle \in 1^H$: By the similar argument as Case $a = 1$, we have $U_x = U_y$ and $\delta^J_\top(U_x) \subseteq U_x$, and thus $\delta^J_\top(U_x) \subseteq U_y$.
- Case $a \in \{a, \overline{a} \mid a \in \mathbf{V}\}$: We distinguish the following sub-cases:
  - Case $\langle x, y\rangle \in \overline{1}^H$: By $\langle x, y\rangle \in a^H \cap \overline{1}^H = a^{\mathsf{G}(w)}$, we have $x = y - 1$ and $a_y = a$. Thus by $\psi^X_{a_y,\overline{1}}(\mathcal{U}_{y-1}, U_{y-1}, \mathcal{U}_y, U_y)$, we have $\delta^J_a(U_x) \subseteq U_y$.
  - Case $a \notin X$: By $a^H = a^{\mathsf{G}(w)}$, we have $x = y - 1$ and $a_y = a$. By the form of $J^{\mathcal{S}_X}$ with $\neg\psi^X_{a_y,1}(\mathcal{U}_{y-1}, U_{y-1}, \mathcal{U}_y, U_y)$ (since $a_y \notin X$), we have $\psi^X_{a_y,\overline{1}}(\mathcal{U}_{y-1}, U_{y-1}, \mathcal{U}_y, U_y)$. Hence, $\delta^J_a(U_x) \subseteq U_y$.
  - Case $\langle x, y\rangle \in 1^H$ and $a \in X$: By the similar argument as Case $a = 1$, we have $U_x = U_y$ (by $\langle x, y\rangle \in 1^H$) and $\delta^J_a(U_x) \subseteq U_x$ (by $a \in X$). Thus, $\delta^J_a(U_x) \subseteq U_y$. ◀

A Complete Graphical Language for Linear Optical Circuits with Finite-Photon-Number Sources and Detectors

Nicolas Heurtel
Quandela, 7 Rue Léonard de Vinci, 91300 Massy, France
Université Paris-Saclay, CNRS, ENS Paris-Saclay, Inria, Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France

Abstract

Graphical languages are powerful and useful to represent, rewrite and simplify different kinds of processes. In particular, they have been widely used for quantum processes, improving the state of the art for compilation, simulation and verification. In this work, we focus on one of the main carrier of quantum information and computation: linear optical circuits. We introduce the $\mathbf{LO}_{fi}$-calculus, the first graphical language to reason on the infinite-dimensional photonic space with circuits only composed of the four core elements of linear optics: the phase shifter, the beam splitter, and auxiliary sources and detectors with bounded photon number. First, we study the subfragment of circuits composed of phase shifters and beam splitters, for which we provide the first minimal equational theory. Next, we introduce a rewriting procedure on those $\mathbf{LO}_{fi}$-circuits that converge to normal forms. We prove those forms to be unique, establishing both a novel and unique representation of linear optical processes. Finally, we complement the language with an equational theory that we prove to be complete: two $\mathbf{LO}_{fi}$-circuits represent the same quantum process if and only if one can be transformed into the other with the rules of the $\mathbf{LO}_{fi}$-calculus.

2012 ACM Subject Classification Theory of computation → Quantum computation theory; Theory of computation → Axiomatic semantics; Hardware → Quantum computation

Keywords and phrases Quantum Computing, Graphical Language, Linear Optical Circuits, Linear Optical Quantum Computing, Completeness, Fock Space

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.38

Related Version *Full Version*: `https://arxiv.org/abs/2402.17693` [27]

Funding This work has been partially funded by the European Commission as part of the EIC accelerator program, under the grant agreement 190188855, by the French National Research Agency (ANR) with the project TaQC ANR-22-CE47-0012 and within the framework of "Plan France 2030", under the research projects EPIQ ANR-22-PETQ-0007, OQULUS ANR-23-PETQ-0013, HQI-Acquisition ANR-22-PNCQ-0001 and HQI-R&D ANR-22-PNCQ-0002, and by the CIFRE 2022/0081.

Acknowledgements We would like to thank Marc de Visme and Vladimir Zamdzhiev for helpful discussions, Alexandre Clément for the insight into the angles of (E3) and the derivation of (oE3), and particularly Shane Mansfield, Benoît Valiron and Renaud Vilmart for helpful discussions, support and reviews of some parts of the paper. We would also like to thank all the anonymous reviewers for their insightful comments and suggestions, which greatly helped to improve the quality of this manuscript.

1 Introduction

Quantum computing is a paradigm for processing information [41, 45] that performs computation with quantum states, instead of the classical states of bits. This computational paradigm allows specific computational problems to be solved with quadratic [24] or even exponential

speedup [50, 26] compared to their classical counterparts. To encode that quantum data, many technologies have been pursued, such as superconducting circuits [32], trapped ions [8] and cold atoms [23].

One of the prominent candidates for quantum computation is linear optics [36, 42, 47], where the *logical* information is encoded into the quantum states of photons, the *particles* of light. For quantum computation, the logical states are encoded onto the *modes* of the photons, i.e. their degrees of freedom like their *positions* in the circuit, and the desired logical operations are performed with optical components. All scalable quantum computations with linear optics [34, 53, 40, 7, 6, 17] encoding with the positions of the photons use predominantly these following elements.

- Sources: they generate the quantum state, i.e. a vector in a Hilbert space,
- Phase shifters: they change the quantum state by adding a phase to the light passing through them[1],
- Beam splitters: they alter the quantum state by causing photons on two different paths to interfere with each other[2],
- Detectors: they project the quantum state on a subspace.

As ubiquitous as the circuits made of those components are in linear optical quantum computation schemes, as illustrated in Figure 1 and 2, many unanswered questions persist regarding optimality, minimality and an efficient use of those components. We wish to have a framework finding the most appropriate implementation for the desired computation or protocol. The purpose of this work is therefore to propose a formal framework to model and manipulate generic circuits composed of the four previous elements.

State of the art. Some main formal frameworks to study, develop or optimize quantum processes are graphical languages [2, 49, 3, 43], representing processes with diagrams and equations between those diagrams. These formalisms have been shown to be very useful for addressing quantum processes in general, such as **ZX**-diagrams [13] with applications in compilation [29, 5, 51], simulation [31, 30, 35] and verification [19, 21]. To completely capture the processes those diagrams model, [28, 25] have introduced a complete set of equations: two equivalent **ZX**-diagrams can always be transformed from one to the other with those equations.

Recently, some works have modeled optical processes with diagrams [4, 12, 39], including notably $\mathbf{LO_v}$ [10], a complete graphical language for linear optical circuits with vacuum sources and detectors, and **QPath** [15], a graphical language to compute amplitudes. Remarkably, both have also led to results beyond the optical realm, as a subfragment of the first led to derive the first complete equational theory for quantum circuits [11], while the second introduced a functor from the **ZX**-calculus [15] and led to a more generic language [16].

However, those two frameworks don't completely capture linear optical circuits with sources and detection schemes. In particular, $\mathbf{LO_v}$ lacks a many-photon semantics and can only cover the single-photon case, while **QPath** uses sums of diagrams in the rewriting process along with generators that are not linear optical components. For instance, we would like to be able to model the photonic implementation of the CZ gate [34, 33], a prominent logical quantum gate, and rewrite it to equivalent forms, as illustrated in Figure 1.

[1] They are typically implemented using thermo-optic components, where the refractive index of the waveguide is changed by heating the material.

[2] In integrated circuits, beam splitters are implemented using waveguides that split and combine light paths.

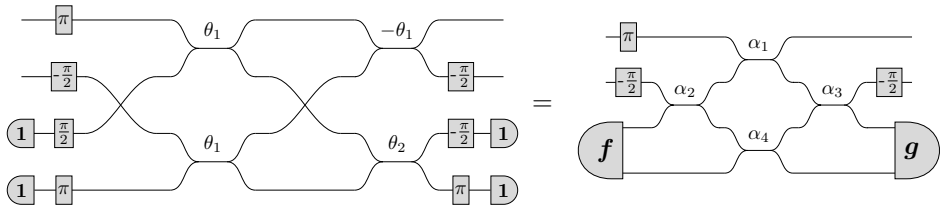


Figure 1 Optical circuits implementing the CZ two-qubit logic gate with auxiliary sources and detectors. On the left is the original circuit[3] of [33]. There are two auxiliary photon generated on the bottom left: if exactly one photon is detected for each of the two last wires on the bottom right, then we know we have performed the operation $|11\rangle \mapsto -|11\rangle$ on the two first wires. This event has a probability $\frac{2}{27}$ to occur. On the right is an equivalent representation in the $\mathbf{LO}_{fi}$-calculus, where $\boldsymbol{f}$ and $\boldsymbol{g}$ are two-photon states and linear forms.

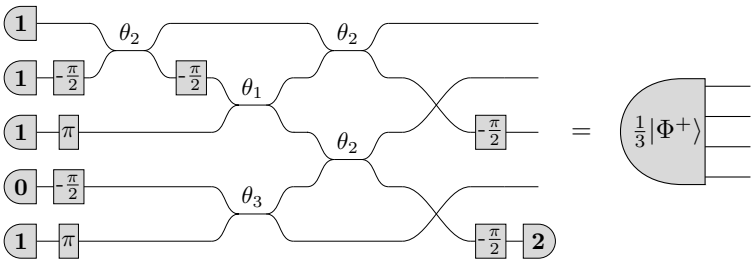


Figure 2 Linear optical circuit generating with a $\frac{1}{9}$ probability the Bell state $\left|\Phi^+\right\rangle = |1010\rangle + |0101\rangle$, with the use of auxiliary sources and detectors. On the left is the original[3] circuit of [20], on the right is an equivalent and modular description. Both are equivalent circuits in the $\mathbf{LO}_{fi}$-calculus.

Challenges. In seeking to develop a graphical language for modeling linear optical circuits with a many-photon semantics, there are two main challenges. First, the bosonic Fock space, representing all the states that photons can be in, is an infinite-dimensional Hilbert space: the bosonic Fock space. In particular, some properties and generators of graphical languages with finite-dimensional theories [44, 52, 18] cannot be used. Second, the interaction of photons, even without bringing auxiliary modes and detections into the picture, is described by the permanents of matrices [48, 1], making cumbersome explicit expression and manipulation of all the involved terms.

Contributions. In this paper, we propose such a framework, and introduce the $\mathbf{LO}_{fi}$-calculus, the first graphical language defined on the bosonic Fock space, with circuits composed of four core elements of linear optics: the phase shifter, the beam splitter, and auxiliary finite-photon-number sources and detectors. Our main contributions are the following.

- A complete equational theory for circuits with phase shifters and beam splitters which is simpler than the one in [10], and that we prove to be minimal (Section 2).

[3] Some phases have been added to take into account the different conventions for the semantics of the beam splitters.

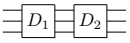
- A new sound and complete equational theory for linear optical circuits that allows all auxiliary finite-photon-number states and detections (Section 3).
- A unique and compact universal form for optical circuits of this kind, obtained through a deterministic rewriting procedure and proven to be unique with new techniques (Section 4).

All the notation introduced in the paper is summarized in Table 1.

2 LOPP: Linear optical quantum circuits with single-photon semantics

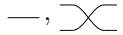
A linear optical quantum circuit consists of spatial modes through which photons pass – which may be physically instantiated by optical fibers, waveguides in integrated circuits, or simply by paths in free space (bulk optics) – and operations that act on those spatial modes, including in particular *beam splitters* (), and *phase shifters* ( ).

2.1 Syntax and single-photon semantics

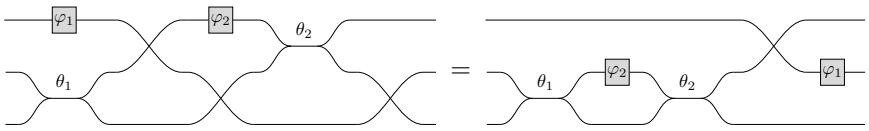
Similarly to [10], in order to formalize linear optical circuits with diagrams, we use the formalism of PROPs [38]. A PRO is a strict monoidal category whose monoid of objects is freely generated by a single X: the objects are all of the form $X \otimes X \otimes \cdots \otimes X$, and simply denoted by n, the number of occurrences of X. PROs are typically represented graphically as circuits: each copy of X is represented by a wire and morphisms by boxes on wires, so that $\otimes$ is represented vertically and morphism composition $\circ$ is represented horizontally. For instance, D_1 and D_2 represented as  and  can be horizontally composed as $D_2 \circ D_1$, represented by  , and vertically composed as $D_1 \otimes D_2$, represented by  . A PROP is the symmetric monoidal analogue of PRO, so it is equipped with a swap. It means the circuits are equivalent through wire deformations and that only the connectivity matters.

▶ **Definition 1.** **LOPP**[4] *is the PROP of* **LOPP***-circuits generated by:*

with $\varphi \in \mathbb{R}$ *and* $\theta \in \mathbb{R}$.

The convention is to go through from left to right, meaning the inputs (resp. outputs) are on the left (resp. right), and from top to bottom, meaning the first (resp. last) input is the top (resp. last) wire. The identity, the swap and the empty diagrams are noted as follows:  .

▶ **Example 2.** Here are two examples of **LOPP**-circuits, that are equivalent up to deformation with the rules of PROPs:



[4] The PROP version of **LOPP** has first been defined in [9], as [10] defined $\mathbf{LO_{PP}}$ as a PRO.

The semantics of linear optical components are usually described by their behavior when there is one single photon passing through those components. Given a circuit of m wires, the single photon can be in a superposition of the m different positions, so its state is a vector in $\mathbb{C}^m$. We consider the standard orthonormal basis $\{|e_i\rangle, i \in [1,m]\}$ where $e_i = |0,\dots,0,1,0,\dots,0\rangle$ with a 1 at the i^{th} component. The object of our PROP is therefore $X = \mathbb{C}$, and the vertical composition is interpreted as the direct sum [10, 11]. The semantics is defined as follows.

▶ **Definition 3** (Semantics of **LOPP**). *The single photon semantics of* **LOPP** *is inductively defined as follows:* $[\![C_1 \otimes C_2]\!]_1 = [\![C_1]\!]_1 \oplus [\![C_2]\!]_1$, $[\![C_2 \circ C_1]\!]_1 = [\![C_2]\!]_1 \circ [\![C_1]\!]_1$ *and:*

$$\left[\!\!\left[\quad \right]\!\!\right]_1 : \mathbb{C} \to \mathbb{C} := |1\rangle \mapsto |1\rangle$$

$$\left[\!\!\left[\quad \boxed{\varphi} \quad \right]\!\!\right]_1 : \mathbb{C} \to \mathbb{C} := |1\rangle \mapsto e^{i\varphi}|1\rangle$$

$$\left[\!\!\left[\quad \right]\!\!\right]_1 : \mathbb{C}^2 \to \mathbb{C}^2 := \begin{array}{lcl} |1,0\rangle & \mapsto & |0,1\rangle \\ |0,1\rangle & \mapsto & |1,0\rangle \end{array}$$

$$\left[\!\!\left[\quad \theta \quad \right]\!\!\right]_1 : \mathbb{C}^2 \to \mathbb{C}^2 := \begin{array}{lcl} |1,0\rangle & \mapsto & c_\theta|1,0\rangle + is_\theta|0,1\rangle \\ |0,1\rangle & \mapsto & is_\theta|1,0\rangle + c_\theta|0,1\rangle \end{array}$$

where $c_\theta = \cos(\theta)$ *and* $s_\theta = \sin(\theta)$.

▶ Remark 4. It is also usual to represent those linear operators as matrices, with $[\![C_1]\!]_1 \oplus [\![C_2]\!]_1 = \left(\begin{array}{c|c} [\![C_1]\!]_1 & 0 \\ \hline 0 & [\![C_2]\!]_1 \end{array}\right)$ and for instance $\left[\!\!\left[\quad \theta \quad \right]\!\!\right]_1 = \begin{pmatrix} c_\theta & is_\theta \\ is_\theta & c_\theta \end{pmatrix}$.

2.2 Simpler equational theory of LOPP

Two distinct **LOPP**-circuits may represent the same quantum evolution. For instance, shifting the phase of a photon by two phase shifters of phase φ_1 and φ_2 is the same as shifting it with one phase $\varphi_1 + \varphi_2$. In order to characterize those equivalences, an equational theory of **LOPP** has been introduced in [10]. In this section, we provide a simpler set of equations in Figure 3. Some of the old equations, given in Figure 4, have been removed, while two Equations (oE2) and (oE3) of Figure 4 have been replaced by the two Equations (E2) and (E3), respectively representing Euler rotations with two and three axes. Previously, those old Euler equations were not directly reversible; while the angles of the right-hand side (RHS) could be uniquely determined by those of the left-hand side (LHS), the inverse was true only with non-trivial constraints, making the equations hardly reversible and not explicitly constructive. More specifically, we made the following simplifications:

- The Equations (b0), (p0) and (pp-b) have been derived and removed from the equational theory.
- A phase has been added in Equation (oE2), so now the LHS can also represent any element of the unitary group $U(2)$. Now the angles of the LHS can be straightforwardly derived without any constraints from the RHS.
- All the phases of Equation (oE3) have been removed. The formulae of the equations are now far simpler, and the equation is now both symmetrical and reversible.

▶ **Definition 5** (**LOPP**-calculus). *Two* **LOPP***-circuits* D, D' *are equivalent according to the rules of the* **LOPP***-calculus, denoted* $\mathbf{LOPP} \vdash D = D'$, *if one can transform* D *into* D' *using the equations given in Figure 3. More precisely,* $\mathbf{LOPP} \vdash \cdot = \cdot$ *is defined as the smallest congruence which satisfies the equations of Figure 3 and the axioms of PROP.*

▶ **Proposition 6** (Soundness of **LOPP**). *For any* **LOPP***-circuits* D *and* D', *if* $\mathbf{LOPP} \vdash D = D'$ *then* $[\![D]\!]_1 = [\![D']\!]_1$.

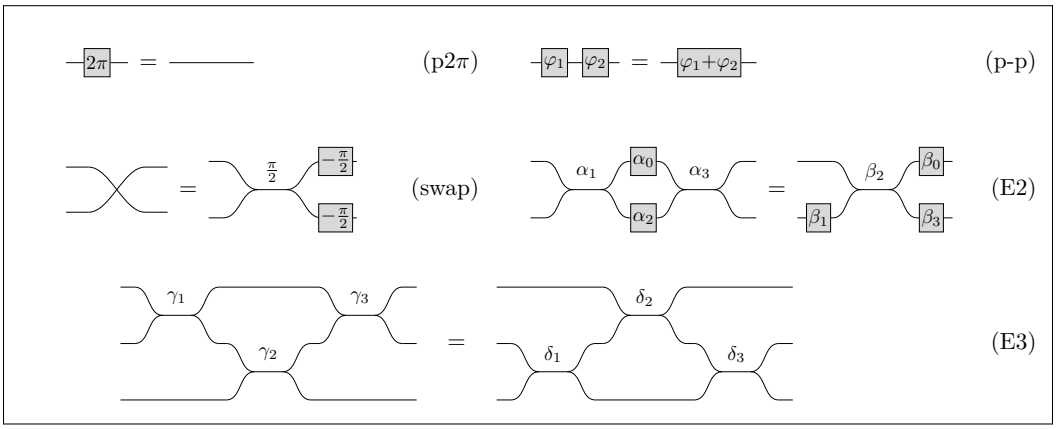


Figure 3 New and minimal equational theory of the **LOPP**-calculus. For any angle of the LHS (resp. RHS) of the Equation (E2) and (E3), there exist angles for the RHS (resp. LHS) such that the equations are sound. The angles of are unique if we restrict $\alpha_0, \alpha_2, \beta_0, \beta_1, \beta_3 \in [0, 2\pi)$, $\alpha_1 \in [0, \frac{\pi}{2}), \alpha_3 \in [0, \pi)$, $\beta_2 \in [0, \frac{\pi}{2}]$, and by taking $\alpha_1 = 0$ if $\alpha_0 - \alpha_2 = 0 \mod \pi$ and $\beta_1 = 0$ if $\beta_2 \in \left\{0, \frac{\pi}{2}\right\}$. The rotations associated with Equations (E2) and (E2) are detailed in the proof of Proposition 6.

Proof. Since semantic equality is a congruence, it suffices to check that for every equation of Figure 3. The soundness of Equations (swap), (p2π) and (p-p) are direct consequences of Definition 3. We can notice that $R_X(-2\theta) = \left[\!\left[\theta \right]\!\right]_1$ and $e^{i\frac{\varphi}{2}} R_Z(\varphi) = \left[\!\left[\varphi \right]\!\right]_1$, where R_X (resp. R_Z) is the rotation operator about the $\hat{x}$ axis (resp. $\hat{z}$ axis) of the Bloch sphere [41]. Any unitary of $U(2)$ can be decomposed into $e^{i\cdot} R_X(\cdot) R_Z(\cdot) R_X(\cdot)$ (resp. $e^{i\cdot} R_Z(\cdot) R_X(\cdot) R_Z(\cdot)$), giving the LHS (resp. RHS) angles of (E2). By transforming the iY-axis into the Y-axis, the three rotations of the LHS (resp. RHS) of (E3) can be seen as three real rotations along the $z - x - z$ real axes (resp. $x - z - x$). The angles are therefore given by the Euler angles [22]. ◀

▶ **Theorem 7** (Completeness of **LOPP**). *For any two* **LOPP***-circuits D and D', if $[\![D]\!]_1 = [\![D']\!]_1$ then* $\mathbf{LOPP} \vdash D = D'$.

Proof. The equational theory of Figure 4 has been proven to be complete in [10]. All equations of Figure 4 can be derived by those of Figure 3. ◀

▶ **Theorem 8** (Minimality). *The equational theory of Figure 3 is minimal for* **LOPP***-circuits, i.e. none of its equations can be derived from the others.*

Proof. We define an alternative interpretation which satisfies all the equations aside from the one we prove to be necessary. In particular:

- (p2π) is the only rule on one wire that changes the sum of the phases.
- (p-p) is the only rule on one wire that can reduce the number of phases different from 2π.
- (swap) is the only rule changing the parity of the number of SWAPs.
- (E2) is the only rule changing the parity of (number of beam splitter + number of SWAPs).

For (E3), here is the sketch of the proof:

- We define an equivalence relation $\sim_\varphi$ on three-wire **LOPP**-circuits.
- We introduce a confluent rewriting procedure that is conserving the relation $\sim_\varphi$, and that is converging to normal forms.

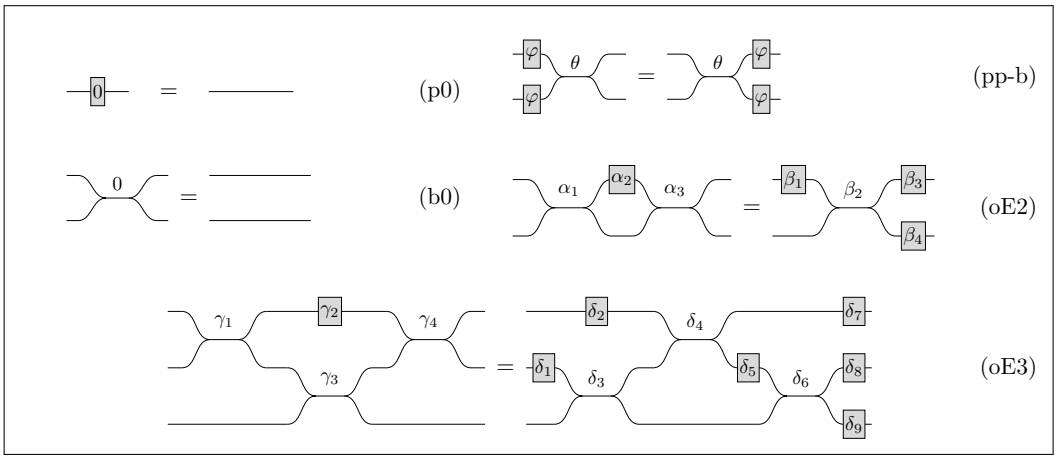


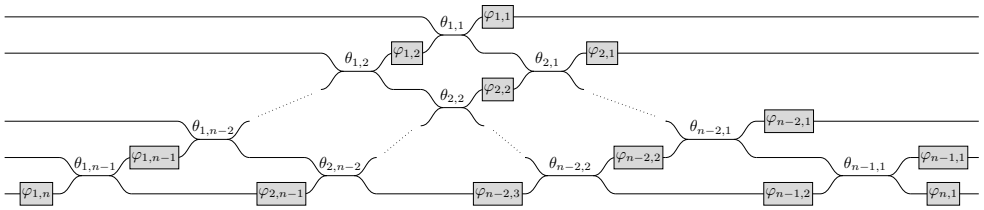
Figure 4 Old axioms of the **LOPP**-calculus that are not in Figure 3. In Equations (oE2) and (oE3), the LHS circuit has arbitrary parameters which uniquely determine the parameters of the RHS circuit. For any $\alpha_i \in \mathbb{R}$, there exist $\beta_j \in [0, 2\pi)$ such that Equation (oE2) is sound, and for any $\gamma_i \in \mathbb{R}$, there exist $\delta_j \in [0, 2\pi)$ such that Equation (oE3) is sound. We can ensure that the angles β_j are unique by assuming that $\beta_1, \beta_2 \in [0, \pi)$ and if $\beta_2 \in \{0, \frac{\pi}{2}\}$ then $\beta_1 = 0$, and we can ensure that the angles δ_j are unique by assuming that $\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6 \in [0, \pi)$. If $\delta_3 \in \{0, \frac{\pi}{2}\}$ then $\delta_1 = 0$, if $\delta_4 \in \{0, \frac{\pi}{2}\}$ then $\delta_2 = 0$, if $\delta_4 = 0$ then $\delta_3 = 0$, and if $\delta_6 \in \{0, \frac{\pi}{2}\}$ then $\delta_5 = 0$. The existence and uniqueness of such β_j and δ_j are given by Lemmas 10 and 11 of [10].

- All the rules of the PROP, (p0), (swap), (p-p) and (E2) also conserve the relation $\sim_\varphi$.
- We conclude that (E3) is necessary, because the LHS and RHS are different normal forms, and therefore can't be transformed from one to the other without (E3). ◀

2.3 Useful triangular forms

In this subsection, we introduce three classes of **LOPP**-circuits, with the following inclusions: ${}_{\tilde{n}}\Diamond_n \subset {}_{\tilde{n}}\triangle_{\tilde{m}} \subset \triangle$. Their shape and properties are illustrated and summarized in Table 2. They are of particular interest as $\triangle$-circuits are the normal forms of the **LOPP**-calculus [10], ${}_{\tilde{n}}\triangle_{\tilde{m}}$-circuits will be used in the normal forms of the $\mathbf{LO}_{\boldsymbol{fi}}$-calculus (Definition 36), and their uniqueness will be proved thanks to use of ${}_{\tilde{n}}\Diamond_n$-circuits (Section 4).

▶ **Definition 9** ($\triangle$-circuits). *A $\triangle$-circuit is a* **LOPP***-circuit with the following shape:*



with $\varphi_{i,j} \in [0, 2\pi), \theta_{i,j} \in [0, \frac{\pi}{2}]$ and the following conditions: $\theta_{i,j} = 0 \Rightarrow (\forall j' > j, \varphi_{i,j'} = \theta_{i,j'} = 0)$ and $\theta_{i,j} = \frac{\pi}{2} \Rightarrow \varphi_{i,j} = 0$. $\theta_{i,j}$ is on the i^{th} right (resp. j^{th} left) diagonal, and on the $(i + j - 1)^{th}$ row of beam splitters.

▶ Remark 10 (Coefficients of $[\![\triangle]\!]_1$). The coefficient $t_{i,j}$ of $[\![\triangle]\!]_1$ is determined by the sum of all the paths from the j^{th} input wire to the i^{th} output wire, where for each path we multiply by a cos (resp. sin) term when the photon is reflected on (resp. transmitted through)

a beam splitter, and by a phase when the path crosses a phase shifter. For instance, $t_{1,2} = \cos(\theta_{1,2})e^{i\varphi_{1,2}}i\sin(\theta_{1,1})e^{i\varphi_{1,1}}$. More generally, we have $t_{i,j} = e^{i\varphi_{i,j}}\cos(\theta_{i,j}) \times q_{i,j} + r_{i,j}$ where $q_{i,j}, r_{i,j}$ are terms depending uniquely on the angles with lower indexes.

▶ **Proposition 11** (Universality and Uniqueness of T). *For any* **LOPP***-circuit D, there exists a unique circuit T in triangular form of Definition 9 such that $[\![D]\!]_1 = [\![T]\!]_1$.*

Proof. The existence is a direct consequence of [46] or the Proposition 26 of [10]. The uniqueness is a consequence of Remark 10 by sequentially showing the uniqueness of $(\varphi_{i,j}, \theta_{i,j})$ in $t_{i,j}$, and by noticing that for any z with $0 < |z| \leq 1$, there are unique $\varphi, \theta \in [0, 2\pi) \times [0, \frac{\pi}{2})$ such that $e^{i\varphi}c_\theta = z$, with $\varphi, \theta = (0, 0)$ for the special case of $z = 0$. More details are provided in Appendix B. ◀

▶ Remark 12. A generic $\triangle$-circuit $T : n \to n$ has $\frac{n(n-1)}{2}$ beam splitters and $\frac{n(n+1)}{2}$ phase shifters, having a total of n^2 different angles, the dimension of the unitary group $U(n)$.

▶ **Definition 13** ($_{\tilde{n}}\triangle_{\tilde{m}}$-circuits). *A* **LOPP***-circuit $\tilde{\triangle} : n + \tilde{n} \to m + \tilde{m}$ is a $_{\tilde{n}}\triangle_{\tilde{m}}$-circuit if:*

1. *$\tilde{\triangle}$ is a $\triangle$-circuit as defined in Definition 9,*
2. *there is no beam splitter or phase shifter fully and directly connected to the $\tilde{n}$ last input wires, ie. $\varphi_{i,j} = \theta_{i,j} = 0$ if $row_{i,j} = i + j - 1 > n$ and there doesn't exist (k, ℓ) such that $k + \ell - 1 = row_{i,j} - 1$, $k < i$ and $\theta_{k,\ell} \neq 0$,*
3. *there is no beam splitter or phase shifter fully and directly connected to the $\tilde{m}$ last output wires, ie. $\varphi_{i,j} = \theta_{i,j} = 0$ if $row_{i,j} = i + j - 1 > m$ and there doesn't exist (k, ℓ) such that $k + \ell - 1 = row_{i,j} - 1$, $k \geq i$ and $\theta_{k,\ell} \neq 0$, and*
4. *there exists one nonzero $\theta_{i,j}$ for each of the last $\max(\tilde{n}, \tilde{m})$ rows.*

The Property 4 is an additional constraint that appears in the normal forms defined in Definition 36. Property 2 and 3 imply the only nonzero angles have indexes $(i \leq m, j \leq n)$, leading to the following proposition, direct consequence of Remark 10 and the proof of Proposition 11.

▶ **Proposition 14** (Uniqueness of $_{\tilde{n}}\triangle_{\tilde{m}}$-circuits on their $m \times n$ submatrix). *For any $_{\tilde{n}}\triangle_{\tilde{m}}$-circuits $\triangle, \triangle' : n + \tilde{n} \to m + \tilde{m}$, if $[\![\triangle]\!]_1 (1 : m, 1 : n) = [\![\triangle']\!]_1 (1 : m, 1 : n)$ then $\triangle = \triangle'$, where $M(1 : k, 1 : \ell)$ is the $k \times \ell$ matrix composed of the first k rows and ℓ columns of M.*

▶ **Definition 15** ($_{\tilde{n}}\Diamond_n$-circuits). *A $_{\tilde{n}}\triangle_{\tilde{m}}$-circuit $\tilde{\triangle} : n + \tilde{n} \to m + \tilde{m}$ is a $_{\tilde{n}}\Diamond_n$-circuit if $\tilde{m} = n$.*

▶ Remark 16. As $\tilde{m} = n$, $_{\tilde{n}}\Diamond_n$-circuits have exactly $\tilde{n} \times n$ beam splitters shaped like in Table 2. Furthermore, their angles are necessarily nonzero, as one zero would imply the rest of the right-diagonal to be zero with Definition 9, contradicting the Property 4. That particular shape and those nonzero properties will be useful in the proofs of Section 4.

3 LO$_{fi}$-calculus

3.1 Fock space

As described in Section 2.1, the state space of one photon with m spatial modes is $\mathbb{C}^m$, as it can be on a superposition of all the different positions. Photons are particles that can bunch and share the same state, so each mode can be occupied by many photons. Furthermore, to observe quantum effects like interferences, we need the photons to be indistinguishable, meaning we can't know *which photon is in which state.*

For those two reasons, the usual way to represent a state with indistinguishable photons is by using the occupation number representation, where we indicate *"how many photons are there in each state"*. We consider the basis states $\Phi_m \coloneqq \{|n_1, n_2, \dots, n_m\rangle, (n_1, n_2, \dots, n_m) \in \mathbb{N}^m\}$ [1], denoted as *Fock states*. The state $|n_1, n_2, \dots, n_m\rangle$ represents a configuration where n_i photons occupy the i^{th} mode. The space of possible many-photon states over m modes, the *bosonic (symmetrical)* Fock space and denoted as $\mathcal{B}_m$, is defined as follows.

▶ **Definition 17** (Fock space). *We define the Hilbert space $\mathcal{B}_m$ as the Hilbert completion $\ell^2(\Phi_m)$ equipped with the bra-ket inner product $\langle\cdot|\cdot\rangle$, with the convention $\mathcal{B}_0 = \mathbb{C}$.*

▶ Remark 18. $\mathcal{B}_1$ contains states that are an *infinite* superposition of basis states, like the coherent states $|\alpha\rangle = e^{-\frac{|\alpha|^2}{2}} \sum_{k=0}^{\infty} \frac{\alpha^k}{k!} |k\rangle$. We can note that $\mathcal{B}_m$ is isomorphic to $\ell^2(\mathbb{N}^m)$.

To describe the space of the auxiliary sources, we consider a sub vector space of $\mathcal{B}_m$.

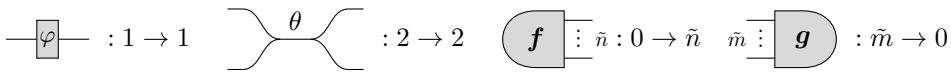
▶ **Definition 19** (Subspace of the Fock space: $\mathcal{B}_m^{\text{pre}}$). *We define the pre-Hilbert space $\mathcal{B}_m^{pre}$ as the linear span of Φ_m equipped with the bra-ket inner product $\langle\cdot|\cdot\rangle$, with the convention $\mathcal{B}_0^{pre} = \mathbb{C}$.*

▶ Remark 20. $\mathcal{B}_m^{\text{pre}}$ only contain states that are *finite* linear combination of the Fock basis states. In particular, the coherent states are not included. We can note that $\mathcal{B}_1^{\text{pre}}$ is isomorphic to c_{00}, i.e. the space of eventually zero sequences.

▶ **Definition 21** ($\mathcal{B}_{\tilde{m}}^{*\text{pre}}$). *We define the pre-Hilbert space $\mathcal{B}_{\tilde{m}}^{*pre}$ as the linear span of $\{\langle n_1, \dots, n_{\tilde{m}}|, (n_1, \dots, n_{\tilde{m}}) \in \mathbb{N}^{\tilde{m}}\}$.*

3.2 Syntax and many-photon semantics

▶ **Definition 22** ($\mathbf{LO}_{fi}$-calculus). *$\mathbf{LO}_{fi}$ is the PROP of $\mathbf{LO}_{fi}$-circuits generated by*



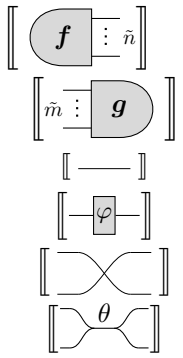
where $\varphi, \theta \in \mathbb{R}$, and $\boldsymbol{f}$ (resp. $\boldsymbol{g}$) is a state in $\mathcal{B}_{\tilde{n}}^{pre}$ (resp. in $\mathcal{B}_{\tilde{m}}^{*pre}$) with $\tilde{n}, \tilde{m} \in \mathbb{N}^+$.

▶ Remark 23. In string diagrams, a process $0 \to \tilde{n}$ (resp. $\tilde{m} \to 0$) is called a state (resp. an effect). We will keep the source (resp. detector) terms to be consistent with their physical representation. A process $0 \to 0$ is called a scalar.

▶ Remark 24. The choice of those generators is discussed in Appendix C.

▶ **Definition 25** (Conventions for the notations). *Bold terms will always be vectors. In particular $\boldsymbol{f}, \boldsymbol{f}_k$ (resp. $\boldsymbol{g}, \boldsymbol{g}_\ell$) will always represent a ket (resp. a bra). Bold integers $\boldsymbol{k}$ (resp. $\boldsymbol{\ell}$) will represent $|k\rangle$ (resp. $\langle\ell|$) in the sources (resp. detectors). The summation term $\sum$ will often be omitted, the index being implicitly the sum index. Note that for clarity, the summation term won't be omitted in Figure 6, and for conciseness, they will be omitted in Figure 5. For instance $\boldsymbol{f} = \sum_{k\in\mathcal{K}} |\boldsymbol{f}_k\rangle |k\rangle$ will simply be noted as $\boldsymbol{f}_k \otimes \boldsymbol{k}$. $|.\rangle$ (resp. $\langle.|$) represents an arbitrary ket (resp. bra) on one mode. $|...\rangle$ (resp. $\langle...|$) represents an arbitrary ket (resp. bra) for an arbitrary number of modes, representing an arbitrary scalar when the number of modes is zero. Those are used to omit terms when the specific value of those terms are not of interest, as in some equations of Figure 5. For the zero vector $\boldsymbol{f} = 0$ (resp. $\boldsymbol{g} = 0$), as there is no term in the sum, we chose to represent it with*  *(resp. an empty detector* *). Note it is different from*  *(resp.* *) representing the nonzero vector $|0\rangle$ (resp. $\langle 0|$).*

▶ **Definition 26.** *Let $C\colon n \to m$ a $\mathbf{LO}_{fi}$-circuit, let $[\![C]\!] : \mathcal{B}_n \to \mathcal{B}_m$ be the linear map inductively defined as $[\![C_2 \circ C_1]\!] = [\![C_2]\!] \circ [\![C_1]\!]$, $[\![C_1 \otimes C_2]\!] = [\![C_1]\!] \otimes [\![C_2]\!]$ and:*



$0 \to \mathcal{B}_{\tilde{n}}$	$\boldsymbol{f} \in \mathcal{B}^{pre}_{\tilde{n}}$
$\mathcal{B}_{\tilde{m}} \to 0$	$\boldsymbol{g} \in \mathcal{B}^{*pre}_{\tilde{m}}$
$\mathcal{B}_1 \to \mathcal{B}_1$	$\lvert k\rangle \mapsto \lvert k\rangle$
$\mathcal{B}_1 \to \mathcal{B}_1$	$\lvert k\rangle \mapsto P_\varphi(\lvert k\rangle)$
$\mathcal{B}_2 \to \mathcal{B}_2$	$\lvert k_1, k_2\rangle \mapsto \lvert k_2, k_1\rangle$
$: \mathcal{B}_2 \to \mathcal{B}_2$	$\lvert k_1, k_2\rangle \mapsto B_\theta(\lvert k_1, k_2\rangle)$

where $B_\theta(|k_1,k_2\rangle) \coloneqq \sum_{\ell_1+\ell_2=k_1+k_2} \sqrt{\frac{\ell_1!\ell_2!}{k_1!k_2!}} \sum_{\substack{p+q=\ell_1 \\ \delta=p-q}} \binom{k_1}{p}\binom{k_2}{q} c_\theta^{k_2+\delta} (is_\theta)^{k_1-\delta} \,|\ell_1,\ell_2\rangle$ *and* $P_\varphi(|k\rangle) \coloneqq e^{ik\varphi}\,|k\rangle$, *with the convention* $\binom{k}{k'} = 0$ *for* $k < k'$.

We can check that P_φ and B_θ are unitary operators [1] and are therefore well-defined on the Hilbert space by continuity and linearity. One can also look at [36] for a more physical interpretation of where the formulas come from.

▶ Remark 27 (Hermitian conjugate). We have $P_\varphi^\dagger = P_{-\varphi}$ and $B_\theta^\dagger = B_{-\theta}$, where $\dagger$ is the Hermitian conjugate. Therefore, $\langle \ell|\, P_\varphi = (P_{-\varphi}\,|\ell\rangle)^\dagger$ and $\langle \ell_1,\ell_2|\, B_\theta = (B_{-\theta}\,|\ell_1,\ell_2\rangle)^\dagger$.

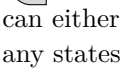
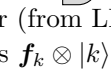
3.3 Equational theory of $\mathbf{LO}_{fi}$

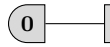
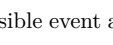
Similarly to Section 2.2, we introduce an equational theory for $\mathbf{LO}_{fi}$ in Figure 5.

▶ **Definition 28 ($\mathbf{LO}_{fi}$-calculus).** *Two $\mathbf{LO}_{fi}$-circuits C, C' are equivalent according to the rules of the $\mathbf{LO}_{fi}$-calculus, denoted $\mathbf{LO}_{fi} \vdash C = C'$, if one can transform C into C' using the equations given in Figure 5.*

▶ Remark 29. The Equation (p-p) is not present for it can be derived with the Equations (p2π), (E2) and (s0-0d), alongside with Equation (b0), that can be derived with the rules of the PROP, and the Equations (swap) and (E2). Note that the Equaiton (h2) can be considered an equation of *diagrams with holes*.

▶ **Proposition 30** (Soundness). *For any two $\mathbf{LO}_{fi}$-circuits C and C', if $\mathbf{LO}_{fi} \vdash C = C'$ then $[\![C]\!] = [\![C']\!]$.*

Proof. Since semantic equality is a congruence, it suffices to check the soundness for every equation of Figure 5, which follows from Proposition 6 and Definition 26. Informally, Axiom (zero) means that if there is the scalar[5] 0, then the semantics of all the circuit ($X \otimes 0 = 0$) is the null function. We can therefore nullify the other wires with the zeros  and  . This rule is specifically used for Remark 38. Axiom (s-0d) means we can either (from LHS to RHS) project on $|0\rangle$ on the last mode or (from right to left) add any states $\boldsymbol{f}_k \otimes |k\rangle$ with $k \neq 0$ as they are trivially orthogonal and cancelling. Axiom (h2) means we can *shift* any function $h : \mathcal{B}_2^{\text{pre}} \to \mathcal{B}_2^{\text{pre}}$ from left to right where there are identity wires, direct consequence of the associativity: $\langle \ell_1,\ell_2|\,(h\,|k_1,k_2\rangle) = (\langle \ell_1,\ell_2|\,h)\,|k_1,k_2\rangle$. The rules (dd), (b-d), (p-d) and (s0-d) are respectively the duals of (ss), (s-b), (s-p) and (s-0d). ◀

[5]  is an impossible event and is one way to represent the scalar $0 = \langle 1|0\rangle =$  .

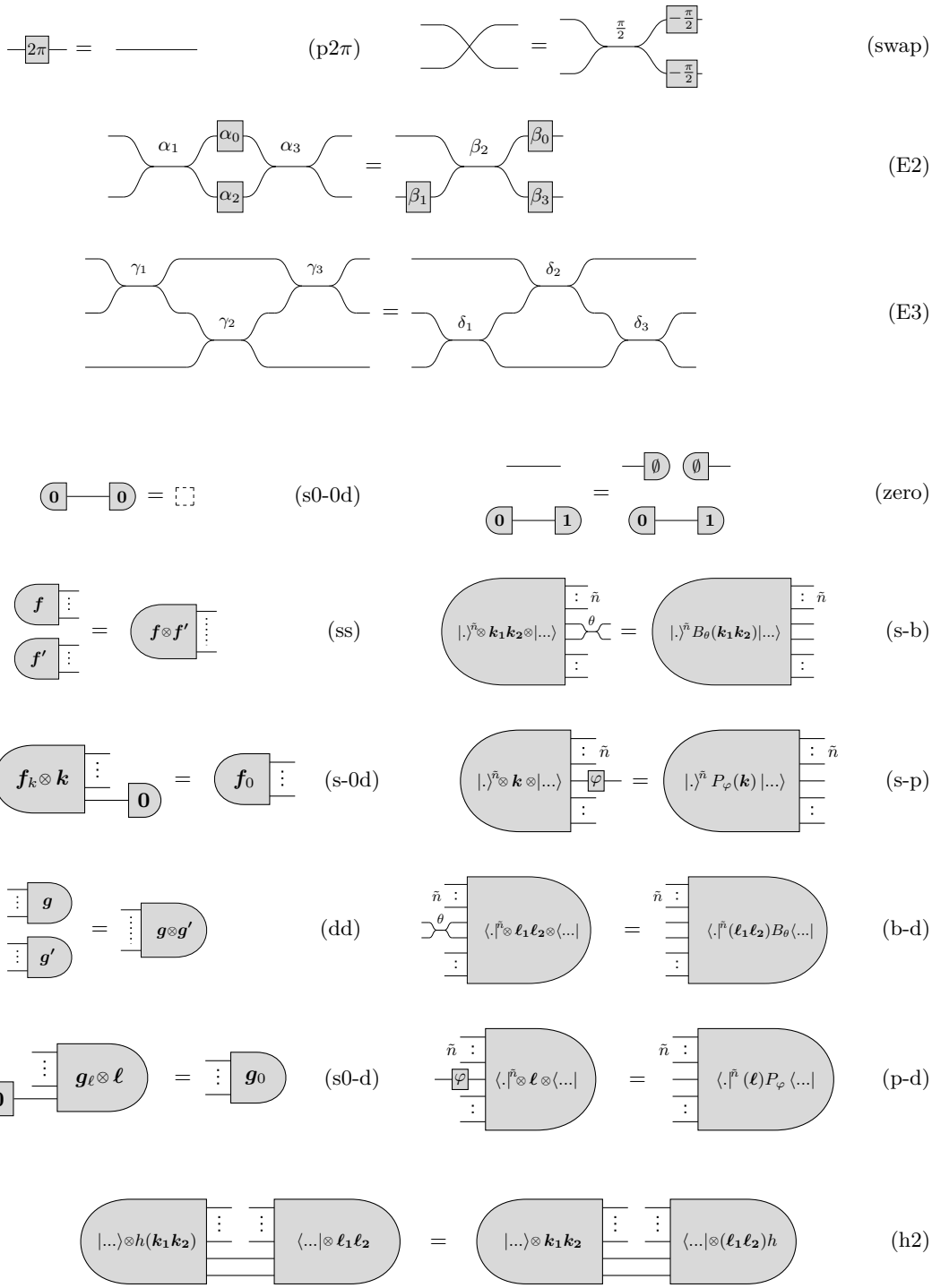


Figure 5 Axioms of the $\mathbf{LO}_{fi}$-calculus. The angles of (E2) and (E3) are the same as in the axioms of the **LOPP**-calculus (Figure 3). h is any linear function $\mathcal{B}_2^{\mathrm{pre}} \to \mathcal{B}_2^{\mathrm{pre}}$. The conventions for $\{\emptyset, |.\rangle, |...\rangle, \langle.|, \langle...|\}$, and the omitted sums are detailed in Definition 25. The interpretations of the axioms are given in Proposition 30.

▶ **Theorem 31** (Completeness). *For any two* $\mathbf{LO}_{fi}$*-circuits* C *and* C'*, if* $[\![C]\!] = [\![C']\!]$ *then* $\mathbf{LO}_{fi} \vdash C = C'$.

Proof. The proof is in Section 4.4, direct consequence of the uniqueness of the normal forms of Section 4. ◀

4 Unique normal forms leading to the completeness of the $\mathbf{LO}_{fi}$-calculus

We introduce a set of oriented rewriting rules in Section 4.1, that converge to a set of $\mathbf{LO}_{fi}$-circuits with specific shape and properties, defined in Section 4.2. The proof of their uniqueness is summarised in Section 4.3. As a direct collorary of the uniqueness of the normal forms, we prove the completeness of the $\mathbf{LO}_{fi}$-calculus in Section 4.4.

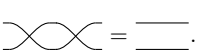
4.1 Deterministic rewriting procedure

A strongly normalising rewriting system, i.e. terminating to normal forms, has been introduced in [10] for **LOPP**-circuits. We mainly reuse all the rules, alongside additional rules to now take into account the sources and the detectors.

▶ **Definition 32** (Rewriting system). *Our rewriting system is defined on the PRO*[6] *of* $\mathbf{LO}_{fi}$*-circuits with the rules of Figure 6.*

We can check that all the rules are sound, and have the following meaning:

- The rules 1-11 are either the same or just slightly different from the rules described in [10]. With those rules, the $\mathbf{LOPP}^{PRO}$-circuits will converge to the triangular $\triangle$-circuits defined in Section 2.3.
- The rule 12 removes any vector $|f_{k'}\rangle \otimes |k'\rangle$ in the sources that is trivially cancelling with the detector on the connected last wire, meaning that $\langle g_{k'}| = 0$, i.e. that $k' \notin \mathcal{L}$.
- The rule 13 removes any $\langle g_{\ell'}| \otimes \langle \ell'|$ in the detectors that is trivially cancelling with the source on the connected last wire, meaning that $|f_{\ell'}\rangle = 0$, i.e. that $\ell' \notin \mathcal{K}$.
- The rule 14 allows, without changing the semantics, to transfer the generic coefficients from the detectors to the sources. Specifically, any term of the form $\sum_\ell \xi_\ell \langle \mathcal{N}_{\tilde{m}}(L)| \langle \ell|$ will be rewritten to $\langle \mathcal{N}_{\tilde{m}}(L)| \langle L|$. The coefficients ξ will be in the source, as $|f_L\rangle |L\rangle$ will be rewriten to $\left(\sum_{i\in\mathcal{K}} \xi_i |f_i\rangle\right) |L\rangle$. At the end and by repeating this rule, there won't be any *degree of freedom* in the detectors, and $\boldsymbol{g} = \sum_{\ell\in\mathcal{L}} \langle \mathcal{N}_{\tilde{m}}(\ell)| \langle \ell|$. The condition $(\xi_L \neq 1) \vee (\exists \ell \neq L, \xi_\ell \neq 0)$ is there to ensure that the rule is only used once for each L, and only when it's necessary.
- The rule 15 uses the bijection $\mathcal{N}_2 : \mathbb{N} \to \mathbb{N}^2$ to remove one identity wire, by just relabelling the indexes in the sources and detectors. Note that one identity wire will always remain at the end.
- The oriented rule (from left to right) coming from the axioms (ss) and (dd) merge all the sources and detectors together.
- The oriented rule (from left to right) coming from the axioms (s-b), (s-p), (b-d) and (p-d) reduce the number of phase shifters and beam splitters as much as possible, by making them be *absorbed* into the sources and detectors.

[6] This is similar to [10], to prevent any deformation of the form  = .

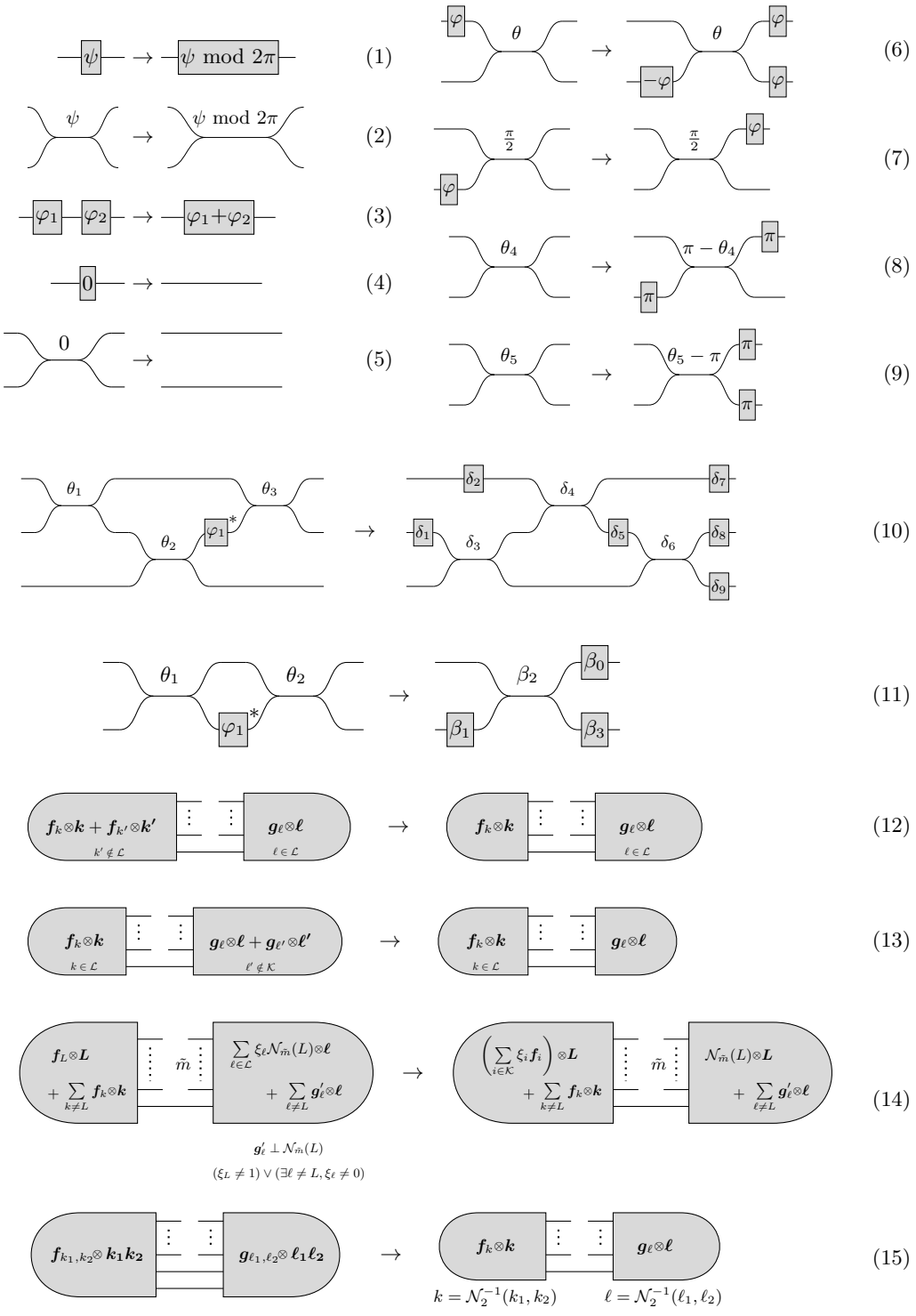


Figure 6 Rewriting system of the $\mathbf{LO}_{fi}$-calculus, alongside with the oriented version, from the LHS to the RHS, of the axioms (ss), (s-b), (s-p), (dd), (b-d), and (p-d). $\psi \in \mathbb{R} \setminus [0, 2\pi)$, $\varphi, \varphi_1, \varphi_2 \in (0, 2\pi)$, $\theta_4 \in (\frac{\pi}{2}, \pi)$, $\theta_5 \in [\pi, 2\pi)$, $\theta, \theta_1, \theta_2, \theta_3 \in (0, \pi)$. $-\boxed{\varphi}^{*}-$ denotes either $-\boxed{\varphi}-$ or $---$. The angles of the RHS of (11) and (10) are given by [10]. $\mathcal{N}_m : \mathbb{N} \to \mathbb{N}^m$ is a bijection arbitrary chosen to be $\mathcal{N}_m^{-1} \coloneqq \mathcal{N}_2^{-1} \circ \mathcal{N}_{m-1}^{-1}$ for $m > 2$, where $\mathcal{N}_2^{-1}(\ell, \ell') \coloneqq \frac{1}{2}(\ell + \ell)(\ell + \ell' + 1) + \ell'$ is the Cantor pairing function and $\mathcal{N}_1$ is the identity. By convention, the summation index is $k \in \mathcal{K}$ for the sources and $\ell \in \mathcal{L}$ for detectors, aside from the rule (14) where the sum is explicit for clarity.

▶ **Definition 33** (Inputs of the rewriting system). *For convenience, the inputs of the rewriting procedures are $\mathbf{LO}_{fi}$-circuits with at least one identity wire connecting sources and generators, and where all the sources (resp. detectors) are on the bottom left (resp. right).*

▶ Remark 34. Note that choice is not restrictive, as the identity wire can always be added with Axiom (s0-0d), and the sources and detectors can be placed at the desired position, without changing the semantics, with the rules of PROP and by adding SWAPs.

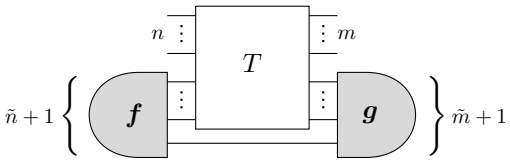
▶ **Lemma 35.** *If C_1 rewrites to C_2 using the rules of Figure 6, then $\mathbf{LO}_{fi} \vdash C_1 = C_2$.*

Proof. As an illustration, we show how we can derive the rule (14) in Appendix D. ◀

4.2 Normal forms of the $\mathbf{LO}_{fi}$-calculus

Formally with the rules of Figure 6 and informally with their meaning described in Section 4.1, we can show that an irreducible form is a $\mathbf{LO}_{fi}$-circuit defined as follows:

▶ **Definition 36** (Normal form). *The normal forms of any nonzero $\mathbf{LO}_{fi}$-circuits are denoted $N(T, \boldsymbol{f}) : n \to m$ and are of the form:*



where:

- $\boldsymbol{f}$ *is a nonzero generic state of* $\mathcal{B}^{pre}_{\tilde{n}+1}$.
- $\boldsymbol{g} = \sum_{\ell \in \mathcal{K}} \langle \mathcal{N}_m(\ell)| \otimes \langle \ell|$, *where* $\mathcal{N}_m : \mathbb{N} \to \mathbb{N}^m$ *is a bijection defined in Figure 6 and* $\mathcal{K}$ *is the nonempty finite set* $\{k \in \mathbb{N} \mid \boldsymbol{f}_k \neq 0\}$ *of* $\boldsymbol{f} = \sum_{k \in \mathcal{K}} \boldsymbol{f}_k \otimes |k\rangle$, *with the convention* $\mathcal{K} = \{0\}$ *if* $\tilde{n} = 0$ *or* $\tilde{m} = 0$.
- $T : n + \tilde{n} \to m + \tilde{m}$ *is a* ${}_{\tilde{n}}\triangle_{\tilde{m}}$*-circuit as defined in Definition 13.*

▶ Remark 37. If $\tilde{n} = \tilde{m} = 0$, then the normal form is a normal form of **LOPP** (can be $\boxed{\ }$ for $n = m = 0$) tensored with the scalar $\boxed{\alpha|0\rangle}\!\!-\!\!\boxed{\langle 0|}$ which has the semantics of a global scalar $\alpha \in \mathbb{C}$.

▶ Remark 38. We could also consider the particular case of $\boldsymbol{f} = 0$, i.e. $\mathcal{K} = \emptyset$, where $[\![N]\!] : \mathcal{B}_n \to \mathcal{B}_m$ is the null function. In that case, $N : n \to m$ can be written to $(\ \boxed{\emptyset}\!- \ )^{\otimes m} \circ (\ -\!\boxed{\emptyset}\ )^{\otimes n}$, which is a more fitted form for representing the null function.

▶ **Lemma 39** (Strongly normalising). *The rewriting system of Figure 6 is strongly normalising.*

Proof. We introduce a ranking function $(x_1, \ldots, x_6) \in \mathbb{N}^6$, where each component of the tuple is determined by properties of the circuit, like the number of beam splitter with angles out of $[0, 2\pi)$, the number of sources and detectors, or the number of identity wires connecting them. One nontrivial component is x_6, that we explicit here.

Let note the generic terms in the sources as $\boldsymbol{f} = \sum \alpha_{k_1,\ldots,k_{\tilde{n}+1}} |k_1, \ldots, k_{\tilde{n}+1}\rangle$ and in the detectors as $\boldsymbol{g} = \sum \beta_{\ell_1,\ldots,\ell_{\tilde{m}+1}} |\ell_1, \ldots, \ell_{\tilde{m}+1}\rangle$. We define:

$$x_6 \coloneqq \sum_{\boldsymbol{f} \in \text{sources}} C_1(\boldsymbol{f}) + \sum_{\boldsymbol{g} \in \text{detectors}} (2C_2(\boldsymbol{g}) - C_3(\boldsymbol{g}))$$

with $C_1(\boldsymbol{f}) \coloneqq \#\{\alpha_{k_1,\dots,k_{\tilde{n}+1}} \neq 0\}$, $C_2(\boldsymbol{g}) \coloneqq \#\{\beta_{\ell_1,\dots,\ell_{\tilde{m}+1}} \neq 0\}$, and $C_3(\boldsymbol{g}) \coloneqq \#\{\beta_{\mathcal{N}_{\tilde{m}}(L),L} = 1, L \in \mathbb{N}\}$. The proof to show that the rule (14) strictly decreases x_6 is the following. Let us consider the two cases: $(\xi_L \neq 1) \wedge (\forall \ell \neq L, \xi_\ell = 0)$ and $(\xi_L = 1) \wedge (\exists \ell \neq L, \xi_\ell = 0)$. The first case doesn't change C_1 and C_2, but the term $-C_3$ strictly decreases by 1. The second case doesn't change C_3, and the increase of C_1, i.e. the amount of new terms in $\boldsymbol{f}$, is bounded by $\#\{\xi_i \neq 0, i \neq L\}$, the number of terms removed in $\boldsymbol{g}$, which is the exact decrease of C_2. Therefore, $C_1 + 2C_2$ decreases by at least $\#\{\xi_i \neq 0, i \neq L\} > 0$. We can conclude that the rule (14) strictly decreases x_6. ◀

Now that the normal forms are well-defined, it remains to prove their uniqueness, which is the purpose of the Section 4.3.

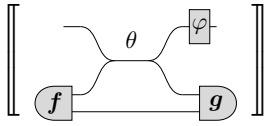
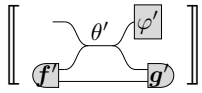
▶ **Lemma 40** (Uniqueness of the Normal Forms). *If two* $\mathbf{LO}_{\boldsymbol{fi}}$*-circuits* N *and* N' *in normal forms are such that* $[\![N]\!] = [\![N']\!]$*, then* $N = N'$*.*

4.3 The normal forms are unique: sketch of the proof

Let $N(T, \boldsymbol{f})$ be a normal form. In order to prove the uniqueness of T and $\boldsymbol{f}$, we proceed with the following steps.

1. We first show that T is unique.
2. We introduce a set of operators $\Omega^{\boldsymbol{i},\boldsymbol{j}}(T)$, such that $[\![N]\!] = \sum_{\boldsymbol{i},\boldsymbol{j}} \omega_{\boldsymbol{i},\boldsymbol{j}} \Omega^{\boldsymbol{i},\boldsymbol{j}}(T)$. We show the $\omega_{\boldsymbol{i},\boldsymbol{j}}$ to be canonically and uniquely associated with the coefficients of $\boldsymbol{f}$.
3. We introduce a set of operators $\Delta^{\boldsymbol{u},\boldsymbol{v}}(T)$, that have very convenient properties and that we show to be linearly independent.
4. We give an isomorphism between the Ω and Δ operators, therefore proving the linear independence of the $\Omega^{\boldsymbol{i},\boldsymbol{j}}(D)$, and proving the uniqueness of the coefficients of $\boldsymbol{f}$.

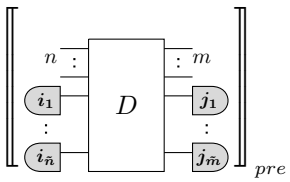
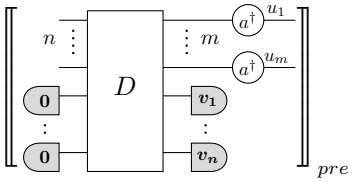
▶ **Lemma 41** (Uniqueness of T). *For any two normal forms* $N(T, \boldsymbol{f})$ *and* $N'(T', \boldsymbol{f'})$*, if* $[\![N]\!] = [\![N']\!]$ *then* $T = T'$*.*

Proof. For any nonzero $W =$  and $W' =$  , we first show that:

$$(\theta, \varphi) \neq (\theta', \varphi') \Rightarrow \exists k \in \mathbb{N}, \lim_{n \to \infty} \frac{\langle n+k| W |n\rangle}{\langle n+k| W' |n\rangle} \neq 1.$$

As $W = W'$, the limit of the ratio is necessarily equal to 1; the parameters can't be different and are therefore equal. The proof relies purely on the semantics defined in Definition 26. We then prove the uniqueness of D by induction on the min (number of inputs, number of outputs). ◀

▶ **Definition 42** (Ω and Δ morphisms). *For any* $\mathbf{LOPP}$*-circuit* D*,* $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{u}, \boldsymbol{v}) \in (\mathbb{N}^{\tilde{n}}, \mathbb{N}^{\tilde{n}}, \mathbb{N}^{\tilde{m}}, \mathbb{N}^{m})$ *we define* $\Omega^{\boldsymbol{i},\boldsymbol{j}}(D) : n \to m$ *and* $\Delta^{\boldsymbol{u},\boldsymbol{v}}(D)$ *as:*

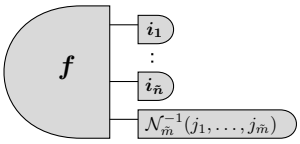
$\Omega^{\boldsymbol{i},\boldsymbol{j}}(D) \coloneqq$  $\Delta^{\boldsymbol{u},\boldsymbol{v}}(D) \coloneqq$ 

where $\hat{a}^\dagger : |k\rangle \mapsto \sqrt{k+1}\,|k+1\rangle$ *is the creation operator and* $[\![\cdot]\!]_{pre} \coloneqq [\![\cdot]\!]|_{pre}$ *is the restriction of* $[\![\cdot]\!]$ *to* $\mathcal{B}^{pre}$*.*

▶ Remark 43. All the proofs regarding the Ω and Δ morphisms only consider the semantics on $[\![.]\!]_{pre}$. That ensures the soundness of the proofs involving the unbounded operator $\hat{a}^\dagger$, as now all sums will be finite.

We give here two propositions that are the core of the proofs.

▶ **Proposition 44** (Unique Ω-decomposition of the normal forms). *For any ${}_{\tilde{n}}\triangle_{\tilde{m}}$-circuit $T : n + \tilde{n} \to m + \tilde{m}$ and finite set $\{\omega_{\boldsymbol{i},\boldsymbol{j}}, (\boldsymbol{i},\boldsymbol{j}) \in (\mathbb{N}^{\tilde{n}}, \mathbb{N}^{\tilde{m}})\}$, there exists an unique normal form $N(T, \boldsymbol{f}) : n \to m$, such that $[\![N]\!]_{pre} = \sum_{\boldsymbol{i},\boldsymbol{j}\in(\mathcal{I},\mathcal{J})} \omega_{\boldsymbol{i},\boldsymbol{j}} \Omega^{\boldsymbol{i},\boldsymbol{j}}(T)$.*

Proof. It follows from the linearity of $[\![.]\!]_{pre}$ and that $\omega_{\boldsymbol{i},\boldsymbol{j}} =$ 

. ◀

▶ **Proposition 45** (Threshold properties of the Δ-morphisms). *For any ${}_{\tilde{n}}\Diamond_n$-circuit $\tilde{\Diamond} : n+m \to n+m$ and $(\boldsymbol{u},\boldsymbol{v}) \in (\mathbb{N}^n, \mathbb{N}^m)$, $\langle \boldsymbol{y}| \, \Delta^{\boldsymbol{u},\boldsymbol{v}}(\tilde{\Diamond}) \, |\boldsymbol{x}\rangle$ is nonzero for $(\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{v},\boldsymbol{u})$ and is zero if $(\boldsymbol{x} \prec_r \boldsymbol{v}) \vee (\boldsymbol{y} \prec_r \boldsymbol{u})$, where $\prec_r$ is the reverse lexicographical order, i.e. $\boldsymbol{y} \prec_r \boldsymbol{v}$ if there exists k such that $y_n = v_n, \ldots, y_{k+1} = v_{k+1}$ and $y_k < v_k$.*

Proof. It is a consequence of the shape of the ${}_{\tilde{n}}\Diamond_n$-circuits (Definition 15), and the properties of $\Delta^{\boldsymbol{u},\boldsymbol{v}}$. As there is no photon in the auxiliary sources, the input needs a certain number of photons for them to be detected in the auxiliary detectors. Similarly, as we create photons at the output with the creation operators $\hat{a}^\dagger$, the output needs a certain number of photons. ◀

The linear independence of Δ will be a consequence of Proposition 45 and a decomposition of the Ω with Δ morphisms will give the independence of the Ω, thus the uniqueness of the $\omega_{\boldsymbol{i},\boldsymbol{j}}$, and therefore the uniquess of the normal forms with Proposition 44.

4.4 Completeness of the $\mathbf{LO}_{fi}$-calculus: Proof of Theorem 31

Let C, C' two $\mathbf{LO}_{fi}$-circuits such that $[\![C]\!] = [\![C']\!]$. They can be rewritten to normal forms by Lemma 35: $\mathbf{LO}_{fi} \vdash C = N$ and $\mathbf{LO}_{fi} \vdash C' = N'$. By soundness of $\mathbf{LO}_{fi}$, we have $[\![N]\!] = [\![C]\!] = [\![C']\!] = [\![N']\!]$ thus $[\![N]\!] = [\![N']\!]$. By Lemma 35, the normal forms are unique. Therefore, $N = N'$ and we have $\mathbf{LO}_{fi} \vdash C = N = N' = C'$, thus $\mathbf{LO}_{fi} \vdash C = C'$, proving the completeness of the $\mathbf{LO}_{fi}$-calculus. ◀

5 Outlook

The formalism of the $\mathbf{LO}_{fi}$-calculus helped to find normal forms for linear optical circuits, and the new operators introduced in Section 4 were particularly relevant for proving their uniqueness. It is an open problem to know if those normal forms and new operators can have further applications in simulation, compilation or the synthesis of linear optical circuits, or even broader reach as the **LOPP**-calculus had for quantum circuits [11]. As those normal forms make only sense with finite states, it is also an open problem to determine whether normal forms exist in the infinite case, let alone their uniqueness.

References

1 Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the 43th Annual ACM Symposium on Theory of Computing (STOC)*, STOC '11, pages 333–342, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/1993636.1993682.

2 Samson Abramsky and Bob Coecke. A categorical semantics of quantum protocols. In *19th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 415–425, 2004. doi:10.1109/LICS.2004.1319636.

3 Samson Abramsky and Bob Coecke. Categorical quantum mechanics, 2008. arXiv:0808.1023.

4 Stefan Ataman. A graphical method in quantum optics. *Journal of Physics Communications*, 2(3):035032, March 2018. doi:10.1088/2399-6528/aab50f.

5 Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski, and John van de Wetering. There and back again: A circuit extraction tale. *Quantum*, 5:421, March 2021. doi:10.22331/q-2021-03-25-421.

6 Sara Bartolucci, Patrick Birchall, Hector Bombin, Hugo Cable, Chris Dawson, Mercedes Gimeno-Segovia, Eric Johnston, Konrad Kieling, Naomi Nickerson, Mihir Pant, Fernando Pastawski, Terry Rudolph, and Chris Sparrow. Fusion-based quantum computation. *Nature Communications*, 14(1):912, 2023. doi:10.1038/s41467-023-36493-1.

7 Daniel E. Browne and Terry Rudolph. Resource-efficient linear optical quantum computation. *Phys. Rev. Lett.*, 95:010501, June 2005. doi:10.1103/PhysRevLett.95.010501.

8 Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews*, 6(2), May 2019. doi:10.1063/1.5088164.

9 Alexandre Clément, Noé Delorme, and Simon Perdrix. Minimal equational theories for quantum circuits, 2023. arXiv:2311.07476, doi:10.48550/arXiv.2311.07476.

10 Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix, and Benoît Valiron. Lov-calculus: A graphical language for linear optical quantum circuits. In *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:16, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2022.35.

11 Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix, and Benoît Valiron. A complete equational theory for quantum circuits. In *Proceedings of the 38th Annual ACM IEEE Symposium on Logic in Computer Science (LICS)*, Boston, MA, USA, 2023. arXiv:2206.10577.

12 Alexandre Clément and Simon Perdrix. PBS-calculus: A graphical language for coherent control of quantum computations. In Javier Esparza and Daniel Kráľ, editors, *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:14, Dagstuhl, Germany, August 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2020.24.

13 Bob Coecke and Ross Duncan. Interacting quantum observables: Categorical algebra and diagrammatics. *New Journal of Physics*, 13(4):043016, April 2011. doi:10.1088/1367-2630/13/4/043016.

14 N. Coste, D. A. Fioretto, N. Belabas, S. C. Wein, P. Hilaire, R. Frantzeskakis, M. Gundin, B. Goes, N. Somaschi, M. Morassi, A. Lemaître, I. Sagnes, A. Harouri, S. E. Economou, A. Auffeves, O. Krebs, L. Lanco, and P. Senellart. High-rate entanglement between a semiconductor spin and indistinguishable photons. *Nature Photonics*, 17(7):582–587, 2023. doi:10.1038/s41566-023-01186-0.

15 Giovanni de Felice and Bob Coecke. Quantum linear optics via string diagrams. *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, 394:83–100, November 2023. doi:10.4204/eptcs.394.6.

16 Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang, and Bob Coecke. Light-matter interaction in the zxw calculus. *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, 384:20–46, August 2023. doi:10.4204/eptcs.384.2.

17 Grégoire de Gliniasty, Paul Hilaire, Pierre-Emmanuel Emeriau, Stephen C. Wein, Alexia Salavrakos, and Shane Mansfield. A spin-optical quantum computing architecture, 2024. arXiv:2311.05605.

18 Marc de Visme and Renaud Vilmart. Minimality in finite-dimensional zw-calculi, 2024. arXiv:2401.16225.

19 Ross Duncan and Maxime Lucas. Verifying the steane code with quantomatic. *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, 171:33–49, December 2014. doi:10.4204/eptcs.171.4.

20 Suren A. Fldzhyan, Mikhail Yu. Saygin, and Sergei P. Kulik. Compact linear optical scheme for bell state generation. *Physical Review Research*, 3(4):043031, 2021. doi:10.1103/PhysRevResearch.3.043031.

21 Liam Garvie and Ross Duncan. Verifying the smallest interesting colour code with quantomatic. *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, 266:147–163, February 2018. doi:10.4204/eptcs.266.10.

22 Herbert Goldstein, Charles Poole, and John Safko. *Classical Mechanics*. Addison Wesley, 3 edition, 2002.

23 Christian Gross and Immanuel Bloch. Quantum simulations with ultracold atoms in optical lattices. *Science*, 357(6355):995–1001, 2017. doi:10.1126/science.aal3837.

24 Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, STOC '96, pages 212–219, New York, NY, USA, July 1996. Association for Computing Machinery. doi:10.1145/237814.237866.

25 Amar Hadzihasanovic, Kang Feng Ng, and Quanlong Wang. Two complete axiomatisations of pure-state qubit quantum computing. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, LICS '18, pages 502–511, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3209108.3209128.

26 Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. doi:10.1103/PhysRevLett.103.150502.

27 Nicolas Heurtel. A complete graphical language for linear optical circuits with finite-photon-number sources and detectors, 2024. arXiv:2402.17693, doi:10.48550/arXiv.2402.17693.

28 Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. A complete axiomatisation of the ZX-calculus for Clifford+T quantum mechanics. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, LICS '18, pages 559–568, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3209108.3209131.

29 Aleks Kissinger and John van de Wetering. Reducing T-count with the ZX-calculus. *Physical Review A*, 102:022406, August 2020. doi:10.1103/PhysRevA.102.022406.

30 Aleks Kissinger and John van de Wetering. Simulating quantum circuits with zx-calculus reduced stabiliser decompositions. *Quantum Science and Technology*, 7(4):044001, July 2022. doi:10.1088/2058-9565/ac5d20.

31 Aleks Kissinger, John van de Wetering, and Renaud Vilmart. Classical simulation of quantum circuits with partial and graphical stabiliser decompositions. In François Le Gall and Tomoyuki Morimae, editors, *Proceedings of the 17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC)*, volume 232 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:13, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.TQC.2022.5.

32 Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11(1):369–395, March 2020. doi:10.1146/annurev-conmatphys-031119-050605.

33 E. Knill. Quantum gates using linear optics and postselection. *Physical Review A*, 66(5):052306, 2002. doi:10.1103/PhysRevA.66.052306.

34 E. Knill, R. Laflamme, and G. J. Milburn. A scheme for efficient quantum computation with linear optics. *Nature*, 409(6816):46–52, 2001. doi:10.1038/35051009.

35 Mark Koch, Richie Yeung, and Quanlong Wang. Speedy contraction of zx diagrams with triangles via stabiliser decompositions, 2023. arXiv:2307.01803.

36 Pieter Kok, W. J. Munro, Kae Nemoto, T. C. Ralph, Jonathan P. Dowling, and G. J. Milburn. Review article: Linear optical quantum computing. *Reviews of Modern Physics*, 79(1):135–174, 2007. doi:10.1103/RevModPhys.79.135.

37 J. C. Loredo, C. Antón, B. Reznychenko, P. Hilaire, A. Harouri, C. Millet, H. Ollivier, N. Somaschi, L. De Santis, A. Lemaître, I. Sagnes, L. Lanco, A. Auffèves, O. Krebs, and P. Senellart. Generation of non-classical light in a photon-number superposition. *Nature Photonics*, 13(11):803–808, 2019. doi:10.1038/s41566-019-0506-3.

38 Saunders Mac Lane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71:40–106, 1965. doi:10.1090/S0002-9904-1965-11234-4.

39 Paul McCloud. The category of linear optical quantum computing, 2022. arXiv:2203.05958.

40 Michael A. Nielsen. Optical quantum computation using cluster states. *Phys. Rev. Lett.*, 93:040503, July 2004. doi:10.1103/PhysRevLett.93.040503.

41 Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, Cambridge, New York, 10th anniversary edition edition, 2010.

42 Jeremy L. O'Brien, Akira Furusawa, and Jelena Vučković. Photonic quantum technologies. *Nature Photonics*, 3(12):687–695, December 2009. doi:10.1038/nphoton.2009.229.

43 Roger Penrose. Angular momentum: an approach to combinatorial space-time. In T. Bastin, editor, *Quantum Theory and Beyond*, pages 151–180. Cambridge University Press, Cambridge, 1971.

44 Boldizsár Poór, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung, and Bob Coecke. Completeness for arbitrary finite dimensions of zxw-calculus, a unifying calculus. In *38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, June 2023. doi:10.1109/lics56636.2023.10175672.

45 John Preskill. Quantum computing 40 years later, 2023. arXiv:2106.10522.

46 Michael Reck, Anton Zeilinger, Herbert J. Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical Review Letters*, 73:58–61, 1994. doi:10.1103/PhysRevLett.73.58.

47 Terry Rudolph. Why I am optimistic about the silicon-photonic route to quantum computing. *APL Photonics*, 2, 2017. doi:10.1063/1.4976737.

48 Stefan Scheel. Permanents in linear optical networks, 2004.

49 Peter Selinger. Dagger compact closed categories and completely positive maps: (extended abstract). *Electronic Notes in Theoretical Computer Science*, 170:139–163, 2007. doi:10.1016/j.entcs.2006.12.018.

50 Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 124–134, November 1994. doi:10.1109/SFCS.1994.365700.

51 John van de Wetering, Richie Yeung, Tuomas Laakkonen, and Aleks Kissinger. Optimal compilation of parametrised quantum circuits, January 2024. arXiv:2401.12877.

52 Quanlong Wang, Boldizsár Poór, and Razin A. Shaikh. Completeness of qufinite zxw calculus, a graphical language for finite-dimensional quantum theory, 2024. arXiv:2309.13014.

53 N. Yoran and B. Reznik. Deterministic linear optics quantum computation with single photon qubits. *Phys. Rev. Lett.*, 91:037903, July 2003. doi:10.1103/PhysRevLett.91.037903.

A Notations

Table 1 Notations used in the paper.

Symbol	Meaning
C, C'	**LO$_{fi}$**-circuits.
$D, D', T, \tilde{\triangle}, \tilde{\Diamond}$	**LOPP**-circuits, cf Table 2 for the specific classes of circuits.
φ, θ	Parameters (angles) of phase shifters and beam splitters
$n, m, \tilde{n}, \tilde{m}$	Integers used for the number of inputs (n or $n+\tilde{n}$) and outputs (m or $m+\tilde{m}$)
i, j, k, ℓ, p, q	Integers used for indexing.
$\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{x}, \boldsymbol{y}$	Fock basis vectors.
$\mathcal{S}, \mathcal{T}, \mathcal{U}, \mathcal{V}$	Finite set of indexes associated with their lowercase vector. Often omitted in the sums.
$\mathcal{B}_m$	Hilbert space of the bosonic Fock space over m modes, cf Definition 17.
$\mathcal{B}_m^{\text{pre}}$	Pre-Hilbert space of the bosonic Fock space over m modes, cf Definition 19.
$\boldsymbol{f}, \boldsymbol{f}'$	Vectors of $\mathcal{B}^{\text{pre}}$.
$\boldsymbol{g}, \boldsymbol{g}'$	Vectors of $(\mathcal{B}^{\text{pre}})^*$, the dual of $\mathcal{B}^{\text{pre}}$.
$\hat{a}_j^\dagger$	Creation operator over the mode j, introduced in Definition 42
$\Omega^{\cdot,\cdot}, \Delta^{\cdot,\cdot}$	Operators defined in Definition 42.
$\prec_r, \preceq_r$	Reverse lexicographic order on vectors, cf Proposition 45.
$\sum, \prod$	Finite sums and products when the upper bound or the set of indexes is omitted.

B Properties of the triangular forms of Section 2.3

Proof of Proposition 11

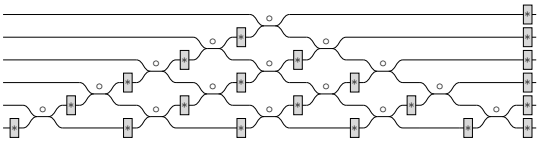
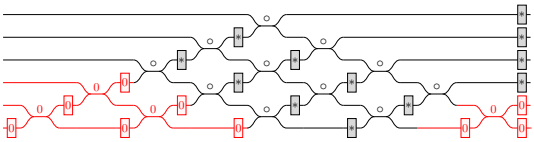
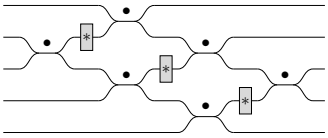
The coefficient $t_{i,j}$ of $[\![\triangle]\!]_1$ is determined by the sum of all the paths from the j^{th} input wire to the i^{th} output wire, where for each path, we multiply by a cos (resp. sin) term when the photon is reflected on (resp. transmitted through) a beam splitter, and by a phase when the path crosses a phase shifter. For instance:

$$\begin{aligned} t_{1,2} &= \cos(\theta_{1,2}) e^{i\varphi_{1,2}} i \sin(\theta_{1,1}) e^{i\varphi_{1,1}} \text{ and} \\ t_{2,2} &= i \sin(\theta_{1,2}) \cos(\theta_{2,2}) e^{\varphi_{2,2}} i \sin(\theta_{2,1}) e^{i\varphi_{2,1}} + \cos(\theta_{1,2}) e^{\varphi_{1,2}} i \sin(\theta_{1,1}) \cos(\theta_{2,1}) e^{\varphi_{1,2}}. \end{aligned}$$

More generally, we have $t_{i,j} = e^{i\varphi_{i,j}} \cos(\theta_{i,j}) \times q_{i,j} + r_{i,j}$ where $q_{i,j}, r_{i,j}$ are terms depending uniquely on the the angles with lower indexes. We can notice there is at most one path from the j^{th} input wire to the i^{th} output wire involving $\theta_{i,j}$ and $\varphi_{i,j}$ and that $q_{i,j} \neq 0$ if and only if all $\theta_{k<i,j}$ and $\theta_{i,\ell<j}$ are nonzero. If one $\theta_{i,\ell<j}$ is zero, then we have $\varphi_{i,j} = \theta_{i,j} = 0$ by the properties of the $\triangle$-circuits. If there are K values of $\theta_{k<i,j}$ which are zero, then all the K diagonals $\theta_{k,\ell'\geq j}$ are zero. By now considersing the path from the $(j+K)^{th}$ input wire to the i^{th} output wire, we recover the same type of equation with $q_{i,j} \neq 0$. Now, we can substract $r_{i,j}$ and dividing by $q_{i,j}$, so that we have $e^{i\varphi_{i,j}} \cos(\theta_{i,j}) = z_{i,j}$ with $z_{i,j} = (t_{i,j} - r_{i,j})/q_{i,j}$. If $z_{i,j} \neq 0$ then $\theta_{i,j} \in [0, \frac{\pi}{2})$ and $\varphi \in [0, 2\pi)$ are uniquely determined. If $z_{i,j} = 0$, then $\theta_{i,j} = \frac{\pi}{2}$ and by the properties of $\triangle$-circuits, we have $\varphi_{i,j} = 0$.

▶ Remark 46. The existence and the uniqueness have been shown in [10] for very similar circuits that have two minor differences; the phases were on the top left of the beam splitters, and the range of the thetas and phases, expect the last layer, were all in $[0, \pi)$. We can therefore have an alternative proof by changing the strongly normalising and confluent rewriting system so the thetas are always in $[0, \frac{\pi}{2}]$ and the phases stay on the bottom left instead of the top left, without restricting their range.

Table 2 Shapes and properties of classes of triangle **LOPP**-circuits: $n+\tilde{n} \to m+\tilde{m}$. $(*, \circ)$ are angles in $[0, 2\pi) \times [0, \frac{\pi}{2}]$ that satisfy the properties of Defitions 9 and 13. We emphasis the nonzero angles of $\tilde{\Diamond}$ by noting $\bullet$ an arbitrary angle in $(0, \frac{\pi}{2}]$. The angles which are necessarily zero for the property 3 and 4 of Definition 13 are in red. We have $n = \tilde{n} = 3, m = 4$ and $\tilde{m} = 2$ for the first two figures, and $\tilde{m} = n = 2$ and $\tilde{n} = m = 3$ for the third.

Shape	Properties
	$\triangle$-circuits (Definition 9) Uniquely determined by $[\![\cdot]\!]_1$ (Proposition 11).
	${}_{\tilde{n}}\triangle_{\tilde{m}}$-circuits (Definition 13) Uniquely determined by the submatrix $[\![\cdot]\!]_1(1:m, 1:n)$ (Proposition 14). Used for the normal forms of $\mathbf{LO}_{fi}$.
	${}_{\tilde{n}}\Diamond_{n}$-circuits (Definition 15) They have exactly $\tilde{n} \times \tilde{m}$ nonzero beam splitters, with no identity wire. They are used in the proofs of Section 4.

C Choice of the generators

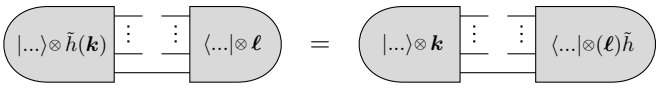
The sources and detectors of the $\mathbf{LO}_{fi}$-calculus allow any and arbitrary finite support state on many modes, which may seem to be too powerful or far from the physical implementation. In that regard, we would like to highlight that:

- Some sources can directly generate more generic states such as a coherent superposition with the vacuum of the 2-photon state [37], or even directly create entangled states [14].
- Linear optical circuits are very modular, and each building block is usually used many times. It would therefore be more convenient to sometimes represent those building blocks directly by specifying what they do, instead of how they are implemented, as illustrated in Figure 2.
- Optical interactions are very combinatorics, thus being unlikely to have a complete equational theory with only single mode sources[7].
- This formalism still allows finding new results for linear optics, like the unique normal forms Section 4.

[7] We can note that [15] bypasses that problem by allowing sums of diagrams

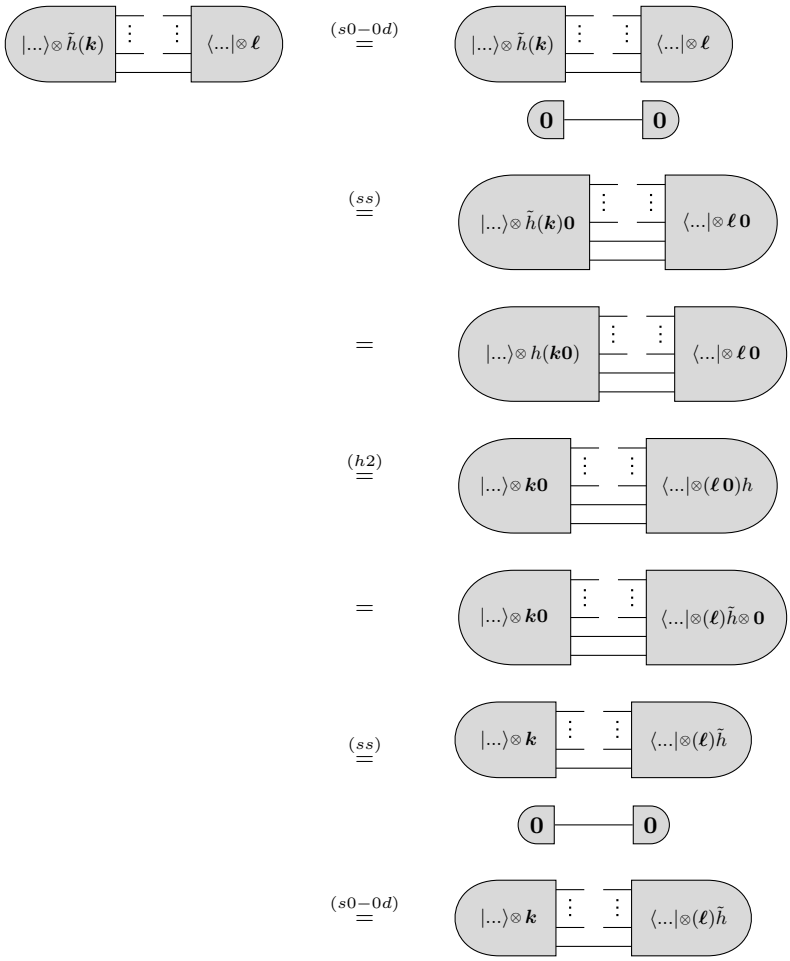
D Derivation in LOPP of the rule (14)

First, we show that we can derive a similar rule of (h2) but only on one wire:

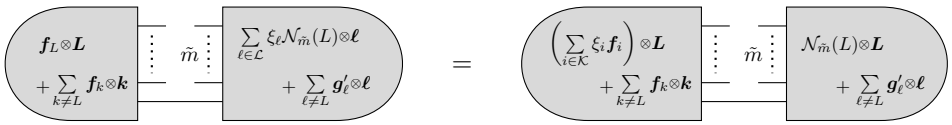


(h1)

To prove it, we consider a linear function $h : \mathcal{B}_2^{\mathrm{pre}} \to \mathcal{B}_2^{\mathrm{pre}}$ such that for every $k \in \mathbb{N}$, $h(|k,0\rangle) = \tilde{h}(|k\rangle)$ and $(\langle k,0|)h = (\langle k|)\tilde{h}$.



▶ **Lemma 47.** *We can derive the equation (14) in the* $\mathbf{LO}_{fi}$*-calculus:*



$g'_\ell \perp \mathcal{N}_{\tilde{m}}(L)$

$(\xi_L \neq 1) \vee (\exists \ell \neq L, \xi_\ell \neq 0)$

Proof. Let $\langle\psi_L| = \sum_{\ell\in\mathcal{L}} \xi_\ell \langle\ell|$. We have $\boldsymbol{g} = \langle\mathcal{N}_{\tilde{m}}(L)| \langle\psi_L| + \sum_{\ell\in\mathcal{L}\setminus\{L\}} \langle g_\ell| \langle\ell|$. Let $\tilde{h} : \mathcal{B}_1^{\mathrm{pre}} \to \mathcal{B}_1^{\mathrm{pre}}$ be a linear function which is the identity on $|i\rangle$ for every $i \in (\mathcal{K}\cup\mathcal{L})\setminus\{L\}$, such that $\langle L|\tilde{h} = \langle\psi_L|$, and zero elsewhere. We can check that $\tilde{h}|k\rangle = |k\rangle + \xi_k|L\rangle$ for $k \neq L$, and $\tilde{h}|L\rangle = \xi_L|L\rangle$. We have:

$$\begin{aligned}\boldsymbol{g} &= \langle\mathcal{N}_{\tilde{m}}(\ell)| \langle\psi_L| + \textstyle\sum_{\ell\in\mathcal{L}\setminus\{L\}} \langle g_\ell| \langle\ell| \\ &= \langle\mathcal{N}_{\tilde{m}}(L)| \langle L|\tilde{h} + \textstyle\sum_{\ell\in\mathcal{L}\setminus\{L\}} \langle g_\ell| \langle\ell|\tilde{h}\end{aligned}$$

The linear function $\tilde{h}$ can therefore be removed with the equation (h1), leading to:

$$\begin{aligned}\boldsymbol{f} &= |f_L\rangle \tilde{h}(|L\rangle) + \textstyle\sum_{k\neq L} |f_k\rangle \tilde{h}(|k\rangle) \\ &= \xi_L|f_L\rangle|L\rangle + \textstyle\sum_{k\neq L} |f_k\rangle (|k\rangle + \xi_k|L\rangle) \\ &= \left(\textstyle\sum_{i\in\mathcal{K}} \xi_i|f_i\rangle\right)|L\rangle + \textstyle\sum_{k\neq L}|f_k\rangle|k\rangle\end{aligned}$$

◀

A Strictly Linear Subatomic Proof System

Victoria Barrett[1]
Department of Computer Science, University of Bath, UK
Inria Saclay, Palaiseau, France

Alessio Guglielmi
Department of Computer Science, University of Bath, UK

Benjamin Ralph
Department of Computer Science, University of Bath, UK

Abstract

We present a subatomic deep-inference proof system for a conservative extension of propositional classical logic with decision trees that is strictly linear. In a strictly linear subatomic system, a single linear rule shape subsumes not only the structural rules, such as contraction and weakening, but also the unit equality rules. An interpretation map from subatomic logic to propositional classical logic recovers the usual semantics and proof theoretic properties. By using explicit substitutions that indicate the substitution of one derivation into another, we are able to show that the unit-equality inference steps can be eliminated from a subatomic system for propositional classical logic with only a polynomial complexity cost in the size of the derivation, from which it follows that the system p-simulates Frege systems, and we show cut elimination for the resulting strictly linear system.

2012 ACM Subject Classification Theory of computation → Proof theory

Keywords and phrases Deep inference, Open deduction, Subatomic logic, Decision trees, Explicit substitutions, Cut elimination, Proof theory

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.39

Funding *Victoria Barrett*: This work was supported by the Engineering and Physical Sciences Research Council [EP/R513155/1] and Inria Exploratory Action IMPROOF.

1 Introduction

A change of formalism can provide us with a new lens on the proof theory of familiar logics, allowing for certain proof-theoretic properties that might be unachievable in more established systems [7, 20]. One of the main motivations behind the deep-inference [13] methodology is the pursuit of **locality**, allowing us to check the correctness of inference steps in constant time. Therefore, many deep-inference proof systems consist of inference rules that are either **atomic** or **linear** [5, 6, 8, 9]. Examples of atomic and linear rules for propositional classical logic are, respectively, the **atomic contraction** and the **medial rule**, shown here:

$$\mathsf{c}\,\frac{a \vee a}{a} \qquad\qquad \mathsf{m}\,\frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

One way that locality has been used to benefit the proof theory of classical logic is via the normalisation mechanisms employing **atomic flows** [3, 14, 15], which use the fact that all rules are either atomic or linear to trace the flow of atoms in a derivation. A proof system with atomic structural rules also allows for finer control of compression mechanisms such as contraction, giving fully lazy sharing when translated into the lambda calculus through a Curry-Howard interpretation [16]. However, the notion of linearity used in such proof systems only applies to atoms. In this work, we define and pursue a more extreme form of

[1] Corresponding Author

linearity, **strict linearity**, in which we require not only linearity with respect to the units instead of the atoms in a derivation. By introducing another proof compression mechanism that has been studied in deep-inference settings [12, 19], **explicit substitutions**, into our proof system, we can define a sound and complete proof system with strictly linear rules, while keeping a handle on proof complexity.

To achieve strict linearity, we further develop the **subatomic logic** approach [1, 2], where atoms are treated as non-commutative self-dual connectives whose arguments are their truth values. Subatomic logics necessitate a deep-inference proof system, since cut elimination for logics with non-commutative self-dual connectives are not possible in Gentzen systems [20]. While the syntax of subatomic logic may seem obscure, semantically they can be understood as the integration of binary decision trees into the the standard language of propositional classical logic [4].

When translated into subatomic logic, both linear and non-linear inference rules can be encoded by a common linear shape, called the **subatomic shape**:

$$\frac{(A\,\alpha\,B)\,\beta\,(C\,\alpha\,D)}{(A\,\beta\,C)\,\alpha\,(B\,\beta\,D)}\quad,$$

Deep-inference proof systems using the subatomic shape are able to capture a range of logics, including those that cannot be expressed in a Gentzen formalism such as BV, and characterise their normalisation in a common way across these different logics [1].

Indeed, although translating a deep-inference proof system into subatomic logic results in a larger space of proofs, normalisation is simplified because there is only a single rule shape, and so the number of possible interactions is limited. Therefore, to ensure the preservation of standard proof-theoretic results such as cut elimination we can take a standard, non-subatomic derivation, translate it to subatomic logic to perform the standardised proof theoretic procedure in a system with only a single rule shape, before projecting back to the standard level.

This paper takes the principles underlying subatomic proof theory even further. In previous work in subatomic logic, the structural rules are subsumed by the subatomic rule shape but inference rules obtained from unit equalities are left intact. However, because the interpretation map from subatomic logic to a non-subatomic logic can collapse these unit equalities itself, there is a redundancy here and a natural question arises: can the rules obtained from unit equalities be eliminated from subatomic proof systems? In this paper we show that the answer is positive.

We call a proof system **strictly linear** if it contains only rules of the subatomic shape. Inference rules based on unit equalities such as $\dfrac{A \wedge \mathsf{t}}{A}$ are not strictly linear, because they are not linear in the units. We have two main motivations for studying such systems.

The first concerns normalisation. With only a single rule shape, normalisation procedures can be simplified yet further, and interference caused by unit-equality inference steps can be eliminated.

The second concerns semantics and complexity. Factorising proofs using explicit substitutions allows for the compression of proofs and the elimination of an unnecessary form of "bureaucracy" [19]. More speculatively, this work provides a theoretical foundation for the development of a proof system with explicit substitutions that retains locality, and without having to exclude derivations containing cycles between cuts and identities [3, 18]. Having defined explicit substitutions for strictly linear proofs and begun to explore how they impact complexity and normalisation, we can in the future lift this to the standard, non-subatomic level using the interpretation from subatomic to standard logic.

In summary, the central contribution of this work is to show that a strictly linear system for the subatomic version of propositional classical logic can be obtained. We do so by showing that a subatomic proof with unit-equality rules can be transformed into one with no unit-equality rules but with explicit substitutions, with only a polynomially-bounded increase in the size of the proof. In doing so, this work furthers the tradition of using the compositional freedom offered by deep-inference formalisms to regularise and homogenise proof systems and normalisation procedures, entirely eliminating all structural variety and non-locality from inference rules.

Outline

In Section 2 we define the preliminaries necessary for the paper: we introduce explicit substitutions and describe the ways in which derivations containing them can be composed, we introduce the proof systems that we will use in this work, in particular we introduce subatomic and strictly linear systems for classical logic.

In Section 3 we introduce the technical machinery that we will use to prove the results in the later chapter, in particular, the Eversion Lemma.

In Section 4 we show that the unit-equality inference steps can be eliminated from a subatomic system for propositional logic. From this result, we can obtain a strictly linear system that is complete for classical logic. In particular, we show that, by using eversion and explicit substitutions, we can achieve this with only a polynomial complexity cost in the size of the derivation.

In Section 5 we show that the cut rule can be eliminated from the strictly linear system in a way that is preserved by interpretation to the standard, non-subatomic level.

2 Preliminaries

▶ **Definition 1.** *We have the following mutually disjoint countable sets of atoms, connectives, units and variables:*

$$\mathcal{A} = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \dots\}, \quad \mathcal{C} = \{\wedge, \vee\}, \quad \mathcal{U} = \{\mathsf{0}, \mathsf{1}\}, \quad \mathcal{V} = \{x, y, z, w, \dots\}$$

The set of ***formulae****,* $\mathcal{F}$ *is defined in the following way:*

$$\mathcal{F} ::= \mathcal{V} \mid \mathcal{U} \mid \mathcal{F}\,\mathcal{A}\,\mathcal{F} \mid \mathcal{F}\,\mathcal{C}\,\mathcal{F} \mid \langle \mathcal{F} | \mathcal{V} \rangle\, \mathcal{F}$$

Note that, since we are working with subatomic logic, atoms are binary connectives rather than atomic formulae. We also can compose formulae by ***explicit substitution****, where* $\langle A|x\rangle\, B$ *denotes the explicit substitution of* A *for a variable* x *in* B*.*

Formulae containing no explicit substitutions are called ***flat formulae****. Given a formula* A*, we write* $\mathsf{fl}\, A$*, the* ***flat expansion*** *of* A*, for the formula where all the explicit substitution terms* $\langle C|x\rangle\, D$ *appearing in* A *are applied, i.e. each instance of the variable* x *in* D *is replaced by* C*, and* $\mathsf{fl}\, A$ *is the (unique) formula so obtained. We denote by* $A \equiv B$ *the syntactic identity of* A *and* B *modulo renaming of variables bound by explicit substitution.*

▶ **Definition 2.** *Let the two operations* ***down-saturation*** *and* ***up-saturation*** *on atoms and connectives be defined as* $\check{\wedge} = \check{\vee} = \vee$*,* $\hat{\wedge} = \hat{\vee} = \wedge$ *and* $\hat{\boldsymbol{a}} = \check{\boldsymbol{a}} = \boldsymbol{a}$ *for each* $\boldsymbol{a} \in \mathcal{A}$*.*

This definition can be extended to formulae by replacing each atom and connective by its up- or down-saturation respectively.

▶ **Definition 3.** *The set of* ***derivations*** $\mathcal{D}$*, denoted by* ϕ*,* ψ*,* χ*,* ω*, . . . , is defined by the grammar:*

$$
\begin{array}{lll}
\mathcal{D} ::= \mathcal{V} \mid \mathcal{U} & & \\
\quad \mid \mathcal{D}\,\mathcal{C}\,\mathcal{D} \mid \mathcal{D}\,\mathcal{A}\,\mathcal{D} & & \textit{composition by } \textbf{\textit{connective}} \textit{ or } \textbf{\textit{atom}}, \\
\quad \left| \boxed{\begin{array}{c}\mathcal{D}\\ \sim\\ \mathcal{D}\end{array}} \right. & & \textit{composition by } \textbf{\textit{expansion}}, \\
\quad \left| \boxed{r\,\dfrac{\mathcal{D}}{\mathcal{D}}} \right. & & \textit{composition by } \textbf{\textit{inference}} \textit{ or } \textbf{\textit{inference step}}, \\
\quad \left| \langle \mathcal{D} | \mathcal{V} \rangle\, \mathcal{D} \right. & & \textit{composition by } \textbf{\textit{explicit substitution}},
\end{array}
$$

We say that $\langle \phi | x \rangle\, \psi$ *is the* ***explicit substitution*** *of* ϕ *into a variable* x *of* ψ*. We define free and bound variables in the usual way: in particular every occurrence of* x *in* ψ *is* ***bound*** *in* $\langle \phi | x \rangle\, \psi$ *and if a variable occurrence is not bound it is* ***free****. We do not consider the substitution variable* x *in* $\langle \phi | x \rangle$ *to be an occurrence of the variable* x *so it is neither free nor bound. We denote by* $\underline{\phi}$ *the set of free variables appearing in the derivation* ϕ*.*

We say that a derivation is ***open*** *if it contains no units (so that its every leaf is a free variable that can be substituted into) and that it is* ***flat*** *if it contains no explicit substitutions. Explicit substitution terms such as* $\langle \phi | x \rangle$ *can be denoted by* π*,* ρ*,* σ*,* τ*, and so on. We may drop parentheses and boxes when there is no ambiguity. We denote by* $\phi \equiv \psi$ *the syntactic identity of* ϕ *and* ψ *modulo renaming of bound variables and associativity of compositions by expansion and inference.*

▶ **Definition 4.** *The* ***size*** $|\phi|$ *of a derivation* ϕ *is the number of occurrences of variables and units appearing in it, not counting the substitution variables in explicit substitution terms, i.e.* $|\langle \phi | x \rangle\, \psi| = |\phi| + |\psi|$*.*

▶ **Definition 5.** *The two maps* ***premise*** *and* ***conclusion****,* $\mathsf{pr}, \mathsf{cn} \colon \mathcal{D} \to \mathcal{F}$ *and the two maps* ***width*** *and* ***height****,* $\mathsf{w}, \mathsf{h} \colon \mathcal{D} \to \mathbb{N}$ *are so defined:*

- *If* $\phi \in \mathcal{F}$*, then* $\mathsf{pr}\,\phi \equiv \mathsf{cn}\,\phi \equiv \phi$ *and* $\mathsf{w}\,\phi = |\phi|$ *and* $\mathsf{h}\,\psi = 0$
- *If* $\phi \equiv \psi\ \alpha\ \chi$*, then*

$$
\begin{array}{ll}
\mathsf{pr}\,\phi \equiv \mathsf{pr}\,\psi\ \alpha\ \mathsf{pr}\,\chi & \mathsf{w}\,\phi = \mathsf{w}\,\psi + \mathsf{w}\,\chi \\
\mathsf{cn}\,\phi \equiv \mathsf{cn}\,\psi\ \alpha\ \mathsf{cn}\,\chi & \mathsf{h}\,\phi = \max(\mathsf{h}\,\psi, \mathsf{h}\,\chi)
\end{array}
$$

- *If* $\phi \equiv \langle \psi | x \rangle\, \chi$*, then*

$$
\begin{array}{ll}
\mathsf{pr}\,\phi \equiv \langle \mathsf{pr}\,\psi | x \rangle\, \mathsf{pr}\,\chi & \mathsf{w}\,\phi = \mathsf{w}\,\psi + \mathsf{w}\,\chi \\
\mathsf{cn}\,\phi \equiv \langle \mathsf{cn}\,\psi | x \rangle\, \mathsf{cn}\,\chi & \mathsf{h}\,\phi = \max(\mathsf{h}\,\psi, \mathsf{h}\,\chi)
\end{array}
$$

- *if* $\phi \equiv \boxed{\dfrac{\psi}{\chi}}$ *or* $\phi \equiv \boxed{\begin{array}{c}\psi\\ \sim\\ \chi\end{array}}$*, then*

$$
\begin{array}{ll}
\mathsf{pr}\,\phi \equiv \mathsf{pr}\,\psi & \mathsf{w}\,\phi = \max(\mathsf{w}\,\psi, \mathsf{w}\,\chi) \\
\mathsf{cn}\,\phi \equiv \mathsf{cn}\,\chi & \mathsf{h}\,\phi = \mathsf{h}\,\psi + \mathsf{h}\,\chi + 1
\end{array}
$$

In Definition 3, we give the definition of a derivation abstracted from any particular proof system, with no correctness criteria given for composition by expansion or inference. However, to be able to identify correct derivations for a specific deep-inference proof system, we need to define what it means to be a correct instance of an inference rule. Furthermore, in order to define proof systems equipped with explicit substitutions, we need to show that the correctness of an instance of composition by expansion $\frac{A}{\sim\!\sim B}$ can be decided in polynomial time on the size of A and B. Proposition 7 adapts Paterson and Wegman's algorithm for linear unification [17] to this problem, in the style of [10].

▶ **Definition 6.** *An **inference rule** is a relation on formulae decidable in polynomial time on the size of its arguments. A **proof system** is a finite set of unit equality and subatomic-linear rules. Given a proof system $\mathcal{S}$, an inference step such that $(\mathsf{cn}\,\psi, \mathsf{pr}\,\chi) \in r \in \mathcal{S}$, for some rule r, is called an **instance of r** and is denoted as $r\,\frac{\psi}{\chi}$.*

*In this paper, we only discuss proof systems with two types of inference rules, **unit-equality rules** and **subatomic rules**. We can therefore specify the unit-equality and subatomic rules of a proof system $\mathcal{S}$ by $\mathcal{S}_{=}$ and $\mathcal{S}_{\mathsf{sa}}$ respectively.*

▶ **Proposition 7.** *Given formulae A and B, the identity $\mathsf{fl}\,A \equiv \mathsf{fl}\,B$ can be decided in linear time with respect to the size of the formulae, by comparing the flat expansions of A and B without actually performing the substitutions.*

Proof. Let us call a **normal representation** of A a formula

$$C \equiv \langle C_n|x_n\rangle \cdots \langle C_1|x_1\rangle\, C_0$$

such that, for $n \geq 0$, $x_1, \ldots, x_n$ are fresh, distinct variables, $C_0, \ldots, C_n$ are flat, $C_0 \notin \{x_1, \ldots, x_n\}$, each of $C_1, \ldots, C_n$ contains one and only one connective and $\mathsf{fl}\,C \equiv \mathsf{fl}\,A$; C can be obtained from A in linear time on $|A|$. Let

$$D \equiv \langle D_m|y_m\rangle \cdots \langle D_1|y_1\rangle\, D_0$$

be a normal representation of B; we can check $\mathsf{fl}\,A \equiv \mathsf{fl}\,B$ by the following recursive procedure invoked as $p(C_0, D_0)$:

Procedure $p(C_i, D_j)$.

1. if $C_i \equiv D_j$, return success;
2. otherwise, if $C_i \equiv x_{i_h} \in \{x_1, \ldots, x_n\}$ and $D_j \equiv y_{j_h} \in \{y_1, \ldots, y_m\}$ and $p(C_{i_h}, D_{j_h})$ succeeds, then return success;
3. otherwise, if $C_i \equiv C_{i_1}\,\alpha\,C_{i_2}$ and $D_j \equiv D_{j_1}\,\alpha\,D_{j_2}$ and, for $h = 1, 2$, $p(C_{i_h}, D_{j_h})$ succeeds, then return success;
4. otherwise, return failure.

To convert the formula A into its normal representation is linear: we transform every subformula $A_1\,\alpha\,A_2$ to $\langle A_1|z\rangle\,\langle A_2|z'\rangle(z\,\alpha\,z')$; the number of such transformations is bounded by the number of connectives in A. To satisfy the constraint that each $C_i, i > 1$ contains exactly one connective, explicit substitutions of the form $\langle A|x\rangle$ for $A \in \mathcal{U} \cup \mathcal{V}$ are applied without increasing the size of the formula.

Where we are comparing formulae $\langle C_i|x_i\rangle\, K\{x_i\}$ and $\langle D_j|y_j\rangle\, K\{y_j\}$, the procedure $p(C_i, D_j)$ need only be performed once, so the comparison is linear on the size of the original formulae. ◀

▶ **Definition 8.** *Given a proof system $\mathcal{S}$, we say that a derivation ϕ in $\mathcal{D}$ is a* ***derivation in $\mathcal{S}$*** *if every inference step in ϕ is an instance of some rule of $\mathcal{S}$ and for each composition by expansion $\overset{\psi}{\underset{\chi}{\sim}}$ we have $\mathsf{fl}\,\mathsf{cn}\,\psi \equiv \mathsf{fl}\,\mathsf{pr}\,\chi$. One way to denote such a derivation is $\phi \| \mathcal{S}$, where $\overset{A}{B}$ A and B are the premise and conclusion of ϕ. We note that by Proposition 7 establishing the correctness of composition by expansion is decidable in linear time.*

▶ **Definition 9.** *We define a set of subatomic rules generated by the following scheme*

$$\hat{\alpha}\beta\,\frac{(x\,\hat{\alpha}\,y)\,\beta\,(z\,\alpha\,w)}{(x\,\beta\,z)\,\alpha\,(y\,\beta\,w)} \qquad \beta\hat{\alpha}\,\frac{(x\,\alpha\,y)\,\beta\,(z\,\hat{\alpha}\,w)}{(x\,\beta\,z)\,\alpha\,(y\,\beta\,w)}$$

$$\breve{\beta}\alpha\,\frac{(x\,\alpha\,y)\,\beta\,(z\,\alpha\,w)}{(x\,\breve{\beta}\,z)\,\alpha\,(y\,\beta\,w)} \qquad \alpha\breve{\beta}\,\frac{(x\,\alpha\,y)\,\beta\,(z\,\alpha\,w)}{(x\,\beta\,z)\,\alpha\,(y\,\breve{\beta}\,w)}$$

where $\alpha, \beta \in \mathcal{C} \cup \mathcal{A}$. We define the set of subatomic rules KDT *to be every rule generated by this scheme together with the mix rule* $\breve{\wedge}\,\dfrac{A \wedge B}{A \vee B}$.

We define a set of unit-equality rules Keq *as follows:*

$$=_1 \frac{x}{x \vee 0} \qquad =_2 \frac{x \vee 0}{x} \qquad =_3 \frac{x}{0 \vee x} \qquad =_4 \frac{0 \vee x}{x}$$

$$=_5 \frac{0}{0\,\alpha\,0} \qquad =_6 \frac{0\,\alpha\,0}{0} \qquad \textit{where} \qquad \alpha \in \{\wedge, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \dots\}$$

$$=_7 \frac{x}{x \wedge 1} \qquad =_8 \frac{x \wedge 1}{x} \qquad =_9 \frac{x}{1 \wedge x} \qquad =_{10} \frac{1 \wedge x}{x}$$

$$=_{11} \frac{1}{1\,\beta\,1} \qquad =_{12} \frac{1\,\beta\,1}{1} \qquad \textit{where} \qquad \beta \in \{\vee, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \dots\}$$

We now define two proof systems by specifying their unit-equality and subatomic rules:

- KDTS*, where* $\mathsf{KDTS}_{=} = \emptyset$ *and* $\mathsf{KDTS}_{\mathsf{sa}} = \mathsf{KDT}$.
- KDTeq*, where* $\mathsf{KDTeq}_{=} = \mathsf{Keq}$ *and* $\mathsf{KDTeq}_{\mathsf{sa}} = \mathsf{KDT}$.

We say that a derivation in KDTS *or* KDTeq *is a* ***proof*** *if its premise is equal to* 1 *with respect to the unit equalities given in* Keq.

▶ Remark 10. Is shown in [4] that the system KDTeq employed in a formalism without explicit substitutions is sound and complete for standard non-subatomic propositional classical logic conservatively extended by decision trees.

Therefore, in this paper, we will only work with KDTeq derivations that are flat, i.e. without any explicit substitutions and we will refer to the logic this system corresponds to as subatomic propositional classical logic.

▶ **Definition 11.** *Let ψ and χ be two derivations such that $\mathsf{cn}\,\psi \equiv \mathsf{pr}\,\chi$. We define a derivation called the* ***synchronal composition*** *of ψ and χ, denoted as*

$$\left[\begin{array}{c}\psi\\ \cdots\\ \chi\end{array}\right];$$

we do so by structural induction, as follows:

1. *if* ψ *is a formula, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \chi$*; similarly, if* χ *is a formula, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \psi$*;*
2. *if* $\psi \equiv \alpha(\psi_1, \ldots, \psi_n)$ *and* $\chi \equiv \alpha(\chi_1, \ldots, \chi_n)$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \alpha\left(\left[\begin{smallmatrix}\psi_1\\ \cdots\\ \chi_1\end{smallmatrix}\right], \ldots, \left[\begin{smallmatrix}\psi_n\\ \cdots\\ \chi_n\end{smallmatrix}\right]\right)$*;*
3. *if* $\psi \equiv \langle \psi_1 | x \rangle\, \psi_2$ *and* $\chi \equiv \langle \chi_1 | x \rangle\, \chi_2$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \left\langle \left[\begin{smallmatrix}\psi_1\\ \cdots\\ \chi_1\end{smallmatrix}\right] \middle| x \right\rangle \left[\begin{smallmatrix}\psi_2\\ \cdots\\ \chi_2\end{smallmatrix}\right]$*;*
4. *if* $\psi \equiv \left[\begin{smallmatrix}\psi_1\\ \sim\\ \psi_2\end{smallmatrix}\right]$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \left[\begin{smallmatrix}\psi_1\\ \sim\\ \psi_2\\ \cdots\\ \chi\end{smallmatrix}\right]$*; similarly, if* $\chi \equiv \left[\begin{smallmatrix}\chi_1\\ \sim\\ \chi_2\end{smallmatrix}\right]$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \left[\begin{smallmatrix}\psi\\ \cdots\\ \chi_1\\ \sim\\ \chi_2\end{smallmatrix}\right]$*;*
5. *if* $\psi \equiv \left[\begin{smallmatrix}\psi_1\\ \hline\\ \psi_2\end{smallmatrix}\right]$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \left[\begin{smallmatrix}\psi_1\\ \hline\\ \psi_2\\ \cdots\\ \chi\end{smallmatrix}\right]$*; similarly, if* $\chi \equiv \left[\begin{smallmatrix}\chi_1\\ \hline\\ \chi_2\end{smallmatrix}\right]$*, then* $\left[\begin{smallmatrix}\psi\\ \cdots\\ \chi\end{smallmatrix}\right] \equiv \left[\begin{smallmatrix}\psi\\ \cdots\\ \chi_1\\ \hline\\ \chi_2\end{smallmatrix}\right]$*.*

▶ **Definition 12.** *A* ***section*** *of a derivation* ϕ *is any formula* A *such that*

$$\phi \equiv \left[\begin{smallmatrix}\psi\\ \cdots\\ A\\ \cdots\\ \chi\end{smallmatrix}\right] \quad ,$$

for some derivations ψ *and* χ*; in the above derivation, each section of* ψ *is said to be* ***above*** *each section of* χ *and each section of* χ *is said to be* ***below*** *each section of* ψ*.*

▶ **Definition 13.** ***Formula contexts*** *are used to indicate formulae with one or more holes, and are denoted* $A\{\,\}\cdots\{\,\}$*, or with other letters as necessary, often* H *and* K*. The holes can be filled by derivations as well as formulae. When unambiguous, we write* $A\{B\}$ *to indicate the formula* A *where the location of its subformula* B *has been singled out.*

▶ **Definition 14.** *We denote an* ***actual substitution*** *that maps* x *to* A *and leaves all other variables unchanged by* $[A|x]$*. Actual substitutions can be applied to derivations and* $[A|x]\,\phi$ *stands for the derivation obtained by replacing every free occurrence of* x *in* ϕ *by the formula* A *and we say that this substitutes* A *for* x *in* ϕ*. In the specific case where we are substituting into a formula, we can extend this notion to allow for the actual substitution of a derivation into a formula, where* $[\psi|x]\,B$ *is obtained by replacing every free occurrence of* x *in the formula* B *by the derivation* ψ*. Note that* $[A|x]\,B\{x\} \equiv B\{A\}$ *if* x *does not appear free in* $B\{\,\}$*.*

We abbreviate the simultaneous actual substitution $[B_1|x_1, \ldots, B_n|x_n]\,A$ *as* $[B_i|x_i]_{1\ldots n}\,A$*. Given a set of variables* $S = \{x_1, \ldots, x_n\}$*, we write* $[B_i|x_i]_S\,A$ *for* $[B_1|x_1, \ldots, B_n|x_n]\,A$*; we might also write* $[B_v|v]_S\,A$ *to stand for* $[B_{x_1}|x_1, \ldots, B_{x_n}|x_n]\,A$*. We denote both individual and simultaneous actual substitutions by* π*,* ρ*,* σ *and* τ*, and so on.*

We extend the conventions on simultaneous actual substitutions to explicit substitutions and derivations. Therefore, we might indicate with $\langle B_v|v\rangle_{\underline{A}}\,\phi$ the substitution $\langle B_{x_1}|x_1, \ldots, B_{x_n}|x_n\rangle\,\phi$, where $\underline{A} = \{x_1, \ldots, x_n\}$ is the set of free variables of A (see Definition 1).

The notation $[B_v|v]_{\underline{A}}A$ is at risk of being ambiguous because the enumeration of the variables is arbitrary. For example, if $\underline{A} = \{v_1, v_2\}$ and $B_1 = v_2$ then $[B_1|v_1]\,[B_2|v_2]\,A \not\equiv [B_2|v_2]\,[B_1|v_1]$. We take care to use this notation only when it is unambiguous and there is no dependency between substitutions.

3 The Merge and Eversion Lemmas

We are now ready to first prove the Merge Lemma, and then its generalisation the Eversion Lemma, which enables the proof of the main results of this paper. As we explained in the introduction, we want to be able to eliminate all the non-linear equality rules from KDTeq to produce a strictly linear proof in KDTS. Using the Eversion Lemma, we are able to transform the derivation on the left into the derivation on the right:

$$\dfrac{\pi\phi}{\pi K\left\{\dfrac{A}{A\vee B}\right\}}\pi\psi \quad\longrightarrow\quad \dfrac{\pi[x\vee B|x]_{\underline{A}}\phi}{\pi K\left\{\begin{array}{l}[x\vee B|x]_{\underline{A}}A\\ \quad\|\\ A\vee\boxed{\begin{array}{l}[B|x]_{\underline{A}}\breve{A}\\ \quad\|\\ \left[\breve{A}^y\middle|y\right]_{\underline{B}}B\end{array}}\end{array}\right\}}\pi\left[\breve{A}^y\middle|y\right]_{\underline{B}}\psi}\,,$$

where we assume that A and B are open formulae and that $\breve{A}^y$ is obtained from A by replacing every variable by y and every connective with its down-saturation. By doing this transformation, the inference step becomes strictly linear. This propagates substitutions up and down the derivation but does not affect its structure.

Before stating and proving the full Eversion Lemma, we state and prove the Merge Lemma, a version of it restricted to a substitution with a single connective or atom.

▶ **Proposition 15** (Merge Lemma). *Let A be an open formula with variables $\{x_1, \ldots, x_n\}$ and let $\beta \in \mathcal{C} \cup \mathcal{A}$. Then there exist* KDTS *derivations:*

$$\begin{array}{c}[y_i\;\beta\;z_i|x_i]_{\underline{A}}A\\ \|\\ [y_i|x_i]_{\underline{A}}A\;\beta\;[z_i|x_i]_{\underline{A}}\breve{A}\end{array}\qquad\qquad\begin{array}{c}[y_i|x_i]_{\underline{A}}A\;\beta\;[z_i|x_i]_{\underline{A}}\widehat{A}\\ \|\\ [y_i\;\beta\;z_i|x_i]_{\underline{A}}A\end{array}\,.$$

The width and height of each derivation are bounded by $2|A|$.

Proof. We consider the derivation on the left and proceed by induction on the structure of A.

- If $A \equiv x_i$ then we take the derivation $y_i\;\beta\;z_i$.
- If $A \equiv C\;\alpha\;D$ then we build:

$$\beta\breve{\alpha}\,\dfrac{\boxed{\begin{array}{c}[y_i\;\beta\;z_i|x_i]_{\underline{C}}C\\ \|\\ [y_i|x_i]_{\underline{C}}C\;\beta\;[z_i|x_i]_{\underline{C}}\breve{C}\end{array}}\;\alpha\;\boxed{\begin{array}{c}[y_i\;\beta\;z_i|x_i]_{\underline{D}}D\\ \|\\ [y_i|x_i]_{\underline{D}}D\;\beta\;[z_i|x_i]_{\underline{D}}\breve{D}\end{array}}}{\left([y_i|x_i]_{\underline{C}}C\;\alpha\;[y_i|x_i]_{\underline{D}}D\right)\;\beta\;\left([z_i|x_i]_{\underline{C}}\breve{C}\;\breve{\alpha}\;[z_i|x_i]_{\underline{D}}\breve{D}\right)}\,.$$

- If $A \equiv \langle C|w\rangle\, D$ then we build:

$$\dfrac{\left\langle \begin{array}{c}[y_i\ \beta\ z_i|x_i]_{\underline{C}}C \\ \| \\ [y_i|x_i]_{\underline{C}}C\ \beta\ [z_i|x_i]_{\underline{C}}\breve{C}\end{array}\middle|\, w\right\rangle [y_i\ \beta\ z_i|x_i]_{\underline{D}}D}{\left\langle [y_i|x_i]_{\underline{C}}C\middle|w'\right\rangle \left\langle [z_i|x_i]_{\underline{C}}\breve{C}\middle|w''\right\rangle \boxed{\begin{array}{c}[y_i\ \beta\ z_i|x_i]_{\underline{D}}\,[w'\ \beta\ w''|w]\,D \\ \| \\ [y_i|x_i]_{\underline{D}}\,[w'|w]\,D\ \beta\ [z_i|x_i]_{\underline{D}}\,[w''|w]\,\breve{D}\end{array}}}{\left\langle [y_i|x_i]_{\underline{C}}C\middle|w\right\rangle [y_i|x_i]_{\underline{D}}D\ \beta\ \left\langle [z_i|x_i]_{\underline{C}}\breve{C}\middle|w\right\rangle [z_i|x_i]_{\underline{D}}\breve{D}}\ .$$

  That is, we perform a merge on C and then, with the first composition by expansion, we replace all occurrences w in D by $w'\ \beta\ w''$, the minimal amount of structure needed to perform the merge on D. The second composition by expansion then rearranges this to the desired structure.

The width of the derivation is at most $2|A|$; note that $\left|[y_i\ \beta\ z_i|x_i]_{\underline{A}}A\right| < 2|A|$ if there are explicit substitutions in A.

For each connective in A there is a corresponding instance of composition by rule in the constructed derivation; and for each explicit substitution in A, there are two corresponding instances of composition by expansion. Therefore the height of the derivation is at most $2|A|$, the worst scenario being a formula only composed of explicit substitutions, each of which is of size 1. ◀

We call the derivations on the left in Proposition 15 **down-merges**, and the derivations on the right **up-merges**.

We can use the Merge Lemma to simulate unit-equality inference steps, without affecting the value or the structure of the rest of the derivation. For example, the unit-equality inference step $=\dfrac{A}{A \wedge 1}$ becomes $\begin{array}{c}[w \wedge 1|w]_{\underline{A}}A \\ \| \\ A \wedge [1|x]_{\underline{A}}\breve{A}\end{array}$. This propagates $w \wedge 1$ upwards through the derivation in place of w, for each variable w occurring in A, and propagates $[1|w]_{\underline{A}}\breve{A}$, which is equal to 1 for any formula A, downwards through the derivation in place of this occurrence of 1.

A naïve approach to eliminating the unit-equality inference steps in this way will blow up the size of the derivation exponentially. To see this, we can consider the following transformation, in which π substitutes a unit onto the variable x and we assume that in each section of ψ, x occurs exactly once, so that the two inference steps shown form a pair:

$$\pi\left[\begin{array}{c}\phi \\ \hdashrule{3em}{0.4pt}{} \\ K\left\{=\dfrac{A}{A\ \alpha\ x}\right\} \\ \hdashrule{3em}{0.4pt}{} \\ \psi \\ \hdashrule{3em}{0.4pt}{} \\ H\left\{=\dfrac{B\ \beta\ x}{B}\right\} \\ \hdashrule{3em}{0.4pt}{} \\ \chi\end{array}\right] \longrightarrow \pi\left[\begin{array}{c}[v\ \beta\ x|v]_{\underline{H\{\ \}}}[w\ \alpha\ x|w]_{\underline{A}}\phi \\ \hdashrule{3em}{0.4pt}{} \\ [v\ \beta\ x|v]_{\underline{H\{\ \}}}K\left\{\begin{array}{c}[w\ \alpha\ x|w]_{\underline{A}}A \\ \| \\ A\ \alpha\ \breve{A}^x\end{array}\right\} \\ \hdashrule{3em}{0.4pt}{} \\ [v\ \beta\ x|v]_{\underline{H\{\ \}}}\left[\breve{A}^x\middle|x\right]\psi \\ \hdashrule{3em}{0.4pt}{} \\ [v\ \beta\ x|v]_{\underline{H\{\ \}}}H\left\{B\ \beta\ \breve{A}^x\right\} \\ \| \\ \boxed{\begin{array}{c}H\{B\} \\ \hdashrule{1em}{0.4pt}{} \\ \chi\end{array}}\ \beta\ [x|v]_{\underline{\breve{H}\{\ \}}}\breve{H}\{\breve{A}^x\}\end{array}\right]$$

Eliminating the two unit-equality inference steps in the way described above would result in a substitution $\left[\widehat{B}^x\middle|x\right]$ being propagated up, and a substitution $\left[\breve{A}^x\middle|x\right]$ being propagated down (where A^x and B^x stand for the result of substituting the variable x onto every leaf of A and B respectively). Therefore, in order to eliminate both unit-equality inference steps, the entire context around one of them must be duplicated, resulting in something like the derivation above. This doubles the width of the derivation, leading to an exponential blow-up in the size when eliminating all unit-equality inference steps in succession.

By iterating the Merge Lemma in a certain way, we derive the Eversion Lemma and can use this to avoid this exponential blow-up.

▶ **Lemma 16** (Eversion Lemma). *Let A and B be open formulae with their free variables denoted $\underline{A} = \{w_1, \ldots, w_n\}$ and $\underline{B} = \{y_1, \ldots, y_m\}$ respectively. Then there exist* KDTS *derivations:*

$$\begin{array}{c}\left[\widehat{B}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \Vert\\ \left[\breve{A}^{y_j}\middle|y_j\right]_{\underline{B}}\widehat{B}\end{array}\qquad\qquad\begin{array}{c}\left[\widehat{B}^{w_i}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \Vert\\ \left[\breve{A}\middle|y_j\right]_{\underline{B}}\widehat{B}\end{array}\quad,$$

where $B^{w_i} \equiv [w_i|y_j]_{\underline{B}}B$ and $A^{y_j} \equiv [y_j|w_i]_{\underline{A}}A$. Both the width and the height of these derivations are $O(|A||B|)$.

Proof. We consider the derivation on the left and proceed by induction on the structure of B.

- If $B \equiv y_j$ then we take the derivation $\breve{A}^{y_j}$.
- If $B \equiv E\,\beta\,F$ then we build:

$$\boxed{\begin{array}{c}\left[\widehat{E}\,\widehat{\beta}\,\widehat{F}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \chi\,\Vert\\ \boxed{\begin{array}{c}\left[\widehat{E}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \phi\,\Vert\\ \left[\breve{A}^{y_j}\middle|y_j\right]_{\underline{E}}\widehat{E}\end{array}}\;\widehat{\beta}\;\boxed{\begin{array}{c}\left[\widehat{F}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \psi\,\Vert\\ \left[\breve{A}^{y_j}\middle|y_j\right]_{\underline{F}}\widehat{F}\end{array}}\end{array}}\quad,$$

  where the derivations ϕ and ψ are obtained by induction and the derivation χ is obtained via the Merge Lemma 15.
- If $B \equiv \langle E|z\rangle\,F$ then we build:

$$\boxed{\begin{array}{c}\left[\left\langle\widehat{E}\middle|z\right\rangle\widehat{F}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \hline \left\langle\widehat{E}\middle|z\right\rangle\boxed{\begin{array}{c}\left[\widehat{F}\middle|w_i\right]_{\underline{A}}\breve{A}\\ \phi\,\Vert\\ \left[\breve{A}^{y_j}\middle|y_j\right]_{\underline{F}\setminus\{z\}}\left[\breve{A}^{z}\middle|z\right]\widehat{F}\end{array}}\\ \hline \left\langle\begin{array}{c}\left[\widehat{E}\middle|z\right]\breve{A}^{z}\\ \psi\,\Vert\\ \left[\breve{A}^{z}\middle|y_j\right]_{\underline{E}}\widehat{E}\end{array}\middle|z\right\rangle\left[\breve{A}^{y_j}\middle|y_j\right]_{\underline{F}\setminus\{z\}}\widehat{F}\end{array}}\quad,$$

  where the derivations ϕ and ψ are obtained by induction.

The width of the derivations generated in each case are all $O(|A||B|)$. The worst-case scenario for the height is when B is composed by connective, in which case the height increases by at most $2|A|$ in the merge χ. The number of iterations is $O(|B|)$, so the height is $O(|A||B|)$. ◀

We call the derivations in Lemma 16 **eversions**.

▶ **Example 17.** Returning to the example of exponential blow-up above, the pair of unit-equality inference steps in the above example can then be eliminated by the following transformation:

$$\pi \boxed{\begin{array}{c} \phi \\ \hline K\left\{ = \dfrac{A}{A\,\alpha\,x} \right\} \\ \hline \psi \\ \hline H\left\{ = \dfrac{B\,\beta\,x}{B} \right\} \\ \hline \chi \end{array}} \quad \longrightarrow \quad \pi \boxed{\begin{array}{c} \left\langle \widehat{B}^x \middle| x \right\rangle [w\;\alpha\;x|w]_{\underline{A}}\phi \\ \hline \left\langle \widehat{B}^x \middle| x \right\rangle K \left\{ \begin{array}{c} [w\;\alpha\;x|w]_{\underline{A}}A \\ \| \\ A\;\beta\;\check{A}^x \end{array} \right\} \\ \hline \left\langle \begin{array}{c} \left[\widehat{B}^x \middle| x \right] \check{A} \\ \| \\ \left[\check{A}^x \middle| x \right] \widehat{B} \end{array} \middle| x \right\rangle \psi \\ \hline \left\langle \check{A}^x \middle| x \right\rangle H \left\{ \begin{array}{c} B\;\beta\;\widehat{B}^x \\ \| \\ [y\;\beta\;x|y]_{\underline{B}}B \end{array} \right\} \\ \hline \left\langle \check{A}^x \middle| x \right\rangle [y\;\beta\;x|y]_{\underline{B}}\chi \end{array}} \quad .$$

This increases the width of the derivation by $O(|A||B|)$ at the widest point, which is the eversion in the explicit substitution, and increases the height by at least $O(|A|) + O(|B|)$ due to the merges, and in the worst case by $O(|A||B|)$, again due to the eversion.

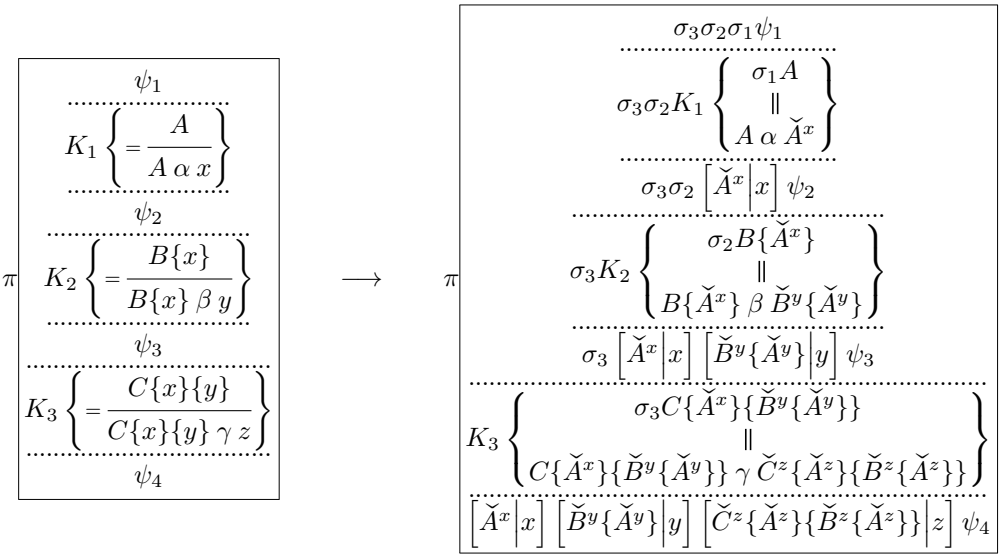
The premise of the transformed derivation is $\pi \left\langle \widehat{B}^x \middle| x \right\rangle [w\;\alpha\;x|w]_{\underline{A}}\,\mathsf{pr}\,\phi$. This is equal to the premise $\pi\,\mathsf{pr}\,\phi$ of the original derivation because $x = \widehat{B}^x$ for any formula B and $\pi(w\,\alpha\,x) = \pi w$ for every variable w appearing in A, since $\pi \boxed{= \dfrac{A}{A\,\alpha\,x}}$ is a unit-equality inference step and therefore either $\alpha \in \mathcal{C}$ and πx is the unit of α or $\alpha \in \mathcal{A}$ and $\pi A \equiv \pi x$.

Similarly, the conclusion becomes $\pi \left\langle \check{A}^x \middle| x \right\rangle [y\;\beta\;x|y]_{\underline{B}}\,\mathsf{cn}\,\chi$, and this is equal to the original conclusion $\pi\,\mathsf{cn}\,\chi$.

4 Strict Linearity

Using the Eversion Lemma, we can design a procedure that avoids most causes of exponential blow-up when eliminating the unit-equality rules. However, when performing this elimination we must pay attention to the size of the substitutions that are propagated up or down the derivation. These accumulated substitutions can also lead to an exponential blow-up in the size of the derivation when the elimination is iterated. To observe this, consider the following example:

▶ **Example 18.**



This shows a case where we have three unit-equality inference steps, and the substitutions accumulate as we eliminate them, because the units introduced get into other unit-equality inference steps. Crucially, we see that in the conclusion, z is replaced by $\breve{C}\{\breve{A}^z\}\{\breve{B}^z\{\breve{A}^z\}\}$, which contains two copies of $\breve{A}^z$: one inherited from x occurring in $B\{x\}$ and one from x occurring in $C\{x\}\{y\}$. This pattern will lead to exponential blow-up for the size of the derivation, but it can be controlled with explicit substitutions. In the elimination procedure we describe below, we will factor out the repeated instances of $\breve{A}^z$ and instead replace z by $\left\langle \breve{A}^z \middle| x \right\rangle \left\langle \breve{B}^z\{x\} \middle| y \right\rangle \breve{C}^z\{x\}\{y\}$, which has size $|A| + |B| + |C|$.

We are now ready to state the main theorem of the paper, that we can convert a flat derivation in KDTeq to a derivation in KDTS, i.e. we convert a derivation with unit-equality rules but no explicit substitutions to one that is strictly linear with explicit substitutions, without exponential blow-up in the size of the proof.

▶ **Definition 19.** *Let* $\phi \begin{Vmatrix} A' \\ B' \end{Vmatrix}$ *be a derivation in* KDTeq. *We call* $\pi \left[\begin{smallmatrix} A \\ \| \\ B \end{smallmatrix}\right]$ *a* ***unit factorisation*** *of* ϕ, *if* $A' = \pi A$ *and* $B' = \pi B$, A *and* B *are open formulae in which no variable occurs more than once, and* π *is a simultaneous actual substitution of all the units that occur in* A' *and* B'.

Note that a derivation can have multiple unit factorisations, but for the purposes of this paper it does not matter which we select.

▶ **Theorem 20.** *Let* $\phi \begin{Vmatrix} A' \\ B' \end{Vmatrix}$ *be a flat derivation in* KDTeq *and* $\pi \left[\begin{smallmatrix} A \\ \| \\ B \end{smallmatrix}\right]$ *a unit factorisation of* ϕ. *We can build a derivation* $\phi' \equiv \pi \left[\begin{smallmatrix} \sigma A \\ \| \\ \tau B \end{smallmatrix}\right]$ *in* KDTS *such that the size of* ϕ' *is polynomial in the size of* ϕ *and* $\pi A = \pi\sigma A$ *and* $\pi B = \pi\tau B$.

Proof (Sketch, full proof is in Appendix A). We eliminate the unit-equality inference steps in two phases. In the first phase, we eliminate all those unit-equality inference steps that propagate a unit downwards through a derivation, those that are labelled by $=_1$, $=_3$, $=_5$, $=_7$, $=_9$ or $=_{11}$; in the second we eliminate the others, which propagate a unit upwards through a derivation.

In the first phase, we replace the inference steps by down-merges, taking care to factor out the accumulating substitutions as described in Example 18, and propagating the resulting substitutions through the derivation.

In the second phase, we replace the remaining inference steps by up-merges, again factoring out the accumulating substitutions, and we reconcile the propagated structures via eversions, as described in Example 17, to obtain the strictly linear derivation ϕ'.

Let w be the width of ϕ and let h be its height. The number of unit-equality inference steps eliminated in each phase is bounded by wh. After the first phase, the maximum width occurs before factoring out the accumulated substitutions and is $O(w^3h^2)$; the height is similarly $O(w^3h^2)$ as this is the most that it increases by for any step. After the second phase, the maximum width again occurs before factoring out the accumulated substitutions and is $O(w^7h^5)$; again this represents the greatest increase of height in any step. Therefore the height and width of ϕ' are each $O(w^7h^5)$. ◀

This result shows that the strictly linear system KDTS is complete for propositional classical logic. In addition, it follows from results shown in [2] and [5] that KDTS p-simulates Frege systems [11].

5 Cut Elimination

We now consider the normalisation of strictly linear proofs by showing a cut elimination procedure. As we mention in the introduction, we are motivated to develop a theory of strict linearity because this combines a theoretical foundation for explicit substitutions with simple normalisation procedures. We would like to be able to take a proof in any standard system (not necessarily subatomic), translate it to a strictly linear system, normalise inside that system, and then project back to the original system without too much difficulty.

We do this by applying the method from [4] of eliminating cuts in subatomic logic via projections, adapting it to be strictly linear.

▶ **Definition 21.** *A **cut on** $\mathbf{a}$ in* KDTeq *is any instance of the rule*

$$\wedge\hat{\mathbf{a}}\,\frac{(A\,\mathbf{a}\,B)\wedge(C\,\mathbf{a}\,D)}{(A\wedge C)\,\mathbf{a}\,(B\wedge D)}$$

such that $A = \mathsf{0} = D$ and $B = \mathsf{1} = C$, or $A = \mathsf{1} = D$ and $B = \mathsf{0} = C$. In the system KDTS*, we take explicit substitutions in the context into account, and so a **cut on** $\mathbf{a}$ is any subderivation $K\left\{\wedge\hat{\mathbf{a}}\,\dfrac{A}{B}\right\}$ inside a derivation such that $\mathsf{fl}\,K\{A\}$ and $\mathsf{fl}\,K\{B\}$ when vertically composed form a cut on $\mathbf{a}$.*

We restrict the procedure that we define here to those proofs that do not exhibit too much nesting of atoms inside themselves; this is sufficient to capture a translation of any proof in the standard deep inference system for propositional classical logic, SKS.

▶ **Definition 22.** *Given a derivation ϕ and an atom $\mathbf{a}$, we say that $\mathbf{a}$ is **unnested** in ϕ if there is no section whose flat expansion is of the form $K\{H\{A\,\mathbf{a}\,B\}\,\mathbf{a}\,C\}$ or $K\{A\,\mathbf{a}\,\{H\{B\,\mathbf{a}\,C\}\}$.*

▶ **Definition 23.** *For a derivation ϕ in* KDTS *and an atom $\boldsymbol{a}$ that is unnested in ϕ, we define the* ***left-projection on $\boldsymbol{a}$*** *of ϕ, written $\mathsf{l}_{\boldsymbol{a}}\,\phi$, as follows:*

- *If $\phi \in \mathcal{V} \cup \mathcal{U}$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \phi$.*
- *If $\phi \equiv \psi\ \boldsymbol{a}\ \chi$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \psi$.*
- *If $\phi \equiv \psi\ \beta\ \chi$ for $\beta \not\equiv \boldsymbol{a}$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \mathsf{l}_{\boldsymbol{a}}\,\psi\ \beta\ \mathsf{l}_{\boldsymbol{a}}\,\chi$.*
- *If $\phi \equiv \langle \psi | x \rangle\ \chi$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \langle \mathsf{l}_{\boldsymbol{a}}\,\psi | x \rangle\ \mathsf{l}_{\boldsymbol{a}}\,\chi$.*
- *If $\phi \equiv \boxed{\dfrac{\psi}{\sim\!\!\sim\ \chi}}$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \boxed{\dfrac{\mathsf{l}_{\boldsymbol{a}}\,\psi}{\sim\!\!\sim\ \mathsf{l}_{\boldsymbol{a}}\,\chi}}$.*
- *If $\phi \equiv \boxed{\breve{\wedge}_{\boldsymbol{a}}\dfrac{\dfrac{\psi}{\vdots\ (A\ \boldsymbol{a}\ B) \wedge (C\ \boldsymbol{a}\ D)}}{\vdots\ (A \vee C)\ \boldsymbol{a}\ (B \wedge D)}}{\chi}$ or $\phi \equiv \boxed{\hat{\vee}_{\boldsymbol{a}}\dfrac{\dfrac{\psi}{\vdots\ (A \wedge C)\ \boldsymbol{a}\ (B \vee D)}}{\vdots\ (A\ \boldsymbol{a}\ B) \vee (C\ \boldsymbol{a}\ D)}}{\chi}$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \boxed{\breve{\wedge}\dfrac{\dfrac{\mathsf{l}_{\boldsymbol{a}}\,\psi}{\vdots\ \mathsf{l}_{\boldsymbol{a}}\,A \wedge \mathsf{l}_{\boldsymbol{a}}\,C}}{\vdots\ \mathsf{l}_{\boldsymbol{a}}\,A \vee \mathsf{l}_{\boldsymbol{a}}\,C}}{\mathsf{l}_{\boldsymbol{a}}\,\chi}}$.*
- *If $\phi \equiv \boxed{\breve{\boldsymbol{a}}\beta\dfrac{\dfrac{\psi}{\vdots\ (A\ \beta\ B)\ \boldsymbol{a}\ (C\ \beta\ D)}}{\vdots\ (A\ \boldsymbol{a}\ C)\ \beta\ (B\ \boldsymbol{a}\ D)}}{\chi}$ or $\phi \equiv \boxed{\hat{\boldsymbol{a}}\beta\dfrac{\dfrac{\psi}{\vdots\ (A\ \boldsymbol{a}\ C)\ \beta\ (B\ \boldsymbol{a}\ D)}}{\vdots\ (A\ \beta\ B)\ \boldsymbol{a}\ (C\ \beta\ D)}}{\chi}$ then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \boxed{\dfrac{\mathsf{l}_{\boldsymbol{a}}\,\psi}{\vdots\ \mathsf{l}_{\boldsymbol{a}}\,\chi}}$, and similarly for $\boldsymbol{a}\breve{\wedge}$ and $\boldsymbol{a}\hat{\vee}$. Note that here β cannot be $\boldsymbol{a}$ due to the assumption that $\boldsymbol{a}$ is unnested in ϕ.*
- *If $\phi \equiv \boxed{r\dfrac{\psi}{\chi}}$ in any other case, then $\mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \boxed{r\dfrac{\mathsf{l}_{\boldsymbol{a}}\,\psi}{\mathsf{l}_{\boldsymbol{a}}\,\chi}}$; note here again that r cannot be $\boldsymbol{a}\breve{\boldsymbol{a}}$ due to the assumption that $\boldsymbol{a}$ is unnested in ϕ.*

The ***right-projection on $\boldsymbol{a}$*** *is denoted by $\mathsf{r}_{\boldsymbol{a}}\,\phi$ and defined in exactly the same way, except for the following case:*

- *If $\phi \equiv \psi\ \boldsymbol{a}\ \chi$ then $\mathsf{r}_{\boldsymbol{a}}\,\phi \equiv \chi$.*

▶ Remark 24. For any atom $\boldsymbol{a}$ that is unnested in a derivation $\phi \in$ KDTS, $\mathsf{l}_{\boldsymbol{a}}\,\phi$ and $\mathsf{r}_{\boldsymbol{a}}\,\phi$ are uniquely determined. Note that it is not the case that $\mathsf{l}_{\boldsymbol{a}}$ and $\mathsf{r}_{\boldsymbol{a}}$ commute: for example $\mathsf{l}_{\boldsymbol{a}}\,\mathsf{r}_{\boldsymbol{a}}(0\ \boldsymbol{a}\ 1) \equiv \mathsf{l}_{\boldsymbol{a}}\,1 \equiv 1$ and $\mathsf{r}_{\boldsymbol{a}}\,\mathsf{l}_{\boldsymbol{a}}(0\ \boldsymbol{a}\ 1) \equiv \mathsf{r}_{\boldsymbol{a}}\,0 \equiv 0$.

▶ Remark 25. For any derivation ϕ in KDTS and any atom $\boldsymbol{a}$ that is unnested in ϕ, the projected derivations $\mathsf{l}_{\boldsymbol{a}}\,\phi$ and $\mathsf{r}_{\boldsymbol{a}}\,\phi$ contain no occurrences of $\boldsymbol{a}$, and so neither contains any cuts on $\boldsymbol{a}$.

It can be the case that eliminating the unit-equality inference steps from a derivation ψ in KDTeq in which $\boldsymbol{a}$ is unnested can create nesting of this atom. This occurs when a derivation contains a pair of unit-equality inference steps $=\dfrac{A\{w\ \boldsymbol{a}\ x\}}{A\{w\ \boldsymbol{a}\ x\}\ \alpha\ u}$ and $=\dfrac{B\{y\ \boldsymbol{a}\ z\}\ \beta\ u}{B\{y\ \boldsymbol{a}\ z\}}$, so that $A\{w\ \boldsymbol{a}\ x\}$ and $B\{y\ \boldsymbol{a}\ z\}$ both contain the atom $\boldsymbol{a}$.

The merge constructions by which we simulate the unit-equality inference steps propagate upwards a substitution $\left\langle \breve{A}^{u}\{u\ \boldsymbol{a}\ u\} \middle| u \right\rangle$ and downwards a substitution $\left\langle \breve{B}^{u}\{u\ \boldsymbol{a}\ u\} \middle| u \right\rangle$. These are resolved by an eversion, which produces an inference step $\boldsymbol{a}\breve{\boldsymbol{a}}\dfrac{(u\ \boldsymbol{a}\ u)\ \boldsymbol{a}\ (u\ \boldsymbol{a}\ u)}{(u\ \boldsymbol{a}\ u)\ \boldsymbol{a}\ (u\ \boldsymbol{a}\ u)}$. That is to say, we create the logical material of $(u\ \boldsymbol{a}\ u)$ twice and substitute one copy into the other. Therefore we define a slightly looser notion of nestedness that captures derivations produced in this way.

▶ **Definition 26.** *Given a derivation ϕ and an atom $\boldsymbol{a}$, we say that $\boldsymbol{a}$ is* ***shallowly nested*** *in ϕ if there is no section whose flat expansion is of the form $K\{H\{L\{A\,\boldsymbol{a}\,B\}\,\boldsymbol{a}\,C\}\,\boldsymbol{a}\,D\}$ or similar, and every instance of the inference rule $\boldsymbol{a}\check{\boldsymbol{a}}$ is of the form $\boldsymbol{a}\check{\boldsymbol{a}}\,\dfrac{(A\,\boldsymbol{a}\,A)\,\boldsymbol{a}\,(A\,\boldsymbol{a}\,A)}{(A\,\boldsymbol{a}\,A)\,\boldsymbol{a}\,(A\,\boldsymbol{a}\,A)}$, for every atom $\boldsymbol{a} \in \mathcal{A}$.*

We can extend the definition of left- and right-projection on $\boldsymbol{a}$ to derivations in which $\boldsymbol{a}$ is shallowly nested as follows:

$$\text{If } \phi \equiv \boxed{\begin{array}{c}\psi\\ \vdots\\ \check{\boldsymbol{a}}\beta\,\dfrac{(A\,\boldsymbol{a}\,A)\,\boldsymbol{a}\,(A\,\boldsymbol{a}\,A)}{(A\,\boldsymbol{a}\,A)\,\boldsymbol{a}\,(A\,\boldsymbol{a}\,A)}\\ \vdots\\ \chi\end{array}} \text{ then } \mathsf{l}_{\boldsymbol{a}}\,\phi \equiv \boxed{\begin{array}{c}\mathsf{l}_{\boldsymbol{a}}\,\psi\\ \vdots\\ A\,\boldsymbol{a}\,A\\ \vdots\\ \mathsf{l}_{\boldsymbol{a}}\,\chi\end{array}} \text{ and } \mathsf{r}_{\boldsymbol{a}}\,\phi \equiv \boxed{\begin{array}{c}\mathsf{r}_{\boldsymbol{a}}\,\psi\\ \vdots\\ A\,\boldsymbol{a}\,A\\ \vdots\\ \mathsf{r}_{\boldsymbol{a}}\,\chi\end{array}}.$$

If an atom is unnested in a derivation ϕ, then that atom will be either unnested or shallowly nested in a derivation produced by eliminating the unit-equality steps from ϕ via the construction given in Section 4.

▶ Remark 27. For any derivation ϕ and any atom $\boldsymbol{a}$ that is shallowly nested in ϕ, the projected derivations $\mathsf{l}_{\boldsymbol{a}}\,\mathsf{l}_{\boldsymbol{a}}\,\phi$, $\mathsf{r}_{\boldsymbol{a}}\,\mathsf{l}_{\boldsymbol{a}}\,\phi$, $\mathsf{l}_{\boldsymbol{a}}\,\mathsf{r}_{\boldsymbol{a}}\,\phi$, and $\mathsf{r}_{\boldsymbol{a}}\,\mathsf{r}_{\boldsymbol{a}}\,\phi$ contain no occurrences of $\boldsymbol{a}$, and so none contains any cuts on $\boldsymbol{a}$. If $A = 1$ then $\mathsf{l}_{\boldsymbol{a}}\,A\,\boldsymbol{a}\,\mathsf{r}_{\boldsymbol{a}}\,A = 1$ for any formula A and any atom $\boldsymbol{a}$.

▶ **Proposition 28.** *For every open formula A in which every variable occurs exactly once, and every atom $\boldsymbol{a}$, there exists a cut-free derivation in* KDTS

$$\chi \equiv \boxed{\begin{array}{c}\mathsf{l}_{\boldsymbol{a}}\,A\,\boldsymbol{a}\,\mathsf{r}_{\boldsymbol{a}}\,A\\ \|\\ [v\,\boldsymbol{a}\,v|v]_V\,A\end{array}}\quad,$$

for the set of variables $V = \underline{A} \setminus (\underline{\mathsf{l}_{\boldsymbol{a}}\,A} \cup \underline{\mathsf{r}_{\boldsymbol{a}}\,A})$.

Proof. The construction follows the same structure as the Merge Lemma. Cut-freeness follows from the fact that all inference rules will be of the form $\alpha\check{\boldsymbol{a}}$, for each connective α in A. ◀

▶ **Theorem 29** (Cut Elimination)**.** *For every* KDTS *proof ϕ in which each atom is either unnested or shallowly nested and whose unit factorisation is $\pi\boxed{\begin{array}{c}A\\ \|\\ B\end{array}}$, we can build a cut-free proof of $\pi\sigma B$ such that $\pi\sigma B = \pi B$.*

Proof. We enumerate the atoms in ϕ on which there is a cut $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$, let $A_0 \equiv A$, $B_0 \equiv B$, and $\phi_0 \equiv \boxed{\begin{array}{c}A\\ \|\\ B\end{array}}$. We then build $\begin{array}{c}A_i\\ \phi_i\,\|\\ B_i\end{array}$ from $\begin{array}{c}A_{i-1}\\ \phi_{i-1}\,\|\\ B_{i-1}\end{array}$ by eliminating any cuts on $\boldsymbol{a}_i$ via the following constructions:

If $\boldsymbol{a}_i$ is unnested in ϕ_{i-1} then we build:

$$\phi_i \equiv \boxed{\begin{array}{c}\mathsf{l}_{\boldsymbol{a}_i}\,\phi_{i-1}\,\boldsymbol{a}_i\,\mathsf{r}_{\boldsymbol{a}_i}\,\phi_{i-1}\\ \chi_i\,\|\\ [v\,\boldsymbol{a}_i\,v|v]_{V_i}\,B_{i-1}\end{array}}$$

where χ_i is the cut-free derivation given by Proposition 28.

If $\boldsymbol{a}_i$ is shallowly nested in ϕ_{i-1} then we build:

$$\phi_i \equiv \dfrac{\dfrac{\mathsf{l}_{\boldsymbol{a}_i}\,\mathsf{l}_{\boldsymbol{a}_i}\,\phi_{i-1}\;\boldsymbol{a}_i\;\mathsf{r}_{\boldsymbol{a}_i}\,\mathsf{l}_{\boldsymbol{a}_i}\,\phi_{i-1}}{\begin{array}{c}\chi_i' \,\|\\ [v\;\boldsymbol{a}_i\;v|v]_{V_i'}\;\mathsf{l}_{\boldsymbol{a}_i}\,B_{i-1}\end{array}}\,\boldsymbol{a}_i\,\dfrac{\mathsf{l}_{\boldsymbol{a}_i}\,\mathsf{r}_{\boldsymbol{a}_i}\,\phi_{i-1}\;\boldsymbol{a}_i\;\mathsf{r}_{\boldsymbol{a}_i}\,\mathsf{r}_{\boldsymbol{a}_i}\,\phi_{i-1}}{\begin{array}{c}\chi_i'' \,\|\\ [v\;\boldsymbol{a}_i\;v|v]_{V_i''}\;\mathsf{r}_{\boldsymbol{a}_i}\,B_{i-1}\end{array}}}{\begin{array}{c}\chi_i \,\|\\ [v\;\boldsymbol{a}_i\;v|v]_{V_i}\,[v\;\boldsymbol{a}_i\;v|v]_{V_i'}\,[v\;\boldsymbol{a}_i\;v|v]_{V_i''}\,B_{i-1}\end{array}}$$

where χ_i, χ_i', and χ_i'' are the cut-free derivations given by Proposition 28.

Then $\pi\phi_n$ contains no cuts on any atom and $\pi\,\mathsf{cn}\,\phi_n \equiv \pi\sigma B$ where σ is a substitution that does not change the value of the formula.

If a free variable appears anywhere in a proof it must also appear in its premise. However, a formula with a free variable cannot be equal to $\mathsf{1}$, therefore a proof in KDTS cannot contain any free variable and we have that $\pi\,\mathsf{pr}\,\phi_n = \mathsf{1}$. ◀

6 Conclusion

We have shown that a strictly linear subatomic system for propositional classical logic can be obtained by eliminating all unit-equality rules and controlling the complexity using explicit substitutions. Furthermore, we have shown that this strictly linear systems allows for a straightforward cut elimination procedure.

Although we do not define a non-subatomic proof system for propositional classical logic, using the interpretation map given in [2], we can construct a corresponding proof in KDTS from a non-subatomic proof and then by Theorem 29, we can eliminate the cuts from that proof to obtain a cut-free proof that can then be translated, preserving cut-freeness, back into the non-subatomic system.

One of the strengths of subatomic logic is that it can describe normalisation procedures that apply to a wide range of logics. However, in this paper we focus almost entirely on propositional classical logic. This is because our primary investigation is into the complexity of strictly linear proof systems with explicit substitutions, and by working in classical logic we are able to compare against the benchmark systems of Frege and substitution Frege [11]. It is nevertheless our intention that these ideas be extended to a wider range of logics, including first- and higher-order logics.

References

1 Andrea Aler Tubella. *A Study of Normalisation Through Subatomic Logic*. PhD thesis, University of Bath, 2017. URL: `https://people.bath.ac.uk/ag248/aat/phd.pdf`.

2 Andrea Aler Tubella and Alessio Guglielmi. Subatomic proof systems: Splittable systems. *ACM Transactions on Computational Logic*, 19(1):5:1–33, 2018. `doi:10.1145/3173544`.

3 Andrea Aler Tubella, Alessio Guglielmi, and Benjamin Ralph. Removing cycles from proofs. In Valentin Goranko and Mads Dam, editors, *26th EACSL Annual Conference on Computer Science Logic (CSL)*, volume 82 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. `doi:10.4230/LIPIcs.CSL.2017.9`.

4 Chris Barrett and Alessio Guglielmi. A subatomic proof system for decision trees. *ACM Transactions on Computational Logic*, 23(4):26:1–25, 2022. `doi:10.1145/3545116`.

5 Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10(2):14:1–34, 2009. `doi:10.1145/1462179.1462186`.

6 Paola Bruscoli, Alessio Guglielmi, Tom Gundersen, and Michel Parigot. Quasipolynomial normalisation in deep inference via atomic flows and threshold formulae. *Logical Methods in Computer Science*, 12(1):5:1–30, 2016. doi:10.2168/LMCS-12(2:5)2016.

7 Kai Brünnler. Two restrictions on contraction. *Logic Journal of the IGPL*, 11(5):525–529, 2003. doi:10.1093/jigpal/11.5.525.

8 Kai Brünnler. Locality for classical logic. *Notre Dame Journal of Formal Logic*, 47(4):557–580, 2006. doi:10.1305/ndjfl/1168352668.

9 Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 2250 of *Lecture Notes in Computer Science*, pages 347–361. Springer, 2001. doi:10.1007/3-540-45653-8_24.

10 Andrea Condoluci, Beniamino Accattoli, and Claudio Sacerdoti Coen. Sharing equality is linear. In *Proceedings of the 21st International Symposium on Principles and Practice of Declarative Programming*, PPDP '19, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3354166.3354174.

11 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.

12 Alessio Guglielmi. Formalism B, 2004. URL: https://people.bath.ac.uk/ag248/p/AG13.pdf.

13 Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1):1:1–64, 2007. doi:10.1145/1182613.1182614.

14 Alessio Guglielmi and Tom Gundersen. Normalisation control in deep inference via atomic flows. *Logical Methods in Computer Science*, 4(1):9:1–36, 2008. doi:10.2168/LMCS-4(1:9)2008.

15 Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In Jean-Pierre Jouannaud, editor, *25th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 284–293. IEEE, 2010. doi:10.1109/LICS.2010.12.

16 Tom Gundersen, Willem Heijltjes, and Michel Parigot. Atomic lambda calculus: A typed lambda-calculus with explicit sharing. In Orna Kupferman, editor, *28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 311–320. IEEE, 2013. doi:10.1109/LICS.2013.37.

17 M.S. Paterson and M.N. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, 1978. doi:10.1016/0022-0000(78)90043-0.

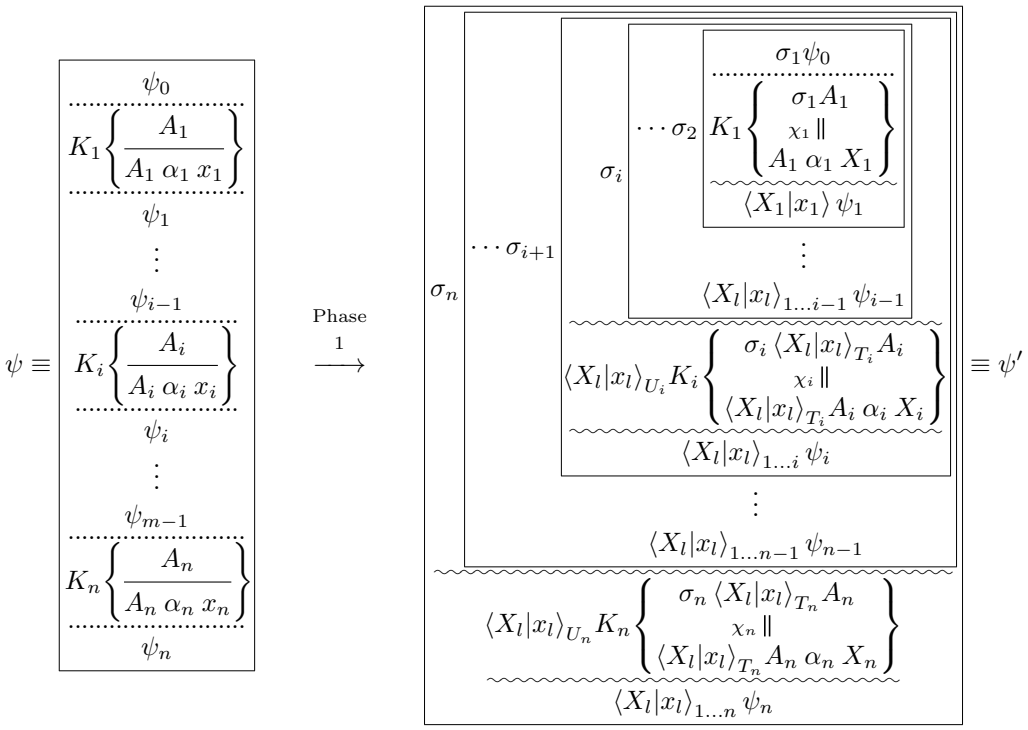
18 Alessio Santamaria. *Towards a Godement calculus for dinatural transformations*. PhD thesis, University of Bath, Somerset, UK, 2019. URL: https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.787523.

19 Lutz Straßburger. From deep inference to proof nets via cut elimination. *Journal of Logic and Computation*, 21(4):589–624, 2011. doi:10.1093/logcom/exp047.

20 Alwen Tiu. A system of interaction and structure II: The need for deep inference. *Logical Methods in Computer Science*, 2(2):4:1–24, 2006. doi:10.2168/LMCS-2(2:4)2006.

A Omitted proofs

Proof of Theorem 20. We refer to Figures 1 and 2. Given a derivation ϕ that contains inference steps in System Keq, we extract all the units into a substitution π, i.e. we obtain a derivation ψ such that $\phi \equiv \pi\psi$, where π is an actual substitution and ψ is open, i.e. it does not contain units. We assume that different occurrences of a unit or variable in each section of ϕ are assigned by π to different variables, and all variables so created are fresh. Moreover, π is such that all the inference steps in ψ except for those in System Keq remain valid, i.e. corresponding units and variables in the premise and the conclusion of a step are assigned the same variable. To be valid, each equality step of ϕ in Keq needs at least one unit that does not appear either in the premise or the conclusion, therefore ψ is not necessarily a derivation in KDTeq.



Where:

$$\sigma_i = [v\ \alpha_i\ x_i|v]_{\underline{A_i}}$$

$$\breve{A}_i^C \equiv [C|v]_{\underline{A_i}\setminus\{x_1,\ldots,x_{i-1}\}}\ \breve{A}_i$$

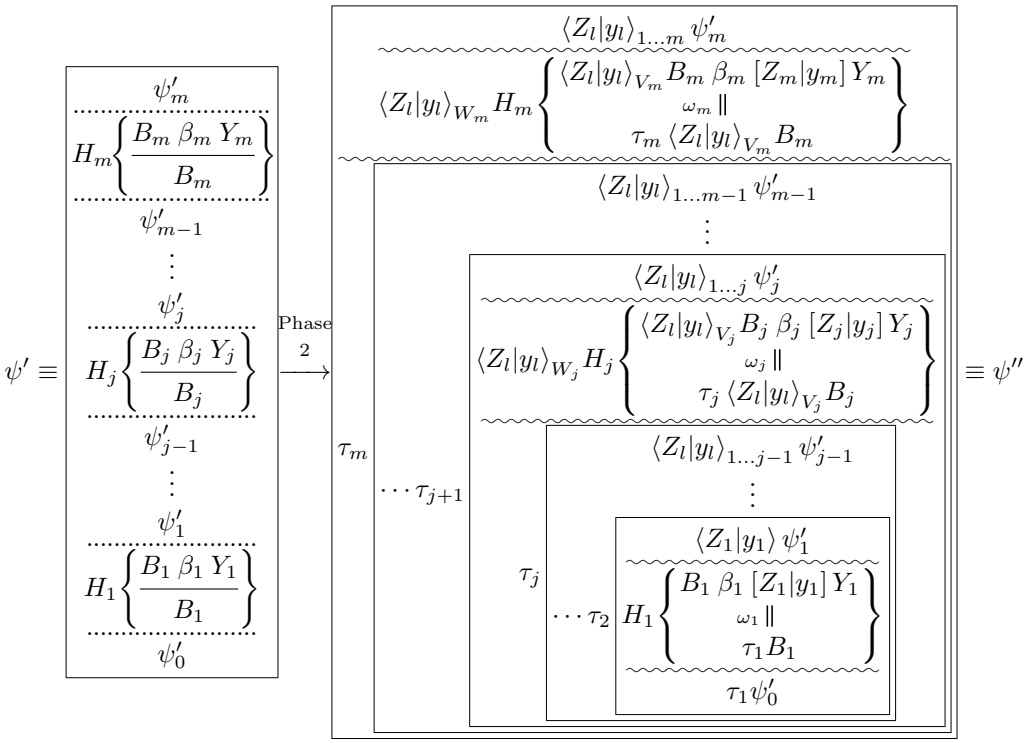
$$X_i \equiv \left\langle \breve{A}_1^{x_i} \middle| x_1 \right\rangle \cdots \left\langle \breve{A}_{i-1}^{x_i} \middle| x_{i-1} \right\rangle \breve{A}_i^{x_i}$$

$$T_i = \{x_1, \ldots, x_{i-1}\} \cap \underline{A_i}$$

$$U_i = \{x_1, \ldots, x_{i-1}\} \setminus \underline{A_i}$$

χ_i is given in Figure 3.

Figure 1 Phase 1 of the construction in Theorem 20.



Where:

$$\underline{Y_j} = \{y_j\}$$

$$\tau_j = \langle Y_j|y_j\rangle\,[v\ \beta_j\ y_j|v]_{\underline{B_j}}$$

$$\widehat{B}_j^C \equiv [C|v]_{\underline{B_j}\setminus\{y_1,\ldots,y_{j-1}\}}\,\widehat{B}_j$$

$$Z_j \equiv \left\langle \widehat{B}_1^{y_j}\middle|y_1\right\rangle \cdots \left\langle \widehat{B}_{j-1}^{y_j}\middle|y_{j-1}\right\rangle \widehat{B}_j^{y_j}$$

$$V_j = \{y_1,\ldots,y_{j-1}\} \cap \underline{B_j}$$

$$W_j = \{y_1,\ldots,y_{j-1}\} \setminus \underline{B_j}$$

ω_j is given in Figure 3.

Figure 2 Phase 2 of the construction in Theorem 20.

We first consider the equality steps that are instances of $=_1$, $=_3$, $=_5$, $=_7$, $=_9$ or $=_{11}$, as given in the definition of Keq; that is, those unit-equality inference steps which create a unit travelling downwards in the derivation. Let $x_1, \ldots, x_n$ be the variables in ψ that correspond to one of the units in those steps, via π. For rules $=_5$ and $=_{11}$, there are two choices and we pick one at random. Figure 1 shows $x_1, \ldots, x_n$ to the right of the α_is but we assume that they might be to the left, without prejudice to this proof. Without loss of generality, we assume that the sections of ψ containing the invalid inference steps are arranged as in the figure. Under the assumptions on π mentioned above, no variable x_i appears in formulae A_1, $\ldots$, A_i, for $1 \leq i \leq n$; on the other hand, x_i might appear in $A_{i+1}, \ldots, A_n$.

We build $\phi' \equiv \pi\psi''$, where ψ'' is obtained from ψ in two phases. Phase 1 and Phase 2 perform similar operations on all the invalid inference steps of ψ: in Phase 1 we fix some of them via down-merges and in Phase 2 we fix the remaining ones via up-merges. Both phases produce substitutions that are propagated through the derivation. Some of these substitutions might conflict; indeed, consider the following situation:

$$\psi \equiv \pi \left[\begin{array}{c} \psi_1 \\ \hdashline K\left\{=\dfrac{A_i}{A_i\ \alpha_i\ x_i}\right\} \\ \hdashline \psi_2 \\ \hdashline H\left\{=\dfrac{B_j\ \beta_j\ x_i}{B_j}\right\} \\ \hdashline \psi_3 \end{array}\right] .$$

Here, x_i would be assigned an instance of $\breve{A}_i$ for a down-merge at the top and an instance of $\widehat{B}_j$ for an up-merge at the bottom. By Lemma 16, these conflicting substitutions can be reconciled via the eversion construction

$$\begin{array}{l} \left[\widehat{B}_j \middle| v\right]_{\underline{A_i}} \breve{A}_i \\ \qquad \| \\ \left[\breve{A}_i \middle| v\right]_{\underline{B_j}} \widehat{B}_j \end{array} .$$

This eversion is implemented in Phase 2 (although it could have been implemented in Phase 1).

Phase 1. Each invalid inference step is replaced by an KDTS derivation χ_i, for $1 \leq i \leq n$, shown in Figure 3. Each variable x_i is replaced by a formula X_i, whose purpose is to make a down-merge of A_i via A_i and α_i possible. X_i is constituted by the formula $\breve{A}_i$ whose variables are to be replaced by formulae only containing the variable x_i. The idea is that the original variable x_i is expanded into a formula, X_i, whose structure matches the surroundings (to be amenable to a merge) but whose value remains that of x_i. Those variables of $\breve{A}_i$ that are not in $\{x_1, \ldots, x_{i-1}\}$ are set to x_i, in $\breve{A}_i^{x_i}$. The other variables of $\breve{A}_i$ must be replaced by substitutions that could match the formulae generated by the $\chi_1, \ldots, \chi_{i-1}$ above χ_1 in the derivation; those formulae are $\breve{A}_1^{x_1}, \ldots, \breve{A}_{i-1}^{x_{i-1}}$ and are matched by $\breve{A}_1^{x_i}, \ldots, \breve{A}_{i-1}^{x_i}$. At its top, χ_i generates the substitution σ_i, which does not change the value of the variables it applies to, and is propagated upwards in the derivation. At its bottom, χ_i generates the substitution $\langle X_i | x_i \rangle$, which is propagated downwards in the derivation and also does not change values because $\pi X_i = \pi x_i$. The rest of the construction in Figures 1 and 2 is bookkeeping, mainly relying on having maximally renamed apart all variables so that we can move substitutions without capturing any.

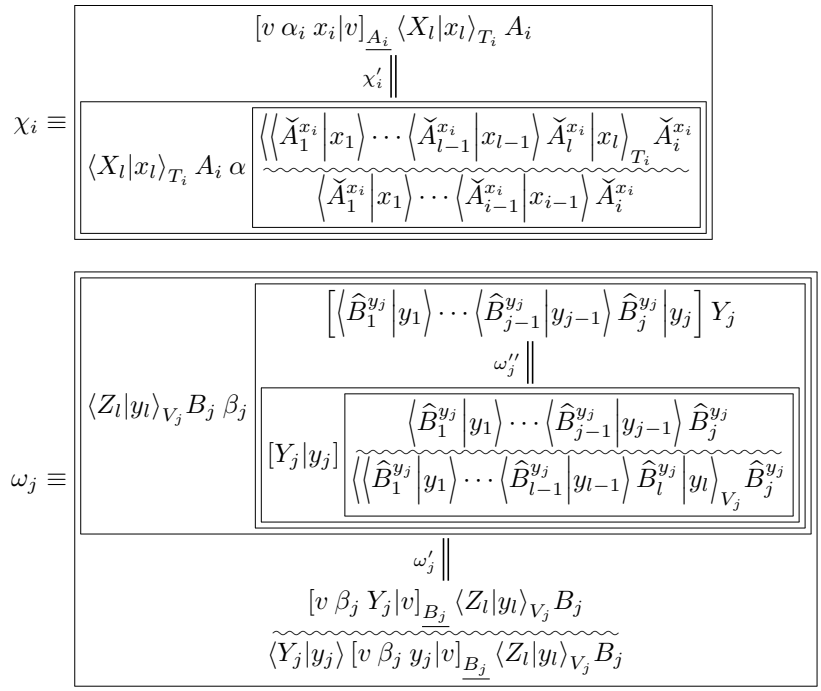


Figure 3 Auxiliary derivations for Phases 1 and 2 in Theorem 20.

Phase 2. Let us call ψ' the derivation produced in Phase 1. We operate on it in a similar way to Phase 1 but in the other direction. The equality steps to fix are those labelled $=_2$, $=_4$, $=_6$, $=_8$, $=_{10}$ and $=_{12}$ in the definition of Keq; that is, those unit-equality inference steps that create a unit travelling upwards in the derivation. For $1 \leq j \leq m$, B_j takes the place of A_i and Y_j that of x_i. One difference is that now Y_j might be one of the formulae X_is, and not just a variable. That said, each Y_j still only contains one variable (potentially in multiple copies), say y_j, and we note that y_j does not appear in $B_1, \ldots, B_j$ and might appear in $B_{j+1}, \ldots, B_m$. In Phase 2, each formula Z_j plays the same role as X_i in Phase 1, and the derivation ω_j, shown in Figure 3, plays the same role as χ_i. There, ω'_j is an up-merge and ω''_j the eversion that we outlined above in this proof. The substitution τ_j is propagated below ω_j; unlike σ_i, τ_j contains an additional substitution $\langle Y_j | y_j \rangle$ but for the rest its role is similar. The result of Phase 2 is a derivation ψ'' in KDTS.

Complexity. We establish upper bounds for the width and height of ψ'. The width of ϕ, say w, dominates the size of $A_1, \ldots, A_n$, and its height, say h, is such that wh dominates n and m. The maximum section width w' of ψ' occurs in the conclusion of some down-merge χ'_i, let us say χ'_n (see also Lemma 15). Therefore,

$$\begin{aligned} w' = \mathsf{w}\,\psi' &\leq \left|\langle X_l | x_l \rangle_{U_n} K_n\{\ \}\right| + 2\left|\langle X_l | x_l \rangle_{T_n} A_n\right| \\ &\leq |K_n\{\ \}| + 2\left|\left\langle\left\langle \breve{A}_1^{x_l} \middle| x_1 \right\rangle \cdots \left\langle \breve{A}_{l-1}^{x_l} \middle| x_{l-1} \right\rangle \breve{A}_l^{x_l} \middle| x_l \right\rangle_{1\ldots n-1} A_n\right| \\ &\leq w + 2(w + 2w + \cdots + (n-1)w + w) \\ &= O(w^3 h^2) \quad . \end{aligned}$$

Because of Lemma 15, the height of χ'_n also is $O(w^3h^2)$, therefore the height h' of ψ' is $O(w^3h^2)$. Similarly, the maximum section width w'' of ψ'' occurs in the premise of some up-merge ω'_j, let us say ω'_m. Therefore,

$$\begin{aligned}
w'' = \mathsf{w}\,\psi'' &\leq \left|\langle Z_l|y_l\rangle_{W_m} H_m\{\ \}\right| + \left|\langle Z_l|y_l\rangle_{V_m} B_m\right| + \left|[Y_m|v]_{\underline{B_m}} \langle Z_l|y_l\rangle_{V_m} \widehat{B}_m\right| \\
&\leq |H_m\{\ \}| + (1+|Y_m|)\left|\left\langle\left\langle \breve{B}_1^{y_l}\middle|y_1\right\rangle \cdots \left\langle \breve{B}_{l-1}^{y_l}\middle|y_{l-1}\right\rangle \breve{B}_l^{y_l}\middle|x_l\right\rangle_{1\ldots m-1} B_m\right| \\
&\leq w' + (1+w')(w' + 2w' + \cdots + (m-1)w' + w') \\
&= O((w')^2(wh)^2) \\
&= O(w^7h^5) \quad ,
\end{aligned}$$

and this is the width of ϕ'. Because of Lemmas 15 and 16, the height of ω_j is also $O(w^7h^5)$, which dominates h', therefore the height of ϕ' is $O(w^7h^5)$. ◀

Completeness of First-Order Bi-Intuitionistic Logic

Dominik Kirst
Université Paris Cité, IRIF, Inria, Paris, France
Ben-Gurion University, Beer-Sheva, Israel

Ian Shillito
The Australian National University, Canberra, Ngunnawal & Ngambri Country, Australia

Abstract

We provide a succinct and verified completeness proof for first-order bi-intuitionistic logic, relative to constant domain Kripke semantics. By doing so, we make up for the almost-50-year-old substantial mistakes in Rauszer's foundational work, detected but unresolved by Shillito two years ago. Moreover, an even earlier but historically neglected proof by Klemke has been found to contain at least local errors by Olkhovikov and Badia, that remained unfixed due to the technical complexity of Klemke's argument. To resolve this unclear situation once and for all, we give a succinct completeness proof, based on and dualising a standard proof for constant domain intuitionistic logic, and verify our constructions using the Coq proof assistant to guarantee correctness.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Logic and verification; Theory of computation → Modal and temporal logics

Keywords and phrases bi-intuitionistic logic, first-order logic, completeness, Coq proof assistant

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.40

Supplementary Material *Software (Coq Code)*: `https://github.com/ianshil/FOBiInt`

Funding *Dominik Kirst*: Received funding from the European Union's Horizon research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101152583 and a Minerva Fellowship of the Minerva Stiftung Gesellschaft für die Forschung mbH.

1 Introduction

In the 1970s, Cecylia Rauszer provided foundations for bi-intuitionistic logic (first studied by Moisil [34]), an extension of intuitionistic logic with a binary operator $\multimap$ called exclusion, dual to the intuitionistic implication $\to$. Her work spanned over most approaches to non-classical logics, ranging from algebras [43, 45], Kripke semantics [44, 46, 47], sequent calculus [42], to Hilbert systems [43, 42]. The impressiveness and exhaustiveness of Rauszer's study of bi-intuitionistic logic is not only measured by the variety of fields she introduced bi-intuitionistic in, but by the analysis in each case of *both* the propositional and first-order logic.

Unfortunately, through time several mistakes were detected in Rauszer's work. First, her sequent calculus for propositional bi-intuitionistic logic was shown by Pinto and Uustalu [38] not to admit cut, contradicting her claim [42, Result 2.3]. To correct this, they provided a calculus based on sequents with richer structure, which they proved to admit cut. Secondly, a confusion around the status of the deduction theorem led Goré and Shillito [18] to notice the conflation in Rauszer's work of two *different* propositional bi-intuitionistic logics. This conflation resulted in an incorrect completeness proof for the propositional case, ultimately resolved by Goré and Shillito. Finally, the errors contained in the propositional case continue being present in Rauszer's work on the first-order case as noted by Shillito [50], who failed to fix the proof in this setting. So, to date, no completeness proof for first-order bi-intuitionistic logic (FOBIL) along the lines of Rauszer's argument is known.

To our knowledge, the only other candidate proof was given by Klemke in 1971 [30], thereby in fact predating Rauszer's work. He attributes the semantics of the logic to Grzegorczyk [19] and uses a Henkin-style argument to construct a universal model. However, its correctness is questioned by Olkhovikov and Badia [35], who write:

> "Incidentally, there is an alternative completeness argument by Klemke, where bi-intuitionistic predicate logic is studied possibly for the first time in print (and, as far as we know, independently from Rauszer's work) and that contains other errors."

As his proof strategy is technically involved and, being written in fairly old style (and German language), the presentation is rather inaccessible to a broader audience, it is hard to assess whether these errors are locally fixable or as substantially unfixable as Rauszer's.

We therefore opt for an alternative route to settle the completeness of FOBIL once and for all: we present a *succinct* proof based on standard techniques, coming in a modern (and English) presentation for easy assessment, and use the Coq proof assistant to *verify* our argument, therefore leaving no room for ambiguity and error.

In that vein, our formal investigation finally establishes solid foundations for FOBIL, and simultaneously tightly connects the provability of the *constant domain axiom* in this logic with constant domain models. That is, contrarily to the propositional case, first-order bi-intuitionistic logic is known not to be a conservative extension of first-order intuitionistic logic [48, p.56][32, 50]: it derives the constant domain axiom (CD), displayed below, which is not provable in the purely intuitionistic counterpart [16].

$$\forall x(\varphi(x) \vee \psi) \to (\forall x\varphi(x) \vee \psi) \tag{CD}$$

Here, the variable x is required not to occur freely in ψ. As the name suggests, this axiom characterises the *constant domain property* on models in the Kripke semantics for the intuitionistic language [19, 16, 36]. Rauszer suggested that this connection between the axiom and the property on models should also hold in the bi-intuitionistic setting [44, 48]. The first-order Kripke semantics she developed uses frames for intuitionistic logic satisfying the constant domain property, thus capturing the semantics for FOCDIL, i.e. first-order intuitionistic logic extended with the (CD) axiom. Our results provide a confirmation of Rauszer's suggestion by showing FOBIL complete relative to the constant domain semantics, notably settling the logic as a conservative extension of FOCDIL [48, p.57][5].

In fact, our presented completeness proof for bi-intuitionistic logic mostly follows the textbook proof of Gabbay, Shehtman, and Skvortsov [17] for FOCDIL. As our only actually novel idea, we observe that their use of a custom Lindenbaum lemma exploiting the (CD) axiom to obtain successor worlds in a universal model can be dualised, namely, to obtain also predecessor worlds, exploiting a dualisation of the (CD) axiom presented below.

$$(\exists x\varphi(x) \wedge \psi) \dashv\!\prec \exists x(\varphi(x) \wedge \psi) \tag{DCD}$$

While (CD) is used as a theorem, i.e. $\top \vdash$ (CD), we exploit the contradictory nature of (DCD) in our custom lemma, as it satisfies (DCD) $\vdash \bot$. The remaining argument is also streamlined to dispose of the usual Henkin-style syntax extensions to obtain a particularly succinct presentation that is feasible to verify in Coq with little technical overhead.

In summary, the contributions of the present paper are as follows:

- We give a succinct completeness proof for FOBIL based on standard techniques, closing a gap in the literature not featuring a single unquestionably correct proof.
- We illustrate the tight connection of FOBIL and FOCDIL, in that our completeness proof of the former extends and dualises the one of the latter.

- We provide a Coq mechanisation verifying all definitions and results in the paper for absolute clarity and correctness, hyperlinked within this paper via clickable icons.
- As a by-product, we contribute, to the best of our knowledge, the first mechanisation of the completeness of FOCDIL and the conservativity of FOBIL over FOCDIL.

After some preliminary remarks on our meta-theory based on constructive type theory in Section 2, we recall the syntax, deduction system, and semantics of FOBIL in Section 3, including a dedicated discussion of the different constant domain axioms. In Section 4, we then prove the three versions of Lindenbaum lemmas needed to establish completeness and conservativity in Section 5. We end with remarks on the Coq development as well as related and future work in Section 6.

2 Preliminaries

The forthcoming mathematical development can be performed in any standard meta-theoretical foundation. To be formally precise and close to the mechanisation, we work in the calculus of inductive constructions (CIC) [4, 37] underlying the Coq proof assistant [53] and briefly sketch the key features we need. The core of the system is a predicative hierarchy of computational types closed under the usual type formers like (dependent) function types and (dependent) pair types. CIC further comes with an impredicative universe $\mathbb{P}$ of propositions in which the above type formers take common logical notation. Inductive types and predicates can be formed via a general scheme, for instance to accommodate the types $\mathbb{N}$ of natural numbers, $\mathbb{B}$ of boolean values, and of finite lists X^* over a given type X.

The logic represented in $\mathbb{P}$ is constructive but also agnostic, so in particular the excluded middle $(\forall P : \mathbb{P}.P \vee \neg P)$ is not provable but it can be assumed consistently. As in this paper we are aiming at a minimalistic proof directed to an audience not necessarily interested in questions of constructivism, we in fact assume the excluded middle globally and highlight its uses in the most crucial cases. Moreover, we assume the axiom of unique choice to freely identify total functional relations $X \to Y \to \mathbb{P}$ with functions $X \to Y$ where convenient. That is, we effectively simulate a traditional foundation based on classical set-theory to make the text as accessible as possible to readers unfamiliar with constructive type theory.

3 Basics of Bi-intuitionistic Logic

We present the basics of first-order bi-intuitionistic logic: its syntax, axiomatic proof system, constant domain Kripke semantics, and known facts of relevance, mostly following the presentations in [50] and [51].

3.1 Syntax

As mentioned above, first-order bi-intuitionistic logic is expressed in the language of first-order intuitionistic logic extended with the exclusion operator $\dot{\prec}$. More formally:

▶ **Definition 1** ( ). *Fix a countable signature $\mathcal{S}$ of function symbols f and predicate symbols P, denoting their arities by $|f|$ and $|P|$, respectively. Let V be the countable type of variables $x, y, z : V$.*

The term and formula language for bi-intuitionistic logic is defined as follows:

$$\mathbb{T} \ ::= \ x \mid f(t_1, \dots, t_{|f|})$$

$$\mathbb{F} \ ::= \ P(t_1, \dots, t_{|P|}) \mid \dot{\bot} \mid \varphi \mathbin{\dot{\wedge}} \varphi \mid \varphi \mathbin{\dot{\vee}} \varphi \mid \varphi \mathbin{\dot{\to}} \varphi \mid \varphi \mathbin{\dot{\prec}} \varphi \mid \dot{\forall} x \varphi \mid \dot{\exists} x \varphi$$

We call a formula of the shape $P(t_1, \ldots, t_{|P|})$ an atomic formula. Here we use dots to distinguish the object-level connectives and quantifiers of bi-intuitionistic logic from the meta-level connectives and quantifiers of the ambient meta-logic. We define $\dot{\top} := (\dot{\bot} \dot{\to} \dot{\bot})$, as well as the abbreviations $\dot{\neg}\varphi := (\varphi \dot{\to} \dot{\bot})$ and $\dot{\sim}\varphi := (\dot{\top} \dot{\prec} \varphi)$, respectively called negation *and* weak negation*.*

The added binary operator $\varphi \dot{\prec} \psi$ is intended to be the dual of $\varphi \dot{\to} \psi$ and is usually read as "φ excludes ψ". Consequently, $\dot{\sim}$ is also defined dually to $\dot{\neg}$.

In the following, we use $t, t_0, t_1, \ldots$ for terms the greek letters $\varphi, \psi, \chi, \delta, \ldots$ for formulas and $\Gamma, \Delta, \Phi, \Psi \ldots$ for sets or lists of formulas, depending on the context. When Γ refers to a set of formulas, we write Γ, φ or φ, Γ to mean $\Gamma \cup \{\varphi\}$. For a set of formulas Γ, we define $\overline{\Gamma}$ as $\{\varphi : \varphi \notin \Gamma\}$, where $\varphi \notin \Gamma$ means $\neg(\varphi \in \Gamma)$.

For a formula φ we denote its set of free variables, i.e. under the scope of a corresponding quantifier by $FV(\varphi)$, and say that it is *closed* if $FV(\varphi) = \emptyset$. A set of formulas is closed if all formulas in Γ are closed. We denote by $\varphi[t/x]$ the substitution of the free occurrences of the variable x in φ by the term t. We sometimes stress that x is free in φ by using the notation $\varphi(x)$ and in such a context just writing ψ is meant to suggest that x is not free in ψ. In that regard, our convention for quantifier scopes is that $\dot{\forall} x \varphi \dot{\to} \psi$ refers to $(\dot{\forall} x \varphi) \dot{\to} \psi$ and not to $\dot{\forall} x (\varphi \dot{\to} \psi)$.

Finally, note that our language is built on countable sets of variables, function symbols and predicate symbols. In consequence, the set of formulas is recursively enumerable.

3.2 Axiomatic Calculus

The generalised Hilbert calculus FOBIH [50] (🦃) for FOBIL extends the one for intuitionistic logic, containing the axioms A_1 to A_9 (for the propositional basis, implicit here) and A_{14} to A_{16} (for the first-order basis), with the axioms A_{10} to A_{13} and the rule (wDN), shown in Figure 1. There, $\mathcal{A}$ in the rule (Ax) refers to the set of all instances of axioms. In the following we write $\Gamma \vdash \varphi$ to mean that the syntactic expression $\Gamma \vdash \varphi$, called a *consecution*, is provable in FOBIH, i.e. there is a tree of consecutions built using the rules in Figure 1 with instances of (Ax) and (El) as leaves. We also abbreviate $\neg(\Gamma \vdash \varphi)$ by $\Gamma \nvdash \varphi$. We formally define the logic FOBIL as the set $\{(\Gamma, \varphi) : \Gamma \vdash \varphi\}$.

Note that our calculus FOBIH is the calculus FOwBIH of [50].[1] In his work, he also considers a stronger system called FOsBIH, obtained by modifying the premise of the rule (wDN) to $\Gamma \vdash \varphi$. As the letters w and s are only used to distinguish the two calculi, we drop w in this paper for simplicity.

The name of the rule (Gen) stands for *Gen*eralisation, while the name of the rule (EC) stands for for *E*xistential *C*onditionalisation.

3.3 Basic Proof-Theoretic Results

Next, we present basic proof-theoretic results from the mechanisation of Shillito [50]. They express properties of the proof system FOBIH, some of which we use to prove completeness.

[1] More precisely, FOBIH is the calculus FOwBIH minus the axiom $\varphi \dot{\to} \top$. This deletion is caused by the fact that $\top$ is not a primitive connective of our language.

$$
\begin{array}{ll}
A_{10} & \varphi \dot{\to} (\psi \dot{\vee} (\varphi \dot{\prec} \psi)) \\
A_{11} & (\varphi \dot{\prec} \psi) \dot{\to} \dot{\sim}(\varphi \dot{\to} \psi) \\
A_{12} & ((\varphi \dot{\prec} \psi) \dot{\prec} \chi) \dot{\to} (\varphi \dot{\prec} (\psi \dot{\vee} \chi)) \\
A_{13} & \dot{\neg}(\varphi \dot{\prec} \psi) \dot{\to} (\varphi \dot{\to} \psi) \\
A_{14} & \dot{\forall} x(\psi \dot{\to} \varphi) \dot{\to} (\psi \dot{\to} \dot{\forall} x\varphi) \\
A_{15} & \dot{\forall} x\varphi \dot{\to} \varphi[t/x] \\
A_{16} & \varphi[t/x] \dot{\to} \dot{\exists} x\varphi
\end{array}
\qquad
\frac{\emptyset \vdash \varphi}{\Gamma \vdash \dot{\neg}\dot{\sim}\varphi}\ (\text{wDN})
\qquad
\frac{\Gamma \vdash \varphi}{\Gamma \vdash \dot{\forall} x\varphi}\ (\text{Gen})
\qquad
\frac{\Gamma \vdash \varphi \dot{\to} \psi}{\Gamma \vdash \dot{\exists} x\varphi \dot{\to} \psi}\ (\text{EC})
$$

$$
\frac{\varphi \in \mathcal{A}}{\Gamma \vdash \varphi}\ (\text{Ax})
\qquad
\frac{\varphi \in \Gamma}{\Gamma \vdash \varphi}\ (\text{El})
\qquad
\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \varphi \dot{\to} \psi}{\Gamma \vdash \psi}\ (\text{MP})
$$

Figure 1 Generalised Hilbert calculus FOBIH, where x is free in ψ and Γ in A_{14}, (Gen) and (EC).

Unsurprisingly, we can prove that FOBIL is a *finitary logic*: it satisfies identity (🐓), monotonicity (🐓), compositionality (🐓), structurality (🐓), and finiteness (🐓) [12, 31]. These properties are expressed below, where σ is a function substituting atomic formulas by composite formulas satisfying some properties[2] and $\subseteq_f$ is the finite subset relation.

Identity	$\varphi \in \Gamma \to \Gamma \vdash \varphi$
Monotonicity	$\Gamma \subseteq \Gamma' \to \Gamma \vdash \varphi \to \Gamma' \vdash \varphi$
Compositionality	$\Gamma \vdash \varphi \to (\forall \gamma \in \Gamma.(\Delta \vdash \gamma)) \to \Delta \vdash \varphi$
Structurality	$\Gamma \vdash \varphi \to \Gamma^\sigma \vdash \varphi^\sigma$
Finiteness	$\Gamma \vdash \varphi \to \exists \Gamma' \subseteq_f \Gamma.(\Gamma' \vdash \varphi)$

To present the next results in an elegant way, we introduce helpful derived notions.

▶ **Definition 2.** *Let Δ be a list of formulas. We define $\dot{\bigvee} : \mathbb{F}^* \to \mathbb{F}$ recursively on the structure of Δ by $\dot{\bigvee}[] := \dot{\bot}$ and $\dot{\bigvee}(\varphi :: \Delta') := \varphi \dot{\vee} (\dot{\bigvee}\Delta')$ (🐓). Analogously, we define $\dot{\bigwedge} : \mathbb{F}^* \to \mathbb{F}$ by $\dot{\bigwedge}[] := \dot{\top}$ and $\dot{\bigwedge}(\varphi :: \Delta') := \varphi \dot{\wedge} (\dot{\bigwedge}\Delta')$ (🐓).*

The function $\dot{\bigvee}$ essentially creates the disjunction of all members of a list of formulas, with an additional disjunct $\dot{\bot}$, the neutral element of $\dot{\vee}$. Using $\dot{\bigvee}$, we can bring consecutions $\Gamma \vdash \varphi$ to a fully symmetric setting via pairs of the shape $[\Gamma \mid \Delta]$, constituted of a left and right context.

▶ **Definition 3.** *We define the following:*

1. $\vdash [\Gamma \mid \Delta]$ *if* $\Gamma \vdash \dot{\bigvee}\Delta'$ *for some* $\Delta' \subseteq_f \Delta$ (🐓)*;*
2. $\nvdash [\Gamma \mid \Delta]$ *if* $\neg(\vdash [\Gamma \mid \Delta])$*, in which case we say that* $[\Gamma \mid \Delta]$ *is* relative consistent*.*

Note that the symmetry in our pairs is only simulated, as it ultimately relies on the asymmetry of consecutions $\Gamma \vdash \varphi$ which we hide via a derived notion. A similar illusion could be obtained by defining an axiomatic system on symmetric consecutions $\varphi \vdash \Delta$ as first-class citizens, and define $\vdash [\Gamma \mid \Delta]$ as the existence of $\Gamma' \subseteq_f \Gamma$ with $\dot{\bigwedge}\Gamma \vdash \Delta$. It would be interesting to see what a truly symmetric axiomatic calculus based on pairs would look like.

While our pairs are crucially used in the completeness proof, as we shall see, they are already convenient to express interesting properties of FOBIH.

[2] This function needs to commute with substitution of variables (🐓), but we omit these details as they are not in the scope of this paper.

▶ **Theorem 4.** *We have the following:*

1. $\vdash [\emptyset \mid \varphi \dot{\vee} \dot{\sim}\varphi]$
2. $\vdash [\emptyset \mid (\varphi \dot{-\!\!<} \psi) \dot{\to} \chi] \quad \leftrightarrow \quad \vdash [\emptyset \mid \varphi \dot{\to} (\psi \dot{\vee} \chi)]$
3. $\vdash [\Gamma, \varphi \mid \psi] \quad \leftrightarrow \quad \vdash [\Gamma \mid \varphi \dot{\to} \psi]$
4. $\vdash [\varphi \mid \psi, \Delta] \quad \leftrightarrow \quad \vdash [\varphi \dot{-\!\!<} \psi \mid \Delta]$
5. $$\frac{\vdash [\varphi \mid \Delta] \quad \vdash [\psi \dot{-\!\!<} \varphi \mid \Delta]}{\vdash [\psi \mid \Delta]}\ (DMP)$$

(1) above shows that a bi-intuitionistic version of the law of excluded-middle holds in FOBIL (🦃). (2) is a syntactic analogue of the algebraic dual residuation property below (🦃).

$$\frac{a \leq b \vee c}{a \dot{-\!\!<} b \leq c}$$

(3) is the deduction-detachment theorem for FOBIL (🦃,🦃), while (4) is its *dual* deduction-detachment theorem (🦃,🦃). (5) is the *D*ual *M*odus *P*onens rule (🦃), which acts as (MP) but on the left-hand side of pairs and using $\dot{-\!\!<}$ instead of $\dot{\to}$.

3.4 Constant Domain Axioms

Early on, Rauszer noticed the provability in FOBIL of the constant domain axiom (CD), as shows the proof below on the left (🦃), where we rely on the commutativity of $\dot{\vee}$ (🦃).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash (\dot{\forall}x(\varphi(x) \dot{\vee} \psi)) \dot{\to} (\varphi(x) \vee \psi)}\,(\text{Ax})}{(\dot{\forall}x(\varphi(x) \dot{\vee} \psi)) \vdash \psi \vee \varphi(x)}\,\textit{Det. Thm}}{\dot{\forall}x(\varphi(x) \dot{\vee} \psi) \dot{-\!\!<} \psi \vdash \varphi(x)}\,\textit{Dual Ded. Thm.}}{(\dot{\forall}x(\varphi(x) \dot{\vee} \psi)) \dot{-\!\!<} \psi \vdash \dot{\forall}x\varphi(x)}\,(\text{Gen})}{\dot{\forall}x(\varphi(x) \dot{\vee} \psi) \vdash \dot{\forall}x\varphi(x) \dot{\vee} \psi}\,\textit{Dual Det. Thm.}}{\vdash \dot{\forall}x(\varphi(x) \dot{\vee} \psi) \dot{\to} (\dot{\forall}x\varphi(x) \dot{\vee} \psi)}\,\textit{Ded. Thm.}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash (\varphi(x) \dot{\wedge} \psi) \dot{\to} \dot{\exists}x(\varphi(x) \dot{\wedge} \psi)}\,(\text{Ax})}{\vdash \varphi(x) \dot{\to} \psi \dot{\to} \dot{\exists}x(\varphi(x) \dot{\wedge} \psi)}\,\textit{Currying}}{\vdash \dot{\exists}x\varphi(x) \dot{\to} \psi \dot{\to} \dot{\exists}x(\varphi(x) \dot{\wedge} \psi)}\,(\text{EC})}{\dot{\exists}x\varphi(x) \vdash \psi \dot{\to} \dot{\exists}x(\varphi(x) \dot{\wedge} \psi)}\,\textit{Det. Thm.}}{\dot{\exists}x\varphi(x) \dot{\wedge} \psi \vdash \dot{\exists}x(\varphi(x) \dot{\wedge} \psi)}\,\textit{Det. Thm.}}{(\dot{\exists}x\varphi(x) \dot{\wedge} \psi) \dot{-\!\!<} \dot{\exists}x(\varphi(x) \dot{\wedge} \psi) \vdash \dot{\bot}}\,\textit{Dual Ded. Thm.}$$

Moreoever, the bi-intuitionistic language enhances expressivity as it contains both connectives or quantifiers and their duals. This allows us to *dualise* formulas: recursively replace any connective or quantifier by its dual, and swap the formula on the left of an implication or exclusion by the one on the right. Therefore, we can dualise the axiom (CD) to obtain the dual constant domain dual-axiom (DCD). While the former is a *theorem* as it is provable from an empty left-context, equivalent to $\dot{\top}$, the latter is a *contradiction* as it proves the empty right-context, i.e. $\dot{\bot}$, as shown above on the right (🦃). We suspect that (DCD) plays a role to enforce constant domains in first-order dual intuitionistic logic, which is expressed in the language of FOBIL without $\dot{\to}$.

Both (CD) and (DCD) will be of crucial use in our completeness proof.

3.5 Constant Domain Kripke Semantics

We proceed to define a Kripke semantics for FOBIL which extends the one for FOCDIL with an interpretation of $\dot{-\!\!<}$. Note that the interpretation we use here is not the traditional one [48] formalised in [50], but an alternative put forward in [51].

Both the traditional semantics and ours are defined using (Kripke) models which are identical to the ones of FOCDIL, as shown below.

▶ **Definition 5** (🐓). *A model $\mathcal{M}$ is a tuple $(W, \leq, D, \mathcal{F}, \mathcal{P})$, where $(W, \leq)$ is a peordered set, D is a non-empty set called the* domain*, $\mathcal{F}$ is a function interpreting each function symbol f of arity n by a function $\mathcal{F}(f) : D^n \to D$, and $\mathcal{P}$ is a function interpreting, in each $w \in W$, each predicate symbol P of arity n by a set $\mathcal{P}(w, P) \subseteq D^n$ such that:*

$$\forall w \leq v. \forall P. \forall d_0, \dots, d_n \in D. \ ((d_0, \dots, d_n) \in \mathcal{P}(w, P) \ \to \ (d_0, \dots, d_n) \in \mathcal{P}(v, P))$$

An assignment α on D is a function $\alpha : V \to D$, and $\alpha[d/x]$ is the assignment α modified in x to output d. An assignment α is extended to the interpretation $\overline{\alpha}(t)$ of a term (🐓) recursively: $\overline{\alpha}(t) = \alpha(x)$ if $t = x$, and $\overline{\alpha}(t) = \mathcal{F}(f)(\overline{\alpha}(t_0), \dots, \overline{\alpha}(t_n))$ if $t = f(t_0, \dots, t_n)$.

Our Kripke semantics extends the usual forcing relation of first-order intuitionistic logic to incorporate $\dot{\prec}$ as follows.

▶ **Definition 6** (🐓). *Given a model $\mathcal{M} = (W, \leq, D, \mathcal{F}, \mathcal{P})$ and an assignment α for $\mathcal{M}$, we define the forcing relation $\mathcal{M}, w, \alpha \Vdash \varphi$ between a world $w \in W$ and a formula recursively by:*

$$\begin{array}{lcl}
\mathcal{M}, w, \alpha \Vdash P(t_0, \dots, t_n) & := & (\overline{\alpha}(t_0), \dots, \overline{\alpha}(t_n)) \in \mathcal{P}(w, P) \\
\mathcal{M}, w, \alpha \Vdash \dot{\bot} & := & \bot \\
\mathcal{M}, w, \alpha \Vdash \varphi \mathbin{\dot{\wedge}} \psi & := & \mathcal{M}, w, \alpha \Vdash \varphi \ \wedge \ \mathcal{M}, w, \alpha \Vdash \psi \\
\mathcal{M}, w, \alpha \Vdash \varphi \mathbin{\dot{\vee}} \psi & := & \mathcal{M}, w\, \alpha \Vdash \varphi \ \vee \ \mathcal{M}, w, \alpha \Vdash \psi \\
\mathcal{M}, w, \alpha \Vdash \varphi \mathbin{\dot{\to}} \psi & := & \forall v \geq w. (\mathcal{M}, v, \alpha \Vdash \varphi \ \to \ \mathcal{M}, v, \alpha \Vdash \psi) \\
\mathcal{M}, w, \alpha \Vdash \varphi \mathbin{\dot{\prec}} \psi & := & \neg(\forall v \leq w. (\mathcal{M}, v, \alpha \Vdash \varphi \ \to \ \mathcal{M}, v, \alpha \Vdash \psi)) \\
\mathcal{M}, w, \alpha \Vdash \dot{\forall} x \varphi & := & \forall d \in D. \mathcal{M}, w, \alpha[d/x] \Vdash \varphi \\
\mathcal{M}, w, \alpha \Vdash \dot{\exists} x \varphi & := & \exists d \in D. \mathcal{M}, w, \alpha[d/x] \Vdash \varphi
\end{array}$$

We abbreviate $\neg(\mathcal{M}, w, \alpha \Vdash \varphi)$ by $\mathcal{M}, w, \alpha \nVdash \varphi$.

Crucially, while the semantic clause for $\dot{\to}$ looks *forward* on the relation $\leq$, the clause for $\dot{\prec}$ looks *backwards*. This circumstance shows that FOBIL shares similarities with tense logic [39, 40, 41]. Additionally, observe that the use of constant domain models allows us to *localise* the interpretation of $\dot{\forall}$ in a single point, in contrast with the case of first-order intuitionistic logic where it is interpreted on all successors.

Note that our semantic clause for $\dot{\prec}$ is *intuitionistically weaker* but *classically equivalent* to the traditional clause for instance used by Rauszer [48]:

$$\exists v \leq w. (\mathcal{M}, v, \alpha \Vdash \varphi \ \wedge \ \mathcal{M}, v, \alpha \nVdash \psi)$$

Two points motivate this clause. First, to our eyes the duality between $\dot{\to}$ and $\dot{\prec}$ is more visibly expressed in our clause. Indeed, it is obtained in two steps by negating the clause for $\dot{\to}$, and by reversing the order between v and w, witnessing the tense logic flavour of $\dot{\prec}$. Secondly, our analysis led us to believe that the strength of the traditional clause more readily forces one to use non-constructive principles, notably in the proof of the Truth lemma (Lemma 17).

The main feature of the Kripke semantics for intuitionistic logic, i.e. persistence, is preserved in our semantics for FOBIL.

▶ **Lemma 7** (Persistence 🐓). *Let $\mathcal{M} = (W, \leq, D, \mathcal{F}, \mathcal{P})$ be a model. The following holds.*

$$\forall \alpha. \forall v, w \in W. \forall \varphi. (w \leq v \ \to \ \mathcal{M}, w, \alpha \Vdash \varphi \ \to \ \mathcal{M}, v, \alpha \Vdash \varphi)$$

Finally, we define the (local) consequence relation $\Gamma \vDash \varphi$ on our semantics (🐓):

$$\Gamma \vDash \varphi \ \text{ if } \ \forall \mathcal{M}. \forall \alpha. \forall w. (\mathcal{M}, w, \alpha \Vdash \Gamma \ \to \ \mathcal{M}, w, \alpha \Vdash \varphi)$$

Here $\mathcal{M}, w, \alpha \Vdash \Gamma$ means $\forall \gamma \in \Gamma. \mathcal{M}, w, \alpha \Vdash \gamma$. We then also abbreviate $\neg(\Gamma \vDash \varphi)$ by $\Gamma \nvDash \varphi$.

Crucially using classical reasoning, soundness of FOBIH is straightforwardly obtained.

▶ **Lemma 8** (Soundness 🦃). *If* $\Gamma \vdash \varphi$ *then* $\Gamma \vDash \varphi$.

Proof. We show $\Gamma \vDash \varphi$ by induction on a given derivation of $\Gamma \vdash \varphi$. The validity of the inference rules holds constructively using routine arguments and so does the validity of all axioms but A_{10}, A_{12}, and A_{13}, which rely on the excluded middle. We here only present the case of A_{10} for illustrative purposes.

In this case, we need to show that assuming $\mathcal{M}, w, \alpha \Vdash \varphi$ we have either $\mathcal{M}, w, \alpha \Vdash \psi$ or $\mathcal{M}, w, \alpha \Vdash \varphi \dot{\prec} \psi$. To proceed, we classical reasoning to distinguish whether $\mathcal{M}, w, \alpha \Vdash \psi$ or $\mathcal{M}, w, \alpha \nVdash \psi$. In the former case we are done, in the latter case we show $\mathcal{M}, w, \alpha \Vdash \varphi \dot{\prec} \psi$, so for a contradiction we assume that $\mathcal{M}, v, \alpha \Vdash \varphi$ implies $\mathcal{M}, v, \alpha \Vdash \psi$ for all predecessors $v \leq w$. For the choice $v := w$ we thus obtain $\mathcal{M}, w, \alpha \Vdash \psi$, in contradiction to the assumption $\mathcal{M}, w, \alpha \nVdash \psi$. ◀

4 A Forest of Lindenbaum Lemmas

In this section we are interested in the generation of *Henkin prime theories*, defined below.

▶ **Definition 9.** *We say that a set of formulas* Γ *is:*

- consistent *if* $\Gamma \nvdash \bot$ (🦃)*;*
- deductively closed *if* $\Gamma \vdash \varphi$ *implies* $\varphi \in \Gamma$ (🦃)*;*
- *a* theory *if it is consistent and deductively closed;*
- prime *if* $\varphi \dot{\vee} \psi \in \Gamma$ *implies* $\varphi \in \Gamma \vee \psi \in \Gamma$ (🦃)*;*
- $\dot{\exists}$-Henkin *if* $\dot{\exists} x \varphi \in \Gamma$ *then one can compute some* $k \in V$ *such that* $\varphi[k/x] \in \Gamma$ (🦃)*;*
- $\dot{\forall}$-Henkin *if* $\dot{\forall} x \varphi \notin \Gamma$ *then one can compute some* $k \in V$ *such that* $\varphi[k/x] \notin \Gamma$ (🦃)*;*
- Henkin *if it is* $\dot{\exists}$*-Henkin and* $\dot{\forall}$*-Henkin.*

Note that we deviate from the standard presentation of the Henkin properties by observing that they actually carry computational content. Later on we use Henkin prime theories as worlds of the canonical model we define to prove completeness.

Traditionally, this proof technique via canonical model construction requires us to connect any set Γ such that $\Gamma \nvdash \varphi$ to a point in the canonical model, i.e. a Henkin prime theory, extending Γ and not containing φ. We call this result the *standard Lindenbaum lemma*.

Additionally, on the way to completeness we are required to show that if a point in the canonical model does not contain $\varphi \dot{\to} \psi$ then we can find an *extension* of this point containing φ but not ψ. We call this result the *constant domain Lindenbaum lemma*.

Similarly, we also need to prove that the presence of $\varphi \dot{\prec} \psi$ in such a point entails the existence of a *restriction* of this point containing φ but not ψ. We call this result the *dual constant domain Lindenbaum lemma*.

In the remainder of this section, we prove these three flavours of Lindenbaum lemma, employing classical logic to describe the underlying extension processes via case distinctions.

4.1 Standard Lindenbaum Lemma

The standard lemma acts on pairs $\nvdash [\Gamma \mid \Delta]$ of closed sets of formulas, which allows us to treat $\Gamma \nvdash \varphi$ as special case. The sets Γ and Δ are required to be closed as we need enough "safe" variables to witness quantifiers throughout the enumeration.

▶ **Lemma 10** (Standard Lindenbaum Lemma 🐓). *For closed* Γ *and* Δ *such that* $\nvdash [\Gamma \mid \Delta]$, *there is a Henkin prime theory* $\Gamma' \supseteq \Gamma$ *such that* $\nvdash [\Gamma' \mid \Delta]$.

Proof. We construct Γ' by iteratively extending the pair $[\Gamma \mid \Delta]$, starting from $\Gamma_0 := \Gamma$ and $\Delta_0 := \Delta$ (🐓) and using an enumeration φ_n of formulas with the additional property that the n-th variable is not free in φ_k for all $k \leq n$.

$$[\Gamma_{n+1} \mid \Delta_{n+1}] := \begin{cases} [\Gamma_n \mid \dot{\exists} x\psi, \Delta_n] & \text{if } \varphi_n = \dot{\exists} x\psi \text{ and } \vdash [\dot{\exists} x\psi, \Gamma_n \mid \Delta_n] \\ [\psi[n/x], \dot{\exists} x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot{\exists} x\psi \text{ and } \nvdash [\dot{\exists} x\psi, \Gamma_n \mid \Delta_n] \\ [\Gamma_n \mid \psi[n/x], \dot{\forall} x\psi, \Delta_n] & \text{if } \varphi_n = \dot{\forall} x\psi \text{ and } \vdash [\dot{\forall} x\psi, \Gamma_n \mid \Delta_n] \\ [\dot{\forall} x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot{\forall} x\psi \text{ and } \nvdash [\dot{\forall} x\psi, \Gamma_n \mid \Delta_n] \\ [\varphi_n, \Gamma_n \mid \Delta_n] & \text{if } \nvdash [\varphi_n, \Gamma_n \mid \Delta_n] \\ [\Gamma_n \mid \varphi_n, \Delta_n] & \text{if } \vdash [\varphi_n, \Gamma_n \mid \Delta_n] \end{cases}$$

We then set $\Gamma' := \bigcup_{n:\mathbb{N}} \Gamma_n$ and name $\Delta' := \bigcup_{n:\mathbb{N}} \Delta_n$ (🐓). We observe $\Gamma' \supseteq \Gamma$ and $\Delta' \supseteq \Delta$ by construction (🐓). Before turning to the remaining properties one-by-one, note that $\nvdash [\Gamma_n \mid \Delta_n]$ is preserved inductively (🐓), ensuring that $\nvdash [\Gamma' \mid \Delta]$ (🐓) and hence the consistency of Γ' (🐓).

- For deductive closure (🐓), assume that $\Gamma' \vdash \varphi$. This entails that when φ is considered at n in the enumeration of formulae, then it must be added to Γ_{n+1}: indeed, we can prove that $\nvdash [\varphi, \Gamma_n \mid \Delta_n]$, as $\vdash [\varphi, \Gamma_n \mid \Delta_n]$ implies $\vdash [\Gamma' \mid \Delta']$, a contradiction, via compositionality as we have that $\Gamma' \vdash \psi$ for all $\psi \in \Gamma_n, \varphi$ (via the rule (El) or via assumption).
- For primeness (🐓), we assume that $\varphi \dot{\vee} \psi \in \Gamma'$. We make case distinctions on whether $\chi \in \Gamma'$ or $\chi \notin \Gamma'$ for $\chi \in \{\varphi, \psi\}$. Clearly, in the case where we have $\varphi \in \Gamma'$ or $\psi \in \Gamma'$ we are done. So, we are left to consider the case where $\varphi \notin \Gamma'$ and $\psi \notin \Gamma'$. From these assumptions, we obtain that $\varphi \in \Delta'$ and $\psi \in \Delta'$. Obviously, combined with $\varphi \dot{\vee} \psi \in \Gamma'$ the two last statements entail the contradiction $\vdash [\Gamma' \mid \Delta']$: We have that the list $[\varphi; \psi]$ is such that all its elements are in Δ', and $\Gamma' \vdash \dot{\bigvee}([\varphi; \psi])$ as $\dot{\bigvee}([\varphi; \psi]) = \varphi \dot{\vee} \psi \dot{\vee} \dot{\bot}$ is equivalent to $\varphi \dot{\vee} \psi \in \Gamma'$.
- To show that Γ' is $\dot{\exists}$-Henkin (🐓), we assume that $\dot{\exists} x\varphi \in \Gamma'$. When $\dot{\exists} x\varphi$ is considered at n in the enumeration of formulae, then it must be added to Γ_{n+1} as well as $\varphi[n/x]$: indeed, we can prove that $\nvdash [\dot{\exists} x\varphi, \Gamma_n \mid \Delta_n]$, as $\vdash [\dot{\exists} x\varphi, \Gamma_n \mid \Delta_n]$ implies $\dot{\exists} x\varphi \in \Delta_{n+1} \subseteq \Delta'$, hence $\vdash [\Gamma' \mid \Delta']$, a contradiction.
- To show that Γ' is $\dot{\forall}$-Henkin (🐓), we assume that $\dot{\forall} x\varphi \notin \Gamma'$. When $\dot{\forall} x\varphi$ is considered at n in the enumeration of formulae, then it must be added to Δ_{n+1} as well as $\varphi[n/x]$: indeed, we can prove that $\vdash [\dot{\exists} x\varphi, \Gamma_n \mid \Delta_n]$, as $\nvdash [\dot{\forall} x\varphi, \Gamma_n \mid \Delta_n]$ implies $\dot{\forall} x\varphi \in \Gamma_{n+1} \subseteq \Gamma'$, hence $\vdash [\Gamma' \mid \Delta']$, a contradiction. ◀

We now have sufficient machinery to generate a Henkin prime theory from a consistent closed theory. Next, we turn to the generation of prime Henkin theories from prime Henkin theories, via *extension* and *restriction*.

4.2 Constant Domain Lindenbaum Lemma

For this subsection and for the next, we generate new Henkin prime theories from previous Henkin prime theories. Here, we take a Henkin prime theory Γ and two formulas ψ_1 and ψ_2 and assume that $\Gamma, \psi_1 \nvdash \psi_2$. We aim at generating a Henkin prime theory Γ' which *extends* $\Gamma \cup \{\psi_1\}$ and does not contain ψ_2. We use this result in the Truth lemma, when assuming that $\psi_1 \dot{\to} \psi_2 \notin \Gamma$ or equivalently $\Gamma \nvdash \psi_1 \dot{\to} \psi_2$ or yet $\Gamma, \psi_1 \nvdash \psi_2$.

We cannot use the standard Lindenbaum lemma 10 to extend $\Gamma \cup \{\psi_1\}$, as it requires *closed* formulas. Given that Γ is Henkin, we are *prima facie* not ensured to have enough safe variables to extend it. However, we can extend $\Gamma \cup \{\psi_1\}$ using a trick relying on the (CD) axiom and the very fact that Γ is Henkin. This trick can be found in the book of Gabbay, Shehtman and Skvortsov [17, Section 7.2], where they use it for superintuitionistic logics based on the constant domain axiom.

We first establish a proof-theoretic lemma which isolates the use of the (CD) axiom.

▶ **Lemma 11.** *Let Γ be a $\dot\forall$-Henkin set of formulas and $\varphi(x), \psi_1, \psi_2$ be formulas.*
1. *If $\Gamma \nvdash (\dot\exists x\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2$, then one can compute k such that $(\varphi[k/x] \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2 \notin \Gamma$ (🐓).*
2. *If $\Gamma \nvdash \psi_1 \mathbin{\dot\to} (\dot\forall x\varphi(x) \mathbin{\dot\vee} \psi_2)$, then one can compute k such that $\psi_1 \mathbin{\dot\to} (\varphi[k/x] \mathbin{\dot\vee} \psi_2 \notin \Gamma)$ (🐓).*

Proof. We give both proofs in detail, noting that only (2) relies on the (CD) axiom.
1. It is sufficient to show that $\dot\forall x((\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2) \notin \Gamma$. Indeed, as Γ is $\dot\forall$-Henkin, one can then compute k with $((\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2)[k/x] \notin \Gamma$ and therefore $(\varphi[k/x] \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2 \notin \Gamma$. So suppose $\dot\forall x((\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2) \in \Gamma$, so in particular $\Gamma \vdash \dot\forall x((\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2)$. From there we can derive $\Gamma \vdash (\dot\exists x\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2$ in contradiction to the assumption using standard proof rules as follows: assuming $\varphi(x_0)$ for some particular x_0 together with ψ_1, we simply instantiate $\dot\forall x((\varphi(x) \mathbin{\dot\wedge} \psi_1) \mathbin{\dot\to} \psi_2)$ to x_0 and obtain ψ_2.
2. It is sufficient to show that $\dot\forall x(\psi_1 \mathbin{\dot\to} (\varphi(x) \mathbin{\dot\vee} \psi_2)) \notin \Gamma$, which again leverages the fact that Γ is $\dot\forall$-Henkin. So suppose $\dot\forall x(\psi_1 \mathbin{\dot\to} (\varphi(x) \mathbin{\dot\vee} \psi_2)) \in \Gamma$ and hence $\Gamma \vdash \dot\forall x(\psi_1 \mathbin{\dot\to} (\varphi(x) \mathbin{\dot\vee} \psi_2))$, we this time derive $\Gamma \vdash \psi_1 \mathbin{\dot\to} (\dot\forall x\varphi(x) \mathbin{\dot\vee} \psi_2)$ for a contradiction. So assuming ψ_1 and then applying the (CD) axiom, it remains to show $\dot\forall x(\varphi(x) \mathbin{\dot\vee} \psi_2)$, so $\varphi(x_0) \mathbin{\dot\vee} \psi_2$ for some arbitrary x_0. This follows directly from instantiating $\dot\forall(\psi_1 \mathbin{\dot\to} (\varphi(x) \mathbin{\dot\vee} \psi_2))$ to x_0. ◀

We can then show how to perform the extension of Γ as Henkin theory.

▶ **Lemma 12** (CD Lindenbaum Lemma 🐓)**.** *For any Henkin theory Γ and formulas ψ_1, ψ_2 such that $\Gamma, \psi_1 \nvdash \psi_2$, there is a Henkin prime theory $\Gamma' \supseteq \Gamma$ with $\psi_1 \in \Gamma'$ and $\psi_2 \notin \Gamma'$.*

Proof. We construct Γ' by iteratively constructing pairs $[\Gamma_n \mid \Delta_n]$, using any enumeration φ_n of formulas and letting $\Gamma_0 := \{\psi_1\}$ and $\Delta_0 := \{\psi_2\}$ (🐓):

$$[\Gamma_{n+1} \mid \Delta_{n+1}] := \begin{cases} [\Gamma_n \mid \dot\exists x\psi, \Delta_n] & \text{if } \varphi_n = \dot\exists x\psi \text{ and } \vdash [\dot\exists x\psi, \Gamma, \Gamma_n \mid \Delta_n] \\ [\psi[k/x], \dot\exists x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot\exists x\psi \text{ and } \nvdash [\dot\exists x\psi, \Gamma, \Gamma_n \mid \Delta_n] \\ & \text{and } k \text{ as obtained from (1) of Lemma 11} \\ [\dot\forall x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot\forall x\psi \text{ and } \vdash [\Gamma, \Gamma_n \mid \dot\forall x\psi, \Delta_n] \\ [\Gamma_n \mid \psi[k/x], \dot\forall x\psi, \Delta_n] & \text{if } \varphi_n = \dot\forall x\psi \text{ and } \nvdash [\Gamma, \Gamma_n \mid \dot\forall x\psi, \Delta_n] \\ & \text{and } k \text{ as obtained from (2) of Lemma 11} \\ [\varphi_n, \Gamma_n \mid \Delta_n] & \text{if } \nvdash [\varphi_n, \Gamma, \Gamma_n \mid \Delta_n] \\ [\Gamma_n \mid \varphi_n, \Delta_n] & \text{if } \vdash [\varphi_n, \Gamma, \Gamma_n \mid \Delta_n] \end{cases}$$

We then set $\Gamma' := \bigcup_{n:\mathbb{N}} \Gamma_n$ (🐓) and name $\Delta' := \bigcup_{n:\mathbb{N}} \Delta_n$. For this choice, $\Gamma' \supseteq \Gamma \cup \{\psi_1\}$ is by construction (🐓,🐓) and $\psi_2 \notin \Gamma'$ (🐓), or equivalently $\Gamma' \nvdash \psi_2$, follows since $\nvdash [\Gamma, \Gamma_n \mid \Delta_n]$ (🐓) and thus $\nvdash [\Gamma' \mid \Delta_n]$ is preserved inductively (🐓). The remaining properties of Γ' being a Henkin prime theory are established mostly as in Lemma 10.

- For deductive closure (🐓) and primeness (🐓), one can follow analogous arguments as in the respective claims of Lemma 10.
- To show that Γ' is $\dot\exists$-Henkin (🐓), we assume that $\dot\exists x\varphi \in \Gamma'$. When $\dot\exists x\varphi$ is considered at n in the enumeration of formulae, then it must be added to Γ_{n+1} as $\nvdash [\Gamma, \dot\exists x\varphi, \Gamma_n \mid \Delta_n]$ follows from $\nvdash [\Gamma' \mid \Delta_n]$. But then Γ_{n+1} by construction also contains $\varphi[k/x]$ for k obtained from (1) of Lemma 11 for the choice of $\psi_1 := \dot\bigwedge \Gamma_n$ and $\psi_2 := \dot\bigvee \Delta_n$.

- To show that Γ' is $\dot{\forall}$-Henkin (🐓) one can use an analogous argument, this time relying on (2) of Lemma 11. ◀

Note that we do not need primeness of the input theory Γ as it is obtained as a side-product of the iterative construction.

4.3 Dual Constant Domain Lindenbaum Lemma

Now, we aim at *restricting* a Henkin prime theory Γ containing $\psi_1 \dot{\prec} \psi_2$ into another such theory Γ' with ψ_1 but not ψ_2. This result is now motivated by the case of $\dot{\prec}$ in the Truth lemma. Note that once more we cannot use the standard Lindenbaum lemma 10.

While we easily imagine how to extend theories, as in Lemma 12, the restriction of a Henkin prime theory into a smaller one appears as a tricky and rather mysterious operation to perform. However, its familiarity is regained once seen as an extension, not of a theory but of the *complement* of a theory. Indeed, as $\Gamma' \subseteq \Gamma \leftrightarrow \overline{\Gamma} \subseteq \overline{\Gamma'}$ we restrict Γ by extending $\overline{\Gamma}$.

The next lemma, again isolating the use of constant domain axioms, relies on this insight, by involving the complement of a theory and exploiting the symmetry of our pairs $[\Phi \mid \Psi]$ by operating on their left.

▶ **Lemma 13.** *Let Γ be a $\dot{\exists}$-Henkin set of formulas and $\varphi(x), \psi_1, \psi_2$ be formulas.*

1. *If $\not\vdash [(\dot{\exists} x \varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2 \mid \overline{\Gamma}]$, then one can compute k with $(\varphi[k/x] \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2 \in \Gamma$ (🐓).*
2. *If $\not\vdash [(\psi_1 \dot{\prec} \dot{\forall} x \varphi(x)) \dot{\prec} \psi_2 \mid \overline{\Gamma}]$, then one can compute k with $(\psi_1 \dot{\prec} \varphi[k/x]) \dot{\prec} \psi_2 \in \Gamma$ (🐓).*

Proof. We give both proofs in detail, noting that (1) relies on the (DCD) dual-axiom and (2) relies on the (CD) axiom.

1. It is sufficient to show that $\dot{\exists} x((\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2) \in \Gamma$. Indeed, as Γ is $\dot{\exists}$-Henkin, we can thus compute k such that $((\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2)[k/x] \in \Gamma$ i.e. $((\varphi[k/x] \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2) \in \Gamma$. So, we assume for *reductio ad absurdum* that $\dot{\exists} x((\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2) \notin \Gamma$. We show that the latter implies $\vdash [(\dot{\exists} x \varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2 \mid \overline{\Gamma}]$, contradicting our initial assumption. By the dual deduction Theorem 4 it suffices to show $\vdash [\dot{\exists} x(x) \varphi \mathbin{\dot{\wedge}} \psi_1 \mid \psi_2, \overline{\Gamma}]$, proved as follows.

$$\frac{\vdash [\dot{\exists} x(\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \mid \psi_2, \overline{\Gamma}] \quad \vdash [(\dot{\exists} x \varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \dot{\exists} x(\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \mid \psi_2, \overline{\Gamma}]}{\vdash [\dot{\exists} x \varphi(x) \mathbin{\dot{\wedge}} \psi_1 \mid \psi_2, \overline{\Gamma}]} \text{ (DMP)}$$

   The right premise is nothing but an instance of the (DCD) dual-axiom, so we are left to prove the left premise. All we need to do is to apply the dual detachment Theorem 4 to reduce our goal to $\vdash [\dot{\exists} x(\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2 \mid \overline{\Gamma}]$, which obviously holds as we have $\dot{\exists} x((\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2) \in \overline{\Gamma}$ by our assumption $\dot{\exists} x((\varphi(x) \mathbin{\dot{\wedge}} \psi_1) \dot{\prec} \psi_2) \notin \Gamma$.
2. It is sufficient to show that $\dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2) \in \Gamma$, which again leverages the fact that Γ is $\dot{\exists}$-Henkin, i.e. $((\psi_1 \dot{\prec} \varphi[k/x]) \dot{\prec} \psi_2) \in \Gamma$. So, we assume for reductio that $\dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2) \notin \Gamma$ and show $\vdash [(\psi_1 \dot{\prec} \dot{\forall} x \varphi(x)) \dot{\prec} \psi_2 \mid \overline{\Gamma}]$, a contradiction. More precisely, we show $(\psi_1 \dot{\prec} \dot{\forall} x \varphi(x)) \dot{\prec} \psi_2 \vdash \dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2)$, noting that $\dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2) \in \overline{\Gamma}$. By the dual deduction theorem it is sufficient to show $\psi_1 \vdash \dot{\forall} x \varphi(x) \mathbin{\dot{\vee}} (\psi_2 \vee \dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2))$.
   We use the (CD) axiom to reduce our goal to $\psi_1 \vdash \dot{\forall} x(\varphi(x) \mathbin{\dot{\vee}} (\psi_2 \mathbin{\dot{\vee}} \dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2)))$, as x is not free in ψ_2 and $\dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2)$. We obtain a proof of the latter applying the rule (Gen), leaving us to prove $\psi_1 \vdash \varphi(x) \mathbin{\dot{\vee}} (\psi_2 \mathbin{\dot{\vee}} \dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2))$. This can easily be proved using the dual detachment theorem as $(\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2 \vdash \dot{\exists} x((\psi_1 \dot{\prec} \varphi(x)) \dot{\prec} \psi_2)$ holds. ◀

Turning back to the restriction of Γ, we note that $\psi_1 \dot{\prec} \psi_2 \in \Gamma$ is equivalent to $\nvdash [\psi_1 \dot{\prec} \psi_2 \mid \overline{\Gamma}]$ by consistency of Γ, and in turn to $\nvdash [\psi_1 \mid \psi_2, \overline{\Gamma}]$. So, to restrict Γ in a way that preserves ψ_1 but excludes ψ_2, we extend $\overline{\Gamma}$ using $\nvdash [\psi_1 \mid \psi_2, \overline{\Gamma}]$ as a stepping stone.

▶ **Lemma 14** (DCD Lindenbaum Lemma). *For any Henkin prime theory Γ and formulas ψ_1, ψ_2 with $\nvdash [\psi_1 \mid \psi_2, \overline{\Gamma}]$, there is a Henkin prime theory $\Gamma' \subseteq \Gamma$ with $\psi_1 \in \Gamma'$ and $\psi_2 \notin \Gamma'$.*

Proof. We construct Γ' by iteratively constructing pairs $[\Gamma_n \mid \Delta_n]$, using any enumeration φ_n of formulas and letting $\Gamma_0 := \{\psi_1\}$ and $\Delta_0 := \{\psi_2\}$ ():

$$[\Gamma_{n+1} \mid \Delta_{n+1}] := \begin{cases} [\Gamma_n \mid \dot{\exists} x\psi, \Delta_n] & \text{if } \varphi_n = \dot{\exists} x\psi \text{ and } \vdash [\dot{\exists} x\psi, \Gamma_n \mid \overline{\Gamma}, \Delta_n] \\ [\psi[k/x], \dot{\exists} x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot{\exists} x\psi \text{ and } \nvdash [\dot{\exists} x\psi, \Gamma_n \mid \overline{\Gamma}, \Delta_n] \\ & \text{and } k \text{ as obtained from (1) of Lemma 13} \\ [\dot{\forall} x\psi, \Gamma_n \mid \Delta_n] & \text{if } \varphi_n = \dot{\forall} x\psi \text{ and } \vdash [\Gamma_n \mid \dot{\forall} x\psi, \overline{\Gamma}, \Delta_n] \\ [\Gamma_n \mid \psi[k/x], \dot{\forall} x\psi, \Delta_n] & \text{if } \varphi_n = \dot{\forall} x\psi \text{ and } \nvdash [\Gamma_n \mid \dot{\forall} x\psi, \overline{\Gamma}, \Delta_n] \\ & \text{and } k \text{ as obtained from (2) of Lemma 13} \\ [\varphi_n, \Gamma_n \mid \Delta_n] & \text{if } \nvdash [\varphi_n, \Gamma_n \mid \overline{\Gamma}, \Delta_n] \\ [\Gamma_n \mid \varphi_n, \Delta_n] & \text{if } \vdash [\varphi_n, \Gamma_n \mid \overline{\Gamma}, \Delta_n] \end{cases}$$

We then set $\Gamma' := \bigcup_{n:\mathbb{N}} \Gamma_n$ (). For this choice, $\psi_1 \in \Gamma'$ () holds by construction and $\psi_2 \notin \Gamma'$ () follows since $\nvdash [\Gamma_n \mid \Delta_n, \overline{\Gamma}]$ () is preserved inductively and $\psi_2 \in \Delta_n$. We also have that $\Gamma' \subseteq \Gamma$ (), as if there is a $\chi \in \Gamma'$ but $\chi \notin \Gamma$ we get that at the point n in the enumeration where χ is added to form Γ_{n+1} we have $\vdash [\Gamma_{n+1} \mid \Delta_{n+1}, \overline{\Gamma}]$, a contradiction. As Γ' can be shown to be a prime theory (,) as in Lemma 12, we focus on its being Henkin.

- To show that Γ' is $\dot{\exists}$-Henkin (), we assume that $\dot{\exists} x\varphi \in \Gamma'$. When $\dot{\exists} x\varphi$ is considered at n in the enumeration of formulae, then it must be added to Γ_{n+1} as $\nvdash [\dot{\exists} x\varphi, \Gamma_n \mid \overline{\Gamma}, \Delta_n]$ follows from $\nvdash [\Gamma' \mid \overline{\Gamma}, \Delta_n]$. But then Γ_{n+1} by construction also contains $\varphi[k/x]$ for k obtained from (1) of Lemma 13 for the choice of $\psi_1 := \dot{\bigwedge} \Gamma_n$ and $\psi_2 := \dot{\bigvee} \Delta_n$.
- To show that Γ' is $\dot{\forall}$-Henkin () one can use an analogous argument, this time relying on (2) of Lemma 13. ◀

5 Completeness and Conservativity

Using the Lindenbaum lemmas of the previous section, we now first turn to the completeness of FOBIL relative to our constant domain semantics.

▶ **Theorem 15** (Completeness). *If $\Gamma \cup \{\varphi\}$ is closed and $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.*

We rely on a canonical model construction based on Henkin prime theories, defined below.

▶ **Definition 16** (). *The canonical model $\mathcal{M}^c = (W^c, \leq^c, D^c, \mathcal{F}^c, \mathcal{P}^c)$ is defined as follows:*

1. $W^c = \{\Gamma : \Gamma$ *is a Henkin prime theory*$\}$;
2. $\Gamma_1 \leq^c \Gamma_2$ *if* $\Gamma_1 \subseteq \Gamma_2$;
3. $D^c = \mathbb{T}$;
4. $\mathcal{F}^c(f)(t_0, \ldots, t_{|f|}) = f(t_0, \ldots, t_{|f|})$;
5. $\mathcal{P}^c(w, P)(t_0, \ldots, t_{|P|}) = \{(t_0, \ldots, t_{|P|}) \mid P(t_0, \ldots, t_{|P|}) \in w\}$.

The canonical assignment α^c is defined as $\alpha^c(x) = x$.

Note that the interpretation of terms in $\mathcal{M}^c$ through the canonical assignment α^c is the identity function: $\overline{\alpha^c}(t) = t$ follows from a simple induction on t ().

As foreshadowed, the two custom Lindenbaum lemmas come in action to show that the canonical model satisfies the crucial Truth lemma, relating elementhood and forcing.

▶ **Lemma 17** (Truth lemma 🦆). *For every $\Gamma \in W^c$ we have $\psi \in \Gamma$ iff $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \psi$.*

Proof. By induction on ψ. We consider the most interesting cases, and refer to the appendix for the remaining cases.

- $\psi = \varphi \dot{\to} \chi$: ($\Rightarrow$) Assume $\varphi \dot{\to} \chi \in \Gamma$. To show $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \dot{\to} \chi$, let $\Gamma' \in W^c$ such that $\Gamma \leq^c \Gamma'$, and assume $\mathcal{M}^c, \Gamma', \alpha^c \Vdash \varphi$. Then, we obtain $\varphi \in \Gamma'$ by induction hypothesis. Using $\Gamma \leq^c \Gamma'$, we get $\varphi \dot{\to} \chi \in \Gamma \subseteq \Gamma'$. Via deductive closure of Γ' we thus obtain $\chi \in \Gamma'$, hence $\mathcal{M}^c, \Gamma', \alpha^c \Vdash \chi$ using the induction hypothesis. So, we are done.
  ($\Leftarrow$) Assume $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \dot{\to} \chi$. Assume for reductio that $\varphi \dot{\to} \chi \notin \Gamma$. Then the constant domain Lindenbaum lemma 12 entails the existence of $\Gamma' \in W^c$ such that $\Gamma \leq^c \Gamma'$ and $\varphi \in \Gamma'$ and $\chi \notin \Gamma'$ (🦆). By induction hypothesis we get $\mathcal{M}^c, \Gamma', \alpha^c \Vdash \varphi$ and $\mathcal{M}^c, \Gamma', \alpha^c \nVdash \chi$. This contradicts $\Gamma \leq^c \Gamma'$ and $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \dot{\to} \chi$. So $\varphi \dot{\to} \chi \in \Gamma$.
- $\psi = \varphi \dot{\prec} \chi$: ($\Rightarrow$) Assume $\varphi \dot{\prec} \chi \in \Gamma$. The dual constant domain Lindenbaum lemma 14 entails the existence of $\Gamma' \in W^c$ with $\Gamma' \leq^c \Gamma$ and $\varphi \in \Gamma'$ and $\chi \notin \Gamma'$ (🦆). By induction hypothesis we get $\mathcal{M}^c, \Gamma', \alpha^c \Vdash \varphi$ and $\mathcal{M}^c, \Gamma', \alpha^c \nVdash \chi$. As $\Gamma' \leq^c \Gamma$ we get $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \dot{\prec} \chi$.
  ($\Leftarrow$) Assume $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \dot{\prec} \chi$. Then, there is $\Gamma' \in W^c$ such that $\Gamma' \leq^c \Gamma$ and $\mathcal{M}^c, \Gamma', \alpha^c \Vdash \varphi$ and $\mathcal{M}^c, \Gamma', \alpha^c \nVdash \chi$. By induction hypothesis we obtain that $\varphi \in \Gamma'$ and $\chi \notin \Gamma'$. Note that $\Gamma' \vdash \varphi \dot{\to} (\chi \dot{\vee} (\varphi \dot{\prec} \chi))$ using axiom A_{10}. Thus by applying (MP) we obtain $\Gamma' \vdash \chi \dot{\vee} (\varphi \dot{\prec} \chi)$, as we have $\Gamma' \vdash \varphi$ knowing $\varphi \in \Gamma'$. Via deductive closure and primeness we get $\chi \in \Gamma'$ or $\varphi \prec \chi \in \Gamma'$. But we know $\chi \notin \Gamma'$, so we have $\varphi \dot{\prec} \chi \in \Gamma'$. We finally obtain $\varphi \dot{\prec} \chi \in \Gamma$, as $\Gamma' \subseteq \Gamma$ given $\Gamma' \leq^c \Gamma$.
- $\psi := \dot{\forall} x\varphi$: ($\Rightarrow$) Assume $\forall x\varphi \in \Gamma$. To show $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\forall} x\varphi$ let $d \in D^c$. We need to show $\mathcal{M}^c, \Gamma, \alpha^c[d/x] \Vdash \varphi$. Note that $d \in \mathbb{T} = D^c$. Using $\forall x\varphi \in \Gamma$ and deductive closure we obtain $\varphi[d/x] \in \Gamma$. Thus, we apply the induction hypothesis to obtain $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi[d/x]$. We finally push the syntactic substitution to a modification of the assignment (🦆) to obtain $\mathcal{M}^c, \Gamma, \alpha^c[d/x] \Vdash \varphi$.
  ($\Leftarrow$) Assume $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\forall} x\varphi$. Assume for reductio that $\forall x\varphi \notin \Gamma$. The theory Γ being $\dot{\forall}$-Henkin, there is a $n \in \mathbb{N}$ such that $\varphi[n/x] \notin \Gamma$. By induction hypothesis we obtain $\mathcal{M}^c, \Gamma, \alpha^c \nVdash \varphi[n/x]$. But this is a contradiction as it implies that $\mathcal{M}^c, \Gamma, \alpha^c[n/x] \nVdash \varphi$ as explained above, while we have $\mathcal{M}^c, \Gamma, \alpha^c[n/x] \Vdash \varphi$ from $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\forall} x\varphi$.
- $\psi := \dot{\exists} x\varphi$: ($\Rightarrow$) Assume $\dot{\exists} x\varphi \in \Gamma$. The theory Γ being $\dot{\exists}$-Henkin, there is $n \in \mathbb{N}$ such that $\varphi[n/x] \in \Gamma$. By induction hypothesis we get $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi[n/x]$. This implies $\mathcal{M}^c, \Gamma, \alpha^c[n/x] \Vdash \varphi$ as argued above. Hence $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\exists} x\varphi$.
  ($\Leftarrow$) Assume $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\exists} x\varphi$. Thus there is a $d \in D^c$ such that $\mathcal{M}^c, \Gamma, \alpha^c[d/x] \Vdash \varphi$. Note that $d \in \mathbb{T} = D^c$. We reason as above to get that $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi[d/x]$. By induction hypothesis we obtain $\varphi[d/x] \in \Gamma$. We thus get $\dot{\exists} x\varphi \in \Gamma$ by deductive closure. ◀

Employing the Truth lemma we can now deduce completeness, also relying on the standard Lindenbaum lemma to extend the initial context into a point of the canonical model.

Proof of Theorem 15. Assume that $\Gamma \vDash \varphi$, and that $\Gamma \nvdash \varphi$ for reductio. As $\Gamma \cup \{\varphi\}$ is closed, the standard Lindenbaum lemma 10 conjointly with our last assumption ensure us of the existence of $\Gamma' \in W^c$ such that $\Gamma' \supseteq \Gamma$ and $\varphi \notin \Gamma'$ (🦆). By the Truth lemma 17 we obtain both $\mathcal{M}^c, \Gamma', \alpha^c \vdash \Gamma$ and $\mathcal{M}^c, \Gamma', \alpha^c \nvdash \varphi$, hence $\Gamma \nvdash \varphi$, a contradiction. ◀

We conclude with two results concerning FOCDIL that illustrate the close connection to our completeness proof for FOBIL. To this end, we write $\mathbb{F}_{\mathrm{CD}}$ for the usual syntax of first-order intuitionistic logic (i.e. $\mathbb{F}$ without $\dot{\prec}$) (🦆), $\vdash_{\mathrm{CD}}$ for the deduction system of FOCDIL (i.e. $\vdash$ without the axioms for $\dot{\prec}$ but extended with the (CD) axiom) (🦆, and $\vDash_{\mathrm{CD}}$ for the semantic consequence relation of FOCDIL (i.e. $\vDash$ without the clause for $\dot{\prec}$) (🦆). To avoid redundancy, we just give proof sketches and refer to the Coq code for full detail.

First, we prove the completeness of FOCDIL simply by a fragment of the proof of FOBIL.

▶ **Theorem 18** (CD Completeness 🐓). *If* $\Gamma \cup \{\varphi\}$ *is closed and* $\Gamma \vDash_{CD} \varphi$ *then* $\Gamma \vdash_{CD} \varphi$*, provided that* $\Gamma \cup \{\varphi\}$ *ranges over* $\mathbb{F}_{CD}$*.*

Sketch. Following exactly the same strategy as Theorem 15, now only relying on the standard Lindenbaum lemma 10 and the constant domain Lindenbaum lemma 12. ◀

Secondly, we deduce the conservativity of FOBIL over FOCDIL.

▶ **Corollary 19** (Conservativity 🐓). *If* $\Gamma \cup \{\varphi\}$ *is closed and* $\Gamma \vdash \varphi$ *then* $\Gamma \vdash_{CD} \varphi$*, provided that* $\Gamma \cup \{\varphi\}$ *ranges over* $\mathbb{F}_{CD}$*.*

Sketch. By composing soundness of FOBIL (Lemma 8) with completeness of FOCDIL (Theorem 18), using that obviously $\Gamma \vDash_{\mathrm{CD}} \varphi$ iff $\Gamma \vDash \varphi$ for $\Gamma \cup \{\varphi\}$ ranging over $\mathbb{F}_{\mathrm{CD}}$. ◀

6 Discussion

In this paper, we provide a succinct and verified completeness proof of FOBIL relative to its constant domain Kripke semantics. Consequentially, we formally establish the conservativity of FOBIL over FOCDIL, notably via the analogous completeness of the latter over the same semantics restricted to the intuitionistic language. We conclude with a brief discussion.

6.1 Coq Development

Our Coq development is based on the design of and is in the process of being integrated[3] into the Coq library for first-order logic [27], which has been developed to unify several projects concerned with different aspects of first-order logics [13, 14, 28, 26, 24, 23, 29]. It spans roughly 8000 lines of code, with about one half each for the separate FOBIL and FOCDIL developments. We globally assume a strong form of the excluded middle, namely $\forall P : \mathbb{P}.P + \neg P$, to enable the definition of functions by logical case distinction (justified by the consistency of the usual excluded middle and unique choice [56]) and left the particular formula enumeration as a parameter that will be obtained routinely from the library framework once merged.

Most notably, in comparison to the paper presentation, where we use named variables for legibility, the mechanisation is based on a de Bruijn encoding of binding [7] following the design of the Autosubst 2 tool [52], i.e. variables are replaced by indices referring to the amount of quantifiers shadowing their relevant binder. For instance, the formula $\dot{\forall} x \dot{\exists} y P(x, y)$ is represented as $\dot{\forall}\dot{\exists} P(1, 0)$, as x is bound by the $\dot{\forall}$ shadowed by the $\dot{\exists}$, whereas y is bound by the unshadowed $\dot{\exists}$. To illustrate just one of the advantages of this approach, in the representation of the deduction calculus, one can use lifting of de Bruijn indices to simulate the usual freshness conditions for variables. For example, the rule (Gen) is encoded as:

$$\frac{\Gamma[\uparrow] \vdash \varphi}{\Gamma \vdash \dot{\forall}\varphi}\ (\mathrm{Gen})$$

By shifting from Γ in the conclusion to $\Gamma[\uparrow]$ in the premise, we lift any free index n in Γ to its successor $n + 1$. As a consequence, the index 0 made free by the change from $\dot{\forall}\varphi$ to φ is not present in $\Gamma[\uparrow]$, thus creating a canonical "fresh" variable. Using this rule for instance allows a particularly easy monotonicity proof, as no on-the-fly renaming of variables is necessary.

[3] `https://github.com/uds-psl/coq-library-fol/pull/7`

Overall, the use of any proof assistant for our project not only provided the additional guarantee of correctness of our completeness proof but actually was worthwhile already in the mathematical development: for instance the dualised Lindenbaum lemma subject to Section 4.2 was developed incrementally starting from the non-dualised case, with the proof assistant pointing towards the remaining gaps while some proof scripts could be reused. The particular choice of Coq allowed to base our code on the design decisions of the existing library for first-order logic and, implementing a constructive foundation, in principle enables a constructive analysis extending [51], as described in the future work section.

6.2 Related Work

Bi-intuitionistic logic. As understood in this paper, bi-intuitionistic logic received some attention in computer science, notably through a formulae-as-types interpretation involving the notion of first-class coroutines [6] and in the context of image processing via its connection to mathematical morphology [49]. We mention another line of work [3, 1, 2, 10, 11] initiated by Wansing [54, 55] on a different bi-intuitionistic logic called 2Int, which is both proof-theoretically and philosophically motivated. The alternative interpretation of $\dot{\prec}$ in this logic allows for the study of the notions of falsification and verification.

Completeness proofs for FOBIL. Rauszer's proof of completeness for FOBIL [46] is erroneous for three main reasons. First, as in the propositional case two logics are conflated. This is noticed by the joint use of the deduction theorem, an exclusive property of the *weak* logic we study here, and of double negation $\neg\sim$ of formulas, an exclusive property of the *strong*. Secondly, her canonical model [46, p.66] is *rooted*, i.e. there is a point-root w for which any v is such that $w \leq v$. However, as noticed by Crolard [5] and confirmed by Shillito [50, Lemma 8.11.3], bi-intuitionistic logic is not complete relative to the class of rooted models. Thirdly, in her proof Rauszer relies on a result from Gabbay [15, Lemma 8.11.1] dealing with the language $\mathbb{F}$ without $\dot{\prec}$, $\dot{\vee}$ and $\dot{\exists}$. She dually proves it for $\mathbb{F}$ without $\dot{\to}$, $\dot{\wedge}$ and $\dot{\forall}$, and proceeds to illegitimately combine them on $\mathbb{F}$, outside of their application range.

In Klemke's proof strategy [30], the main construction is in Satz 6.1, where the extension of consistent pairs (M, N) of sets of formulas over an alphabet $\{x_1, x_2, \dots\}$ is described. The extension yields a family of maximal pairs (M_s, N_s) in the extended alphabet $\{x_1, x_2, \dots\} \cup \{y_1, y_2, \dots\}$ where s ranges over the partial order (U, Q) of strings over two copies of natural numbers ($\mathbb{N}$ and $\mathbb{N}^*$) such that $s \leq s'$ if s and s' agree on a prefix and from there continue in the separate copies. This order is called the "universal bush" and a universal model is then defined over the structure $(U, Q, \{x_1, x_2, \dots\} \cup \{y_1, y_2, \dots\})$, i.e. on the universal bush with the full alphabet as individuals, interprets variables with the identity and interprets atoms $P(x, y, z \dots)$ at s with $P(x, y, z \dots)$ in M_s. From Satz 6.1 the conclusion to completeness is standard. To date, we were neither able to identify an explicit use of the constant domain axioms, nor to confirm or refute the claim by Olkhovikov and Badia [35] concerning errors.

Finally, Shillito [50] tried, but failed, to correct Rauszer's work in Coq. More precisely, he gave an incomplete proof of completeness relying on two assumptions corresponding to our custom Lindenbaum lemmas 12 and 14. We consequently closed the gap in his proof.

Mechanisation of completeness proofs. There is a rather long list of works mechanising completeness proofs which for the most prominent case of first-order logic is summarised in [14] and [25]. The only mechanised completeness proofs for (propositional) bi-intuitionistic logic we are aware of are those by Shillito [50] as well as Shillito and Kirst [51].

Regarding other formalisms like bi-intuitionistic logic with a modal aspect, we are aware of the works in Coq of Doczkal and Smolka on CTL [9], Doczkal and Bard on converse PDL [8], and Hagemeier and Kirst on IEL [21], the work in HOL Light of Maggesi and Perini Brogi on the provability logic GL [33]. We finally mention the recent formalisation in Lean of Guo, Chen and Bentzen on propositional intuitionistic logic [20].

6.3 Future Work

While the purpose of this paper is to present the clearest and most minimalistic completeness proof for FOBIL, which for very natural reasons encompasses classical assumptions, we plan to continue in the spirit of Shillito and Kirst [51] to analyse which logical strength is exactly required. In their case of propositional bi-intuitionistic logic, they observe that the principle WLEMS, identified by Kirst [25] for the case of IEL and applied to first-order logic by Herbelin and Kirst [22], is equivalent to a weak but natural formulation of completeness. However, this observation relies on the property that theories obtained from the standard Lindenbaum lemma are negatively described (i.e. membership of formulas is characterised by non-derivability), while the custom Lindenbaum lemmas yield also positively described theories (membership of universally quantified formulas is characterised by derivability). Therefore it seems unlikely that an exactly analogous analysis is realistic and in fact it might be that the completeness of FOBIL requires a stronger fragment of classical meta-logic.

References

1 Sara Ayhan. Uniqueness of logical connectives in a bilateralist setting. In Martin Blicha and Igor Sedlár, editors, *The Logica Yearbook 2020*, pages 1–16. College Publications, 2021. doi:10.48550/arXiv.2210.00989.

2 Sara Ayhan. Meaning and identity of proofs in a bilateralist setting: A two-sorted typed lambda-calculus for proofs and refutations. *Journal of Logic and Computation*, 2024. doi:10.48550/arXiv.2307.01079.

3 Sara Ayhan and Heinrich Wansing. On synonymy in proof-theoretic semantics. the case of 2int. *Bulletin of the Section of Logic*, 52(2), 2023.

4 Thierry Coquand and Gérard Huet. *The calculus of constructions*. PhD thesis, INRIA, 1986. doi:10.1016/0890-5401(88)90005-3.

5 Tristan Crolard. Subtractive logic. *TCS*, 254:1-2:151–185, 2001. doi:10.1016/S0304-3975(99)00124-3.

6 Tristan Crolard. A formulae-as-types interpretation of subtractive logic. *Journal of Logic and Computation*, 14:4:529–570, 2004. doi:10.1093/logcom/14.4.529.

7 Nicolaas Govert De Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the church-rosser theorem. In *Indagationes mathematicae (proceedings)*, volume 75(5), pages 381–392. Elsevier, 1972.

8 Christian Doczkal and Joachim Bard. Completeness and Decidability of Converse PDL in the Constructive Type Theory of Coq. In *Certified Programs and Proofs*, Los Angeles, United States, January 2018. doi:10.1145/3167088.

9 Christian Doczkal and Gert Smolka. Completeness and decidability results for ctl in coq. In Gerwin Klein and Ruben Gamboa, editors, *Interactive Theorem Proving*, pages 226–241, Cham, 2014. Springer International Publishing. doi:10.1007/978-3-319-08970-6_15.

10 Sergey Drobyshevich. A bilateral hilbert-style investigation of 2-intuitionistic logic. *J. Log. Comput.*, 29(5):665–692, 2019. doi:10.1093/logcom/exz010.

11 Sergey Drobyshevich. Tarskian consequence relations bilaterally: some familiar notions. *Synthese*, 198(Suppl 22):5213–5240, 2021. doi:10.1007/s11229-019-02267-w.

12 Josep M. Font. *Abstract Algebraic Logic: An Introductory Textbook*. Studies in Logic and the Foundations of Mathematics. College Publications, 2016.

13 Yannick Forster, Dominik Kirst, and Gert Smolka. On synthetic undecidability in coq, with an application to the entscheidungsproblem. In *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 38–51, 2019. doi:10.1145/3293880.3294091.

14 Yannick Forster, Dominik Kirst, and Dominik Wehr. Completeness theorems for first-order logic analysed in constructive type theory: Extended version. *Journal of Logic and Computation*, 31(1):112–151, 2021. arXiv:2006.04399, doi:10.48550/arXiv.2006.04399.

15 Dov M. Gabbay. Applications of trees to intermediate logics. *Journal of Symbolic Logic*, 37(1):135–138, 1972. doi:10.2307/2272556.

16 Dov M. Gabbay. *Semantical investigations in Heyting's intuitionistic logic*, volume 148. D. Reidel Pub. Co, Dordrecht, Holland, 1981.

17 Dov M. Gabbay, Valentin B. Shehtman, and Dmitrij P. Skvortsov. *Quantification in Nonclassical Logic*, volume 153. Elsevier, London, Amsterdam, 1st edition, 2009.

18 Rajeev Goré and Ian Shillito. Bi-Intuitionistic Logics: A New Instance of an Old Problem. In *Advances in Modal Logic 13, papers from the thirteenth conference on "Advances in Modal Logic," held online, 24-28 August 2020*, pages 269–288, 2020. URL: http://www.aiml.net/volumes/volume13/Gore-Shillito.pdf.

19 Andrzej Grzegorczyk. A philosophically plausible formal interpretation of intuitionistic logic. *Indagationes Mathematicae*, 26:596–601, 1964.

20 Huayu Guo, Dongheng Chen, and Bruno Bentzen. Verified completeness in Henkin-style for intuitionistic propositional logic. In Bruno Bentzen, Beishui Liao, Davide Liga, Reka Markovich, Bin Wei, Minghui Xiong, and Tianwen Xu, editors, *Joint Proceedings of the Third International Workshop on Logics for New-Generation Artificial Intelligence and the International Workshop on Logic, AI and Law, September 8-9 and 11-12, 2023, Hangzhou*, pages 36–48. College Publications, 2023.

21 Christian Hagemeier and Dominik Kirst. Constructive and mechanised meta-theory of iel and similar modal logics. *Journal of Logic and Computation*, 32(8):1585–1610, 2022. doi:10.1093/logcom/exac068.

22 Hugo Herbelin and Dominik Kirst. New observations on the constructive content of first-order completeness theorems. In *29th International Conference on Types for Proofs and Programs*, 2023.

23 Marc Hermes and Dominik Kirst. An analysis of Tennenbaum's theorem in constructive type theory. In *International Conference on Formal Structures for Computation and Deduction*. LIPIcs, 2022. doi:10.4230/LIPIcs.FSCD.2022.9.

24 Johannes Hostert, Andrej Dudenhefner, and Dominik Kirst. Undecidability of dyadic first-order logic in Coq. In *International Conference on Interactive Theorem Proving*. LIPIcs, 2022. doi:10.4230/LIPIcs.ITP.2022.19.

25 Dominik Kirst. *Mechanised Metamathematics: An Investigation of First-Order Logic and Set Theory in Constructive Type Theory*. PhD thesis, Saarland University, 2022.

26 Dominik Kirst and Marc Hermes. Synthetic undecidability and incompleteness of first-order axiom systems in Coq. In *International Conference on Interactive Theorem Proving*. LIPIcs, 2021. doi:10.4230/LIPIcs.ITP.2021.23.

27 Dominik Kirst, Johannes Hostert, Andrej Dudenhefner, Yannick Forster, Marc Hermes, Mark Koch, Dominique Larchey-Wendling, Niklas Mück, Benjamin Peters, Gert Smolka, and Dominik Wehr. A Coq library for mechanised first-order logic. In *Coq Workshop*, 2022.

28 Dominik Kirst and Dominique Larchey-Wendling. Trakhtenbrot's theorem in Coq: Finite model theory through the constructive lens. *Logical Methods in Computer Science*, 18, 2022. doi:10.46298/lmcs-18(2:17)2022.

29 Dominik Kirst and Benjamin Peters. Gödel's theorem without tears: Essential incompleteness in synthetic computability. In *Annual conference of the European Association for Computer Science Logic*. LIPIcs, 2023. doi:10.4230/LIPIcs.CSL.2023.30.

30 Dieter Klemke. Ein Henkin-Beweis für die Vollständigkeit eines Kalküls relativ zur Grzegorczyk-Semantik. *Archiv für mathematische Logik und Grundlagenforschung*, 14:148–161, 1971.

31 Marcus Kracht. *Tools and Techniques in Modal Logic*. Elsevier, 1999.

32 E.G.K. López-Escobar. On intuitionistic sentential connectives I. *Revista Colombiana de Matemáticas*, 19(1-2):117–130, January 1985.

33 Marco Maggesi and Cosimo Perini Brogi. A Formal Proof of Modal Completeness for Provability Logic. In Liron Cohen and Cezary Kaliszyk, editors, *12th International Conference on Interactive Theorem Proving (ITP 2021)*, volume 193 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITP.2021.26.

34 C. Moisil. Logique modale. *Disquisitiones malhematicae et physicae*, 2:3–98, 1942.

35 Grigory K Olkhovikov and Guillermo Badia. Craig interpolation theorem fails in bi-intuitionistic predicate logic. *The Review of Symbolic Logic*, 17(2):611–633, 2024. doi:10.1017/s1755020322000296.

36 Hiroakira Ono. Craig's Interpolation Theorem for the Intuitionistic Logic and Its Extensions: A Semantical Approach. *Studia Logica: An International Journal for Symbolic Logic*, 45(1):19–33, 1986. doi:10.1007/BF01881546.

37 Christine Paulin-Mohring. Inductive definitions in the system coq rules and properties. In *International Conference on Typed Lambda Calculi and Applications*, pages 328–345. Springer, 1993. doi:10.1007/BFb0037116.

38 Luís Pinto and Tarmo Uustalu. Proof search and counter-model construction for bi-intuitionistic propositional logic with labelled sequents. In M. Giese and A. Waaler, editors, *Proc. TABLEAUX*, pages 295–309, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-02716-1_22.

39 Arthur N. Prior. *Time and Modality*. Greenwood Press, Westport, Conn., 1955.

40 Arthur N. Prior. *Past, Present and Future*. Clarendon P., Oxford,, 1967.

41 Arthur N. Prior. *Papers on Time and Tense*. Oxford University Press UK, Oxford, England, 1968.

42 Cecylia Rauszer. A Formalization of the Propositional Calculus of H-B Logic. *Studia Logica: An International Journal for Symbolic Logic*, 33(1):23–34, 1974.

43 Cecylia Rauszer. Semi-Boolean algebras and their application to intuitionistic logic with dual operations. *Fundamenta Mathematicae LXXXIII*, pages 219–249, 1974.

44 Cecylia Rauszer. On the Strong Semantical Completeness of Any Extension of the Intuitionistic Predicate Calculus. *Bulletin de l'Académie Polonaise des Sciences*, XXIV(2):81–87, 1976.

45 Cecylia Rauszer. An algebraic approach to the Heyting-Brouwer predicate calculus. *Fundamenta Mathematicae*, 96(2):127–135, 1977.

46 Cecylia Rauszer. Applications of Kripke Models to Heyting-Brouwer Logic. *Studia Logica: An International Journal for Symbolic Logic*, 36(1/2):61–71, 1977.

47 Cecylia Rauszer. Model Theory for an Extension of Intuitionistic Logic. *Studia Logica*, 36(1-2):73–87, 1977.

48 Cecylia Rauszer. *An Algebraic and Kripke-Style Approach to a Certain Extension of Intuitionistic Logic*. PhD thesis, Instytut Matematyczny Polskiej Akademi Nauk, 1980.

49 Katsuhiko Sano and John G. Stell. Strong completeness and the finite model property for bi-intuitionistic stable tense logics. In *Proc. Methods for Modalities 2017*, volume 243 of *Electronic Proceedings in Theoretical Computer Science*, pages 105–121. Open Publishing Association, 2017. doi:10.4204/EPTCS.243.8.

50 Ian Shillito. *New Foundations for the Proof Theory of Bi-Intuitionistic and Provability Logics Mechanized in Coq*. PhD thesis, Australian National University, Canberra, 2023.

51 Ian Shillito and Dominik Kirst. A mechanised and constructive reverse analysis of soundness and completeness of bi-intuitionistic logic. In Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, pages 218–229. ACM, 2024. doi:10.1145/3636501.3636957.

52 Kathrin Stark, Steven Schäfer, and Jonas Kaiser. Autosubst 2: reasoning with multi-sorted de bruijn terms and vector substitutions. In *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 166–180, 2019. doi:10.1145/3293880.3294101.

53 The Coq Development Team. The coq proof assistant, June 2023. doi:10.5281/zenodo.8161141.

54 Heinrich Wansing. Falsification, natural deduction and bi-intuitionistic logic. *J. Log. Comput.*, 26(1):425–450, 2016. doi:10.1093/logcom/ext035.

55 Heinrich Wansing. Connexive Logic. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2023 edition, 2023.

56 Benjamin Werner. Sets in types, types in sets. In *International Symposium on Theoretical Aspects of Computer Software*, pages 530–546. Springer, 1997. doi:10.1007/BFb0014566.

A Appendix

Proof of Truth lemma 17. By induction on ψ, only listing the missing cases:

- $\psi := P(t_0, \dots, t_{|P|})$: we have $P(t_0, \dots, t_{|P|}) \in \Gamma$ iff $(t_0, \dots, t_{|P|}) \in \mathcal{P}^c(\Gamma, P)$ by definition of the canonical model. The latter is equivalent to $\mathcal{M}^c, \Gamma, \alpha^c \Vdash P(t_0, \dots, t_{|P|})$ by definition and the fact that terms are interpreted as themselves via α^c.
- $\psi = \dot{\bot}$: we have that $\dot{\bot} \notin \Gamma$ by consistency. We also have $\mathcal{M}^c, \Gamma, \alpha^c \nVdash \dot{\bot}$ by definition. So, we trivially have $\dot{\bot} \in \Gamma$ iff $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \dot{\bot}$.
- $\psi = \varphi \mathbin{\dot{\wedge}} \chi$: we have that $\varphi \mathbin{\dot{\wedge}} \chi \in \Gamma$ iff $\varphi \in \Gamma$ and $\chi \in \Gamma$ via deductive closure. By induction hypothesis this holds if and only if $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi$ and $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \chi$. Then $\varphi \mathbin{\dot{\wedge}} \chi \in \Gamma$ iff $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \mathbin{\dot{\wedge}} \chi$.
- $\psi = \varphi \mathbin{\dot{\vee}} \chi$: we have that $\varphi \mathbin{\dot{\vee}} \chi \in \Gamma$ iff $[\varphi \in \Gamma$ or $\chi \in \Gamma]$ by primeness and deductive closure. By induction hypothesis this holds if and only if $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi$ or $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \chi$. Then $\varphi \mathbin{\dot{\vee}} \chi \in \Gamma$ iff $\mathcal{M}^c, \Gamma, \alpha^c \Vdash \varphi \mathbin{\dot{\vee}} \chi$. ◀

Taking Bi-Intuitionistic Logic First-Order: A Proof-Theoretic Investigation via Polytree Sequents

Tim S. Lyon
Technische Universität Dresden, Germany

Ian Shillito
The Australian National University, Canberra, Ngunnawal & Ngambri Country, Australia

Alwen Tiu
The Australian National University, Canberra, Ngunnawal & Ngambri Country, Australia

Abstract

It is well-known that extending the Hilbert axiomatic system for first-order intuitionistic logic with an exclusion operator, that is dual to implication, collapses the domains of models into a constant domain. This makes it an interesting problem to find a sound and complete proof system for first-order bi-intuitionistic logic with non-constant domains that is also conservative over first-order intuitionistic logic. We solve this problem by presenting the first sound and complete proof system for first-order bi-intuitionistic logic with increasing domains. We formalize our proof system as a polytree sequent calculus (a notational variant of nested sequents), and prove that it enjoys cut-elimination and is conservative over first-order intuitionistic logic. A key feature of our calculus is an explicit eigenvariable context, which allows us to control precisely the scope of free variables in a polytree structure. Semantically this context can be seen as encoding a notion of Scott's existence predicate for intuitionistic logic. This turns out to be crucial to avoid the collapse of domains and to prove the completeness of our proof system. The explicit consideration of the variable context in a formula sheds light on a previously overlooked dependency between the residuation principle and the existence predicate in the first-order setting, which may help to explain the difficulty in designing a sound and complete proof system for first-order bi-intuitionistic logic.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Modal and temporal logics; Theory of computation → Constructive mathematics; Theory of computation → Automated reasoning

Keywords and phrases Bi-intuitionistic, Cut-elimination, Conservativity, Domain, First-order, Polytree, Proof theory, Reachability, Sequent

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.41

Related Version *Extended Version*: `https://arxiv.org/abs/2404.15855` [26]

Funding *Tim S. Lyon*: European Research Council, Consolidator Grant DeciGUT (771779)

1 Introduction

Propositional bi-intuitionistic logic (BIP), also referred to as *Heyting-Brouwer logic* [33], is a conservative extension of propositional intuitionistic logic (IP), obtained by adding the binary connective $\multimap$ (referred to as *exclusion*)[1] among the traditional intuitionistic connectives. This logic has proven relevant in computer science, having a formulae-as-types interpretation in terms of first-class coroutines [7] and where modal extensions have found import in image processing [38]. While in intuitionistic logic the connectives $\wedge$ and $\rightarrow$ form a residuated pair,

[1] Also referred to as *pseudo-difference* [33], *subtraction*, and *co-implication* [13].

i.e. $(\varphi \wedge \psi) \to \chi$ is valid *iff* $\varphi \to (\psi \to \chi)$ is valid *iff* $\psi \to (\varphi \to \chi)$ is valid, in bi-intuitionistic logic the connectives $\vee$ and $\multimap$ also form a residuated pair, i.e. $\varphi \to (\psi \vee \chi)$ is valid *iff* $(\varphi \multimap \psi) \to \chi$ is valid *iff* $(\varphi \multimap \chi) \to \psi$ is valid.[2] To put it succinctly, BIP is a bi-intuitionistic extension of IP that is (1) conservative and (2) has the residuation property, i.e. $(\wedge, \to)$ and $(\multimap, \vee)$ form residuated pairs.

When extending first-order intuitionistic logic (IQ) to its bi-intuitionistic counterpart, a "natural" axiomatization seems to be one obtained by adding the universal axioms (Ax1) $\forall x\varphi \to \varphi(t/x)$, (Ax2) $\forall x(\psi \to \varphi) \to (\psi \to \forall x\varphi)$ (where x is not free in ψ), and the rule (Gen) $\varphi / \forall x\varphi$ to the axioms of BIP. This extension, which we refer to as the logic $\mathsf{BIQ}(\mathcal{CD})$, turns out *not* to be conservative over first-order intuitionistic logic IQ, as it allows one to prove the *quantifier shift axiom* $\forall x(\varphi \vee \psi) \to \forall x\varphi \vee \psi$ (where x is not free in ψ), which is not valid intuitionistically. A proof of the quantifier shift axiom is given below, where MP stands for modus ponens, Res stands for the residuation property described above, and $\delta := \forall x((\forall x(\varphi \vee \psi) \multimap \psi) \to \varphi) \to ((\forall x(\varphi \vee \psi) \multimap \psi) \to \forall x\varphi)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\forall x(\varphi \vee \psi) \to (\varphi \vee \psi)}\,\text{Ax1}}{(\forall x(\varphi \vee \psi) \multimap \psi) \to \varphi}\,\text{Res}}{\forall x((\forall x(\varphi \vee \psi) \multimap \psi) \to \varphi)}\,\text{Gen} \quad \dfrac{}{\delta}\,\text{Ax2}}{(\forall x(\varphi \vee \psi) \multimap \psi) \to \forall x\varphi}\,\text{MP}}{\forall x(\varphi \vee \psi) \to \forall x\varphi \vee \psi}\,\text{Res}$$

It is well-known that the quantifier shift axiom characterizes the class of first-order intuitionistic Kripke models with *constant domains* [10, 16], thus forcing the models for $\mathsf{BIQ}(\mathcal{CD})$ to satisfy this constraint. Indeed, various works in the literature (e.g., [32, 34]) have shown that completeness for $\mathsf{BIQ}(\mathcal{CD})$ requires the domain to be constant. These works and the above example strongly suggest that it might not be possible to have a proof system for a bi-intuitionistic logic with non-constant domains, at least not as a traditional Hilbert system. As far as we know, there is no prior successful attempt at solving this problem.

In this paper, we provide the first sound and complete proof system for first-order bi-intuitionistic logic with increasing domains, which we refer to here as $\mathsf{BIQ}(\mathcal{ID})$. With some minor modifications, the proof system for $\mathsf{BIQ}(\mathcal{ID})$ can be converted into a proof system for $\mathsf{BIQ}(\mathcal{CD})$. A key insight in avoiding the collapse of domains in $\mathsf{BIQ}(\mathcal{ID})$ is to consider the universal quantifier as implicitly carrying an assumption about the existence of the quantified variable. Proof theoretically, this could be done by introducing a notion of an *existence predicate*, first studied by Scott [35]. An existence predicate such as $E(x)$ postulates that x exists in the domain under consideration. By insisting that all universally quantified variables be guarded by an existence predicate, i.e. universally quantified formulae would have the form $\forall x(E(x) \to \varphi(x))$, the quantifier shift axiom can be rewritten as: $\forall x(E(x) \to (\varphi \vee \psi)) \to (\forall x(E(x) \to \varphi) \vee \psi)$. Attempting a bottom-up construction of a derivation similar to our earlier example for this rewritten axiom, we get stuck at the the top-most residuation rule, which is in fact not a valid instance of Res:

$$\dfrac{\dfrac{\dfrac{\dfrac{E(x) \to [\forall x(E(x) \to (\varphi \vee \psi)] \to (\varphi \vee \psi)}{E(x) \to [(\forall x(E(x) \to (\varphi \vee \psi)) \multimap \psi] \to \varphi}\,\text{Res}}{\forall x(E(x) \to [\forall x(E(x) \to (\varphi \vee \psi)) \multimap \psi] \to \varphi)}\,\text{Gen} \quad \cdots}{[\forall x(E(x) \to (\varphi \vee \psi)) \multimap \psi] \to \forall x(E(x) \to \varphi)}\,\text{MP}}{\forall x(E(x) \to (\varphi \vee \psi)) \to (\forall x(E(x) \to \varphi) \vee \psi)}\,\text{Res}$$

[2] However, they are not logically equivalent, e.g., $[\varphi \to (\psi \vee \chi)] \to [(\varphi \multimap \psi) \to \chi]$ is not valid.

For the proof construction to proceed, we would have to somehow discharge the assumption $E(x)$ in the premise of Gen before applying the residuation rule. In the logic of constant domains $\mathsf{BIQ}(\mathcal{CD})$, $E(x)$ is equivalent to $\top$ (i.e. the interpretation of any term in the logic is an object that exists in all worlds in the underlying Kripke model). So the version of the quantifier shift axiom with the existence predicate is provably equivalent to the original one in $\mathsf{BIQ}(\mathcal{CD})$. This is not the case, however, in the logic of increasing domains $\mathsf{BIQ}(\mathcal{ID})$, since the assumption $E(x)$ cannot always be discharged. What this example highlights is that a typical proof-theoretical argument used to show the provability of the quantifier shift axiom (and hence the collapse of domains) implicitly depends on an existence assumption on objects in the domains in the underlying Kripke model. What we show here is that by making this dependency explicit and by carefully managing the use of the existence assumptions in proofs, we are able to obtain a sound and complete proof system for $\mathsf{BIQ}(\mathcal{ID})$.

One issue with the existence predicate is that it is not clear how it should interact with the exclusion operator. Semantically, a formula like $\forall x[E(x) \to ((p(x) \multimap \exists y(E(y) \wedge p(y))) \to \bot)]$ asserts that, if an object x exists in the current domain, then postulating that $p(x)$ holds in a predecessor world should imply that x exists as well in that predecessor world. This is valid in our semantics, but it was not at all obvious how a proof system that admits this tautology, and does not also degenerate into a logic with constant domains, should be designed. We shall come back to this example later in Section 3. Additionally, the existence predicate poses a problem when proving conservativity over first-order intuitionistic logic that does not feature this predicate. We overcome this remaining hurdle by enriching sequents with an explicit variable context, which can be seen as essentially encoding the existence predicate, while avoiding introducing it explicitly in the language of formulae.

The proof systems for $\mathsf{BIQ}(\mathcal{ID})$ and $\mathsf{BIQ}(\mathcal{CD})$ are both formalized using *polytree sequents* [5], which are connected binary graphs whose vertices are traditional Gentzen sequents and which are free of (un)directed cycles. Polytree sequents are a restriction of traditional labeled sequents [37, 41] and are notational variants of nested sequents [3, 18, 2]. (NB. For details on the relationship between polytree and nested sequents, see [5].) Nested sequents were introduced independently by Bull [3] and Kashima [18] and employ trees of Gentzen sequents in proofs. Both polytree sequents and nested sequents allow for simple formulations of proof systems for various non-classical logics that enjoy important proof theoretical properties such as cut-elimination and subformula properties. Such systems have also found a range of applications, being used in knowledge integration algorithms [24], serving as a basis for constructive interpolation and decidability techniques [21, 25, 40], and even being used to solve open questions about axiomatizability [17]. We make use of polytree sequents in our work as they admit a formula interpretation (at least in the intuitionistic case), which can be leveraged for direct translations of proofs into sequent calculus or Hilbert calculus proofs.

The calculi for $\mathsf{BIQ}(\mathcal{ID})$ and $\mathsf{BIQ}(\mathcal{CD})$ are based on these richly structured sequents, which internalize the existence predicate into syntactic components, called *domain atoms*, present in each node of the sequent. The rich structure of these sequents is exploited by special rules within our calculi called *reachability rules*, which traverse paths in a polytree sequent, propagating and/or consuming data. We demonstrate that our calculi enjoy the height-preserving invertibility of every rule, and show that a wide range of novel and useful structural rules are height-preserving admissible, culminating in a non-trivial proof of cut-elimination.

Outline of Paper. In Section 2, we define a semantics for first-order bi-intuitionistic logic with increasing domains $\mathsf{BIQ}(\mathcal{ID})$ and constant domains $\mathsf{BIQ}(\mathcal{CD})$. In Section 3, we define our polytree sequent calculi showing them sound and complete relative to the provided

semantics. In Section 4, we establish admissibility and invertibility results as well as prove a non-trivial cut-elimination theorem. We conclude and discuss future work in Section 5. Due to space constraints, most proofs have been deferred to the online appended version [26].

2 Logical Preliminaries

In this section, we introduce the language, models, and semantics for first-order bi-intuitionistic logic with increasing domains, dubbed $\mathsf{BIQ}(\mathcal{ID})$, and with constant domains, dubbed $\mathsf{BIQ}(\mathcal{CD})$. Let $\mathrm{Var} := \{x, y, z, \ldots\}$ be a countably infinite set of *variables* and $\mathrm{Fun} = \{f, g, h, \ldots\}$ be a countably infinite set of *function symbols* containing countably many function symbols of each arity $n \in \mathbb{N}$. We let $ar(f) = n$ denote that the arity of the function symbol f is n and let $a, b, c, \ldots$ denote *constants*, which are function symbols of arity 0. For a set $X \subseteq \mathrm{Var}$, we define the set $\mathrm{Ter}(X)$ of *X-terms* to be the smallest set satisfying the following two constraints: (1) $X \subseteq \mathrm{Ter}(X)$, and (2) if $f \in \mathrm{Fun}$, f is of arity n, and $t_1, \ldots, t_n \in \mathrm{Ter}(X)$, then $f(t_1, \ldots, t_n) \in \mathrm{Ter}(X)$. The complete set of terms Ter is defined to be $\mathrm{Ter}(\mathrm{Var})$. We use t, s, ... (potentially annotated) to denote (X-)terms and let $VT(t)$ denote the set of variables occurring in the term t. We will often write a list $t_1, \ldots, t_n$ of terms as $\vec{t}$, and define $VT(\vec{t}) = VT(t_1) \cup \cdots \cup VT(t_n)$.

We let $\mathrm{Pred} := \{p, q, \ldots\}$ be a countably infinite set of predicates containing countably many predicates of each arity $n \in \mathbb{N}$. We denote the arity of a predicate p as $ar(p)$ and refer to predicates of arity 0 as *propositional atoms*. An *atomic formula* is a formula of the form $p(t_1, \ldots, t_n)$, obtained by prefixing a predicate p of arity $ar(p) = n$ to a tuple of terms of length n. We will often write atomic formulae $p(t_1, \ldots, t_n)$ as $p(\vec{t})$.

▶ **Definition 1** (The Language $\mathcal{L}$). *The* language $\mathcal{L}$ *is defined to be the set of formulae generated via the following grammar in Backus-Naur form:*

$$\varphi ::= p(\vec{t}) \mid \bot \mid \top \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \multimap \varphi \mid \varphi \to \varphi \mid \exists x \varphi \mid \forall x \varphi$$

where p ranges over Pred*, the terms $\vec{t} = t_1, \ldots, t_n$ range over* Ter*, and x ranges over the set* Var*. We use φ, ψ, χ, ... to denote formulae.*

The occurrence of a variable x in φ is defined to be *free* given that x does not occur within the scope of a quantifier binding x. We let $FV(\varphi)$ denote the set of all free variables occurring in the formula φ and use $\varphi(x_1, \ldots, x_n)$ to denote that $FV(\varphi) = \{x_1, \ldots, x_n\}$. We let $\varphi(t/x)$ denote the formula obtained by replacing each free occurrence of the variable x in φ by t, potentially renaming bound variables to avoid unwanted variable capture; e.g. $(\forall y p(x, y))(y/x) = \forall z p(y, z)$. The *complexity* of a formula φ, written $|\varphi|$, is recursively defined as follows: (1) $|p(t_1, \ldots, t_n)| = |\bot| = |\top| := 0$, (2) $|Qx\varphi| := |\varphi| + 1$ for $Q \in \{\forall, \exists\}$, and (3) $|\varphi \circ \psi| := |\varphi| + |\psi| + 1$ for $\circ \in \{\vee, \wedge, \to, \multimap\}$.

Following [32], we give a Kripke-style semantics for $\mathsf{BIQ}(\mathcal{ID})$, defining the models used first, and explaining how formulae are evaluated over them second.

▶ **Definition 2** (ID-Frame). *An* ID-frame *(or,* frame*) is a tuple $F = (W, \leq, U, D)$ such that:*

- *W is a non-empty set $\{w, u, v, \ldots\}$ of* worlds*;*
- *$\leq \subseteq W \times W$ is a reflexive and transitive binary relation;*
- *U is a non-empty set referred to as the* universe*;*
- *$D : W \to \mathcal{P}(U)$ is a* domain function *mapping each $w \in W$ to a non-empty set $D(w) \subseteq U$ with $U = \bigcup_{w \in W} D(w)$, which satisfies the* increasing domain condition*:* (ID) *If $w \leq u$, then $D(w) \subseteq D(u)$.*

▶ **Definition 3** (ID-Model). *We define an* ID-Model *(or,* model*) M to be an ordered triple (F, I_1, I_2) such that:*

- *$F = (W, \leq, U, D)$ is a frame;*
- *I_1 is a function interpreting each function symbol $f \in$ Fun such that $ar(f) = n$ by a function $I_1(f) : U^n \to U$, satisfying two conditions:* (C_1) *For each $w \in W$ and constant a, $I_1(a) \in D(w)$, and* (C_2) *For each $w \in W$, $\vec{a} \in D(w)^n$ iff $I_1(f)(\vec{a}) \in D(w)$.*
- *I_2 is a function interpreting, in each $w \in W$, each predicate $p \in$ Pred such that $ar(p) = n$ by a set $I_2(w, p) \subseteq D(w)^n$, satisfying the following* monotonicity condition*:* (M) *If $w \leq u$, then $I_2(w, p) \subseteq I_2(u, p)$.*

▶ **Definition 4** (M-assignment). *Let $M = (F, I_1, I_2)$ be a model. We define an* M-assignment *to be a function $\alpha : \text{Var} \to U$. We note $\alpha[a/x]$ is the function α modified in x such that $\alpha[a/x](x) = a$ and $\alpha[a/x](y) = \alpha(y)$ if $y \neq x$. Given an M-assignment α, we define the interpretation of t in M given α, denoted $\overline{\alpha}(t)$, inductively as follows: $\overline{\alpha}(x) := \alpha(x)$ and $\overline{\alpha}(f(t_1, ..., t_n)) := I_1(f)(\overline{\alpha}(t_1), ..., \overline{\alpha}(t_n))$.*

▶ **Definition 5** (Semantics). *Let $M = (W, \leq, U, D, I_1, I_2)$ be a model with $w \in W$ and α an M-assignment. The satisfaction relation $\Vdash$ is defined as follows:*

- *$M, w, \alpha \Vdash p(t_1, \ldots, t_n)$ iff $(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_n)) \in I_2(w, p)$;*
- *$M, w, \alpha \nVdash \bot$;*
- *$M, w, \alpha \Vdash \top$;*
- *$M, w, \alpha \Vdash \varphi \vee \psi$ iff $M, w, \alpha \Vdash \varphi$ or $M, w, \alpha \Vdash \psi$;*
- *$M, w, \alpha \Vdash \varphi \wedge \psi$ iff $M, w, \alpha \Vdash \varphi$ and $M, w, \alpha \Vdash \psi$;*
- *$M, w, \alpha \Vdash \varphi \multimap \psi$ iff there exists a $u \in W$ such that $u \leq w$, $M, u, \alpha \Vdash \varphi$, and $M, u, \alpha \nVdash \psi$;*
- *$M, w, \alpha \Vdash \varphi \to \psi$ iff for all $u \in W$, if $w \leq u$ and $M, u, \alpha \Vdash \varphi$, then $M, u, \alpha \Vdash \psi$;*
- *$M, w, \alpha \Vdash \exists x \varphi$ iff there exists an $a \in D(w)$ such that $M, w, \alpha[a/x] \Vdash \varphi$;*
- *$M, w, \alpha \Vdash \forall x \varphi$ iff for all $u \in W$ and all $a \in D(u)$, if $w \leq u$, then $M, u, \alpha[a/x] \Vdash \varphi$.*

For a set $\Gamma \subseteq \mathcal{L}$ of formulae, we write $\Gamma \Vdash \varphi$ iff for all models M, M-assignments α, and worlds w in M, if $M, w, \alpha \Vdash \psi$ for each $\psi \in \Gamma$, then $M, w, \alpha \Vdash \varphi$. A formula φ is valid *iff $\emptyset \Vdash \varphi$. Finally, we define the logic $\mathsf{BIQ}(\mathcal{ID})$ to be the set $\{\varphi \mid \emptyset \Vdash \varphi\}$ of all valid formulae.*

Note that here we define logics as *sets of formulae*, and not *consequence relations*. While this is fit for our purpose, the reader should be warned that historical confusions emerged around this distinction in the case of propositional bi-intuitionistic logic [15, 36], notably pertaining to the deduction theorem.

▶ **Proposition 6.** *Let $M = (W, \leq, U, D, I_1, I_2)$ be a model with α an M-assignment. For any $\varphi \in \mathcal{L}$, if $M, w, \alpha \Vdash \varphi$ and $w \leq u$, then $M, u, \alpha \Vdash \varphi$.*

▶ Remark 7. We define a CD-*model* to be a model satisfying the *constant domain condition*: (CD) If $w, u \in W$, then $D(w) = D(u)$. If we impose the (CD) condition on models, then first-order bi-intuitionistic logic with *constant domains*, dubbed $\mathsf{BIQ}(\mathcal{CD})$, can be defined as the set of all valid formulae over the class of CD-models. In what follows, we let $\mathcal{ID}$ denote the class of ID-models and $\mathcal{CD}$ denote the class of CD-models.

▶ **Example 8.** Consider the formula $\forall x((p(x) \multimap \exists y p(y)) \to \bot)$, discussed in the introduction, but with the existence predicate removed. In the semantics with increasing domains, this formula is valid. To see this, suppose otherwise, i.e. that there exists a world w where the formula is false. Thus, there is a successor $w \leq u$ such that $\bar{\alpha}(x) \in D(u)$ and $p(x) \multimap \exists y p(y)$ is true, for some assignment α. The latter implies that for some u' such that $u' \leq u$, $p(x)$ is true (i.e. $\alpha(x) \in I_P(u', p)$), but $\exists y p(y)$ is false. The former implies that $\alpha(x) \in D(u')$, so by the semantic clause for the $\exists$ quantifier, $\exists y p(y)$ must be true – contradiction.

3 Polytree Sequent Systems

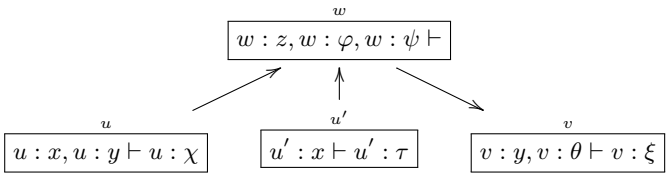
Let $\mathrm{Lab} = \{w, u, v, \ldots\}$ be a countably infinite set of labels. For a formula $\varphi \in \mathcal{L}$ and label $w \in \mathrm{Lab}$, we define $w : \varphi$ to be a *labeled formula*. We use Γ, Δ, Σ, ... to denote finite multisets of labeled formulae, and let $w{:}\Gamma$ denote a multiset of labeled formulae all labeled with w. A *relational atom* is an expression of the form wRu and a *domain atom* is an expression of the form $w : x$, where $w, u \in \mathrm{Lab}$ and $x \in \mathrm{Var}$. Intuitively, the domain atom formalizes an existence predicate: $w : x$ can be interpreted as saying that the interpretation of x exists at world w. We use $\mathcal{R}$ and $\mathcal{T}$ (and annotated versions thereof) to denote finite multisets of, respectively, relational atoms and domain atoms. Also, we define $w{:}VT(t) = w : x_1, \ldots, w : x_n$ with $VT(t) = \{x_1, \ldots, x_n\}$, define $w{:}VT(\vec{t}) = w : VT(t_1), \ldots, w : VT(t_n)$ with $\vec{t} = t_1, \ldots, t_n$, and let $w : \vec{x} = w : x_1, \ldots, w : x_n$ for $\vec{x} = x_1, \ldots, x_n$. For multisets X and Y of labeled formulae, relational atoms, and/or domain atoms, we let X, Y denote the multiset union of X and Y, and $\mathrm{Lab}(X)$ denote the set of labels occurring in X.

▶ **Definition 9** (Polytree Sequent). *We define a* polytree sequent *to be an expression of the form* $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ *such that (1) if* $\mathcal{R} \neq \emptyset$, *then* $\mathrm{Lab}(\mathcal{T}, \Gamma, \Delta) \subseteq \mathrm{Lab}(\mathcal{R})$ *and if* $\mathcal{R} = \emptyset$, *then* $|\mathrm{Lab}(\mathcal{T}, \Gamma, \Delta)| = 1$, *and (2)* $\mathcal{R}$ *forms a* polytree, *i.e. the graph* $G = (V, E)$ *such that* $V = \mathrm{Lab}(\mathcal{R})$ *and* $E = \{(w, u) \mid wRu \in \mathcal{R}\}$ *is connected and free of both directed and undirected cycles. We refer to* $\mathcal{R}, \mathcal{T}, \Gamma$ *as the* antecedent *and* Δ *as the* consequent *of a polytree sequent. We will often refer to* polytree sequents *more simply as* sequents.

We sometimes use S, S_0, S_1, ... to denote sequents, and for $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$, we define $\mathrm{Lab}(S) = \mathrm{Lab}(\mathcal{R}, \mathcal{T}, \Gamma, \Delta)$. A *flat sequent* is an expression of the form $\mathcal{T}, \Gamma \vdash \Delta$ such that $|\mathrm{Lab}(\mathcal{T}, \Gamma, \Delta)| = 1$, i.e. all labeled formulae and domain atoms share the same label. Polytree sequents encode certain binary graphs whose nodes are flat sequents and such that if you ignore the orientation of the edges, the graph is a tree (cf. [5]). For example, the sequent

$$S = \underbrace{u'Rw, uRw, wRv}_{\mathcal{R}}, \underbrace{u' : x, u : x, u : y, w : z, v : y}_{\mathcal{T}}, \underbrace{w : \varphi, w : \psi, v : \theta}_{\Gamma} \vdash \underbrace{u' : \tau, u : \chi, v : \xi}_{\Delta}$$

can be graphically depicted as the polytree $pt(S)$, shown below:



▶ Remark 10. To simplify the proofs of our results in Section 4, we assume w.l.o.g. that sequents with isomorphic polytree representations are mutually derivable from one another.

3.1 Semantics and Proof Systems

The following definition specifies how to interpret sequents. In essence, we lift the semantics of $\mathcal{L}$ to sequents by means of "M-interpretations", mapping sequents into models.

▶ **Definition 11** (Sequent Semantics). *Let* $M = (W, \leq, U, D, I_1, I_2)$ *be a model and* α *an* M*-assignment. We define an* M-interpretation *to be a function* ι *mapping every label* $w \in \mathrm{Lab}$ *to a world* $\iota(w) \in W$. *The* satisfaction *of multisets* $\mathcal{R}$, $\mathcal{T}$, *and* Γ *are defined accordingly:*

- $M, \iota, \alpha \models \mathcal{R}$ *iff for all* $wRu \in \mathcal{R}$, $\iota(w) \leq \iota(u)$;
- $M, \iota, \alpha \models \mathcal{T}$ *iff for all* $w : x \in \mathcal{T}$, $\overline{\alpha}(x) \in D(\iota(w))$;
- $M, \iota, \alpha \models \Gamma$ *iff for all* $w : \varphi \in \Gamma$, $M, \iota(w), \alpha \Vdash \varphi$.

$$\frac{}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}p(\vec{t}) \vdash \Delta, u{:}p(\vec{t})}\ (ax)^{\dagger_1} \qquad \frac{}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\bot \vdash \Delta}\ (\bot\mathrm{L}) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi, w{:}\psi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi \vee \psi}\ (\vee\mathrm{R})$$

$$\frac{}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\top}\ (\top\mathrm{R}) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi, w{:}\psi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \wedge \psi \vdash \Delta}\ (\wedge\mathrm{L}) \qquad \frac{\mathcal{R}, \mathcal{T}, w{:}y, \Gamma, w{:}\varphi(y/x) \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\exists x\varphi \vdash \Delta}\ (\exists\mathrm{L})^{\dagger_2}$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \vdash \Delta \qquad \mathcal{R}, \mathcal{T}, \Gamma, w{:}\psi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \vee \psi \vdash \Delta}\ (\vee\mathrm{L}) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi \qquad \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\psi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi \wedge \psi}\ (\wedge\mathrm{R})$$

$$\frac{\mathcal{R}, uRw, \mathcal{T}, \Gamma, u{:}\varphi \vdash \Delta, u{:}\psi}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \prec \psi \vdash \Delta}\ (\prec\mathrm{L})^{\dagger_3} \qquad \frac{\mathcal{R}, wRu, \mathcal{T}, \Gamma, u{:}\varphi \vdash \Delta, u{:}\psi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi \to \psi}\ (\to\mathrm{R})^{\dagger_3}$$

$$\frac{\mathcal{R}, \mathcal{T}, w{:}VT(\vec{t}), \Gamma, w{:}p(\vec{t}) \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}p(\vec{t}) \vdash \Delta}\ (ds) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\exists x\varphi, w{:}\varphi(t/x)}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\exists x\varphi}\ (\exists\mathrm{R})^{\dagger_4}$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \to \psi \vdash \Delta, u{:}\varphi \qquad \mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \to \psi, u{:}\psi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \to \psi \vdash \Delta}\ (\to\mathrm{L})^{\dagger_1}$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, u{:}\varphi \prec \psi, w{:}\varphi \qquad \mathcal{R}, \mathcal{T}, \Gamma, w{:}\psi \vdash \Delta, u{:}\varphi \prec \psi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, u{:}\varphi \prec \psi}\ (\prec\mathrm{R})^{\dagger_1}$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\forall x\varphi, u{:}\varphi(t/x) \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\forall x\varphi \vdash \Delta}\ (\forall\mathrm{L})^{\dagger_5} \qquad \frac{\mathcal{R}, wRu, \mathcal{T}, u{:}y, \Gamma \vdash \Delta, u{:}\varphi(y/x)}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\forall x\varphi}\ (\forall\mathrm{R})^{\dagger_6}$$

Side Conditions:

$\dagger_1 := w \twoheadrightarrow^*_{\mathcal{R}} u$ $\quad \dagger_3 := u$ is fresh $\quad \dagger_5 := w \twoheadrightarrow^*_{\mathcal{R}} u$ and $\mathsf{A}(t, X_u, \mathcal{R}, \mathcal{T})$

$\dagger_2 := y$ is fresh $\quad \dagger_4 := \mathsf{A}(t, X_w, \mathcal{R}, \mathcal{T}) \quad \dagger_6 := u$ and y are fresh

Figure 1 The System $\mathsf{LBIQ}(\mathcal{ID})$.

We define a sequent $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ to be satisfied *on M with ι and α, written $M, \iota, \alpha \models S$, iff if $M, \iota, \alpha \models \mathcal{R}$, and $M, \iota, \alpha \models \mathcal{T}$, as well as $M, \iota, \alpha \models \Gamma$, then there exists a $w : \psi \in \Delta$ such that $M, \iota, \alpha \models w : \psi$. We write $M, \iota, \alpha \not\models S$ when a sequent S is* not satisfied *on M with ι and α. A sequent S is defined to be valid iff for every model M, every M-interpretation ι, and every M-assignment α, we have $M, \iota, \alpha \models S$; otherwise, we say that S is invalid and write $M, \iota, \alpha \not\models S$.*

Given a sequent $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$, we define the *term substitution* $S(t/x)$ to be the sequent obtained by replacing (1) every labeled formula $w : \varphi$ in Γ, Δ by $w : \varphi(t/x)$ and (2) $\mathcal{T}$ by $\mathcal{T}(t/x) := (\mathcal{T} \setminus \{w{:}x \mid w{:}x \in \mathcal{T}\}) \cup \{w{:}y \mid w{:}x \in \mathcal{T} \text{ and } y \in VT(t)\}$. For example, if $S = wRu, w{:}x, u{:}x, u{:}y, w{:}p(x) \vdash u{:}\forall y q(x, y)$, then

$$S(f(y,z)/x) = wRu, w{:}y, w{:}z, u{:}y, u{:}z, u{:}y, w{:}p(f(y,z)) \vdash u{:}\forall x' q(f(y,z), x')$$

where the bound variable y in $\forall y q(x, y)$ was renamed to x' to avoid capture. We now define two *reachability relations* $\twoheadrightarrow^+_{\mathcal{R}}$ and $\twoheadrightarrow^*_{\mathcal{R}}$ as well as the notion of *availability* [9, 22] – all of which are required to properly formulate certain inference rules in our calculi.

▶ **Definition 12** ($\twoheadrightarrow^+_{\mathcal{R}}$, $\twoheadrightarrow^*_{\mathcal{R}}$). *Let $\mathcal{R}$ be a finite multiset of relational atoms such that $w, u \in \mathrm{Lab}(\mathcal{R})$. We say that u is* strictly reachable *from w, written $w \twoheadrightarrow^+_{\mathcal{R}} u$, iff there exist $v_1, \ldots, v_n \in \mathrm{Lab}(\mathcal{R})$ such that $wRv_1, \ldots, v_nRu \in \mathcal{R}$ with $n \in \mathbb{N}$. We say that u is* reachable *from w, written $w \twoheadrightarrow^*_{\mathcal{R}} u$, iff $w \twoheadrightarrow^+_{\mathcal{R}} u$ or $w = u$. We write $w \not\twoheadrightarrow^*_{\mathcal{R}} u$ if $w \twoheadrightarrow^*_{\mathcal{R}} u$ does not hold.*

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathcal{R}, w' : x, u : \forall x(p \vee r(x)), u : p, v : q \vdash u : p, w' : r(x)}\,(ax)}{\pi \quad \mathcal{R}, w' : x, u : \forall x(p \vee r(x)), u : r(x), v : q \vdash u : p, w' : r(x)}\,(ax)}{wRu, uRv, vRw', w' : x, u : \forall x(p \vee r(x)), u : p \vee r(x), v : q \vdash u : p, w' : r(x)}\,(\vee\mathrm{L})}{wRu, uRv, vRw', w' : x, u : \forall x(p \vee r(x)), v : q \vdash u : p, w' : r(x)}\,(\forall\mathrm{L})}{wRu, uRv, u : \forall x(p \vee r(x)), v : q \vdash u : p, v : \forall x r(x)}\,(\forall\mathrm{R})}{wRu, u : \forall x(p \vee r(x)) \vdash u : p, u : q \to \forall x r(x)}\,(\to\mathrm{R})}{wRu, u : \forall x(p \vee r(x)) \vdash u : p \vee (q \to \forall x r(x))}\,(\vee\mathrm{R})}{\vdash w : \forall x(p \vee r(x)) \to (p \vee (q \to \forall x r(x)))}\,(\to\mathrm{R})
$$

Figure 2 An example proof in $\mathsf{LBIQ}(\mathcal{CD})$ for bi-intuitionistic logic with constant domains.

▶ **Definition 13** (Available). *Let $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ be a sequent with $w \in \mathrm{Lab}(S)$. We define a term t to be* available *for w in $\mathcal{R}, \mathcal{T}$, written $\mathsf{A}(t, X_w, \mathcal{R}, \mathcal{T})$, iff $t \in \mathrm{Ter}(X_w)$ such that*

$$X_w = \{x \mid u{:}x \in \mathcal{T} \text{ and } u \twoheadrightarrow^*_{\mathcal{R}} w \text{ for some } u \in \mathrm{Lab}(S)\}.$$

Our polytree calculus $\mathsf{LBIQ}(\mathcal{ID})$ for $\mathsf{BIQ}(\mathcal{ID})$ is shown in Figure 1. The (ax), $(\bot\mathrm{L})$, and $(\top\mathrm{R})$ rules serve as *initial rules*, the *domain shift rule* (ds) encodes the fact that $I_2(p, w) \subseteq D(w)^n$ in any model. We define the *principal formula* in an inference rule to be the one explicitly mentioned in the conclusion, the *auxiliary formulae* to be the non-principal formulae explicitly mentioned in the premises, and an *active formula* to be either a principal or auxiliary formula. For example, $w : \exists x\varphi$ is principal, $w : \varphi(t/x)$ is auxiliary, and both are active in $(\exists\mathrm{R})$. Note that all rules of our calculus preserve the property of being a polytree-structured sequent. We define a *proof* and its *height* as usual [39]. Two unique features of our calculi are the inclusion of *reachability rules* and the *domain shift rule* (ds), which we elaborate on next.

3.2 Reachability Rules

A unique feature of our calculi is the inclusion of *reachability rules* (introduced in [20]), a generalization of *propagation rules* (cf. [4, 8, 14]), which are not only permitted to propagate formulae throughout a polytree sequent when applied bottom-up, but may also check to see if data exists along certain paths. The rules (ax), $(\to\mathrm{L})$, $(\prec\mathrm{R})$, $(\exists\mathrm{R})$, and $(\forall\mathrm{L})$ serve as our reachability rules. The side conditions of our reachability rules are listed at the bottom of Figure 1. Moreover, we define a label u or a variable y to be *fresh* in a rule application (as in the $(\exists\mathrm{L})$ and $(\forall\mathrm{R})$ rules) *iff* it does not occur in the conclusion of the rule.

▶ Remark 14. If we set $\dagger_4 :=$ "$t \in \mathrm{Ter}$", $\dagger_5 :=$ "$w \twoheadrightarrow^*_{\mathcal{R}} u$ and $t \in \mathrm{Ter}$", and remove the (ds) rule, then we obtain a polytree calculus, dubbed $\mathsf{LBIQ}(\mathcal{CD})$, for the constant domain version of the logic $\mathsf{BIQ}(\mathcal{CD})$. We also note that in the constant domain setting, domain atoms are unnecessary and can be omitted from sequents.

To provide intuition, we give an example showing the operation of a reachability rule.

▶ **Example 15.** Let $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ such that $\mathcal{R} = uRw, wRv$, $\mathcal{T} = w : x, u : y, v : z$, $\Gamma = w : \forall x p(x), w : p(f(y)), w : p(z)$, and $\Delta = u : q(x) \prec q(x), v : r(y)$. A representation of S as a polytree is shown below. We explain (in)valid applications of the $(\forall\mathrm{L})$ reachability rule.

$$
\boxed{u : y \vdash \overset{u}{u : q(x) \prec q(x)}} \longrightarrow \boxed{w : x, w : \forall x p(x), w : p(f(y)), w : p(z) \vdash \overset{w}{}} \longrightarrow \boxed{v : z \vdash \overset{v}{v : r(y)}}
$$

The term $f(y)$ is available for w in S since $u \twoheadrightarrow^*_{\mathcal{R}} w$, namely there is an edge from u to w, and $f(y) \in \mathrm{Ter}(X_w)$ since $X_w = \{x, y\}$. Therefore, we may (top-down) apply the ($\forall$L) rule to delete $w : p(f(y))$ and derive the sequent $S' = \mathcal{R}, \mathcal{T}, \Gamma' \vdash \Delta$ with $\Gamma' = w : \forall x p(x), w : p(z)$. By contrast, $w : p(z)$ cannot be deleted via an application of ($\forall$L) because the term z is *not* available for w in S (observe that w is not reachable from v) meaning $z \notin \mathrm{Ter}(X_w)$.

▶ Remark 16. We note that for any set $X \subseteq \mathrm{Var}$, $\mathrm{Ter}(X) \neq \emptyset$ since all constants are contained in $\mathrm{Ter}(X)$ by definition. This means that bottom-up applications of ($\exists$R) and ($\forall$L) may instantiate existential and universal formulae with any constant.

The reachability rules (ax), ($\to$L) and ($\multimap$R) are important to ensure completeness for both $\mathsf{LBIQ}(\mathcal{CD})$ and $\mathsf{LBIQ}(\mathcal{ID})$. The reachability rules for ($\exists$R) and ($\forall$L) are relevant only for $\mathsf{LBIQ}(\mathcal{ID})$ to ensure that the domains in the model do not collapse into a constant domain. We illustrate the importance of these reachability rules with a couple of examples.

▶ **Example 17** (An Intuitionistic Formula Valid in Constant Domain Models)**.** Consider the intuitionistic formula $\forall x(p \vee r(x)) \to (p \vee (q \to \forall x r(x)))$. This formula was adapted from an example in [27], which was used to illustrate the difficulty of obtaining a sound and complete sequent system for intuitionistic logic with constant domains. A proof of this formula in $\mathsf{LBIQ}(\mathcal{CD})$ is shown in Figure 2 and crucially relies on reachability rules. In the figure, the relational atoms $\mathcal{R} = wRu, uRv, vRw'$ in the instances of (ax) allow us to conclude that $u \twoheadrightarrow^*_{\mathcal{R}} u$ and $u \twoheadrightarrow^*_{\mathcal{R}} w'$, justifying the left and right instances of (ax), respectively.

▶ **Example 18** (Non-Provability of the Quantifier Shift Axiom in the Increasing Domain Setting)**.** Let us consider again the quantifier shift axiom $\forall x(\varphi \vee \psi) \to (\forall x \varphi \vee \psi)$ and an attempt to construct a proof (bottom-up) of one of its instances in $\mathsf{LBIQ}(\mathcal{ID})$.

$$\dfrac{\dfrac{\dfrac{wRu, uRv, v : x, u : \forall x(p(x) \vee q) \vdash v : p(x), u : q}{wRu, u : \forall x(p(x) \vee q) \vdash u : \forall x p(x), u : q}\,(\forall\mathrm{R})}{wRu, u : \forall x(p(x) \vee q) \vdash u : \forall x p(x) \vee q}\,(\vee\mathrm{R})}{\vdash w : \forall x(p(x) \vee q) \to (\forall x p(x) \vee q)}\,(\to\mathrm{R})$$

It is obvious that to finish this proof, we would need to instantiate the $\forall x$ quantifier in the labeled formula $u : \forall x(p(x) \vee q)$ with x by applying the ($\forall$L) rule. However, to do so, we would need to demonstrate that the world u is reachable from v where the domain atom $v : x$ resides. Yet, u is not reachable from v, so x is not available at u to be used by ($\forall$L).

3.3 The Domain Shift Rule (ds)

Although the reachability rules for the quantifiers prevent the quantifier shift axiom from being proved, it turns out that they are not sufficient to ensure the completeness of $\mathsf{LBIQ}(\mathcal{ID})$ with respect to the sequent semantics for the logic $\mathsf{BIQ}(\mathcal{ID})$. Interestingly, this incompleteness only arises when the exclusion connective is involved – if one considers the intuitionistic fragment of $\mathsf{LBIQ}(\mathcal{ID})$, these reachability rules are sufficient to prove completeness (see Lemma 26 in Section 3.5). To see this incompleteness issue, consider the formula in Example 8, which is semantically valid, and the following attempt at a (bottom-up) construction of a proof:

$$\dfrac{\dfrac{\dfrac{wRu, uRv, u'Rv, u : x, u' : p(x) \vdash u' : \exists y p(y), v : \bot}{wRu, uRv, u : x, v : p(x) \multimap \exists y p(y) \vdash v : \bot}\,(\multimap\mathrm{L})}{wRu, u : x \vdash u : (p(x) \multimap \exists y p(y)) \to \bot}\,(\to\mathrm{R})}{\vdash w : \forall x((p(x) \multimap \exists y p(y)) \to \bot)}\,(\forall\mathrm{R})$$

We have so far applied only invertible rules, so the original sequent is provable *iff* the top sequent in the above derivation also is. To proceed with the proof construction, one needs to instantiate the existential quantifier $\exists y$ with x. However, the only domain atom containing x is located at the world u, which is not available to u' where the existential formula is located.

It is not so obvious how the reachability rules for quantifiers could be amended to allow this example to be proved. Looking at the above derivation, it might be tempting to augment the calculus with a rule that allows a backward reachability condition for domain atoms, e.g., making $u : x$ available to u' for when $u' \twoheadrightarrow^*_{\mathcal{R}} u$ under certain admissibility conditions, but this could easily lead to a collapse of the domains if one is not careful. Instead, our approach here is motivated by the semantic clause for predicates: when $p(x)$ holds in a world, its interpretation requires that x is also defined in that world. Proof theoretically, we could think of this as postulating an axiom such as $\forall x(p(x) \to E(x))$ where $E(x)$ is an existence predicate (which, as we recall, was behind the semantics of the domain atoms). Translated into our calculus, this gives us the (ds) rule as shown in Figure 1. Using the (ds) rule, the above derivation can now be completed to a proof:

$$\dfrac{\dfrac{\dfrac{}{\mathcal{R}, u : x, u' : x, u' : p(x) \vdash u' : p(x), u' : \exists y p(y), v : \bot}\,(ax)}{\mathcal{R}, u : x, u' : x, u' : p(x) \vdash u' : \exists y p(y), v : \bot}\,(\exists \mathrm{R})}{wRu, uRv, u'Rv, u : x, u' : p(x) \vdash u' : \exists y p(y), v : \bot}\,(ds)$$

Note that the (ds) rule can only be applied to atomic predicates, but not arbitrary formulae, which rules out unsound instances. It may be possible to relax the restriction to atomic predicates by imposing some positivity conditions on the occurrences of x, but we did not find this necessary – neither for completeness, nor for cut-elimination.

▶ Remark 19. The (ds) rule can be removed without affecting the cut-elimination result for $\mathsf{LBIQ}(\mathcal{ID})$. This raises the possibility of defining a first-order bi-intuitionistic logic strictly weaker than $\mathsf{BIQ}(\mathcal{ID})$. It is unclear what the semantics for such a logic would look like.

3.4 Soundness and Completeness

▶ **Theorem 20** (Soundness). *Let S be a sequent. If S is provable in $\mathsf{LBIQ}(\mathcal{ID})$ ($\mathsf{LBIQ}(\mathcal{CD})$), then S is (CD-)valid.*

Proof. By induction on the height of the given proof; see Appendix A for details. ◀

The completeness of our polytree calculi (see Theorem 22 below) is shown by taking a sequent of the form $w : \vec{x} \vdash w : \varphi(\vec{x})$ as input and showing that if the sequent is not provable, then the calculus can be used to construct an infinite derivation from which a counter-model of the end sequent can be extracted. We note that completeness only holds relative to sequents of the form $w : \vec{x} \vdash w : \varphi(\vec{x})$, which includes a domain atom for each free variable in $\varphi(\vec{x})$. This restriction is needed because quantifier rules can only (bottom-up) instantiate quantified formulae with the free variables $\vec{x}$ of $\varphi(\vec{x})$ if such free variables occur as domain atoms, and such free variables must be accessible to quantifier rules to properly extract a counter-model of the end sequent (see [23] for a relevant discussion).

Below, we outline the cut-free completeness proof for $\mathsf{LBIQ}(\mathcal{ID})$ as the proof for $\mathsf{LBIQ}(\mathcal{CD})$ is similar; the complete proof can be found in the online, appended version [26]. Our proof outline makes use of various new notions, which we now define. A *pseudo-derivation* is defined to be a (potentially infinite) tree whose nodes are sequents and where every parent node corresponds to the conclusion of a rule in $\mathsf{LBIQ}(\mathcal{ID})$ with the children nodes corresponding to the premises. We remark that a proof in $\mathsf{LBIQ}(\mathcal{ID})$ is a finite pseudo-derivation where all

top sequents are instances of (ax), $(\bot\text{L})$, or $(\top\text{R})$. A *branch* $\mathcal{B}$ is defined to be a maximal path of sequents through a pseudo-derivation, starting from the conclusion. The following lemma is useful in proving completeness.

▶ **Lemma 21.** *Let $\mathcal{C} \in \{\mathcal{ID}, \mathcal{CD}\}$. For each $i \in \{0,1,2\}$, let $S_i = \mathcal{R}_i, \mathcal{T}_i, \Gamma_i \vdash \Delta_i$ be a sequent.*

1. *If $w \twoheadrightarrow^*_{\mathcal{R}} u$ holds for the conclusion of a rule (r) in $\mathsf{LBIQ}(\mathcal{C})$, then $w \twoheadrightarrow^*_{\mathcal{R}} u$ holds for the premises of (r);*
2. *If $w : p(\vec{t}) \in \Gamma_0, \Delta_0$ and S_0 is the conclusion of a rule (r) in $\mathsf{LBIQ}(\mathcal{C})$ with S_1 (and S_2) the premise(s) of (r), then $w : p(\vec{t}) \in \Gamma_1, \Delta_1$ (and $w : p(\vec{t}) \in \Gamma_2, \Delta_2$, resp.);*
3. *If $w : x \in \mathcal{T}_0$ and S_0 is the conclusion of a rule (r) in $\mathsf{LBIQ}(\mathcal{C})$ with S_1 (and S_2) the premise(s) of (r), then $w : x \in \mathcal{T}_1$ (and $w : x \in \mathcal{T}_2$, resp.).*

The lemma tells us that propagation paths, the position of atomic formulae, and the position of terms are bottom-up preserved in rule applications.

▶ **Theorem 22** (Completeness). *If $w : \vec{x} \vdash w : \varphi(\vec{x})$ is (*CD-*)valid, then $w : \vec{x} \vdash w : \varphi(\vec{x})$ is provable in $\mathsf{LBIQ}(\mathcal{ID})$ ($\mathsf{LBIQ}(\mathcal{CD})$).*

Proof (Outline). We assume that $S = w : \vec{x} \vdash w : \varphi(\vec{x})$ is not provable in $\mathsf{LBIQ}(\mathcal{ID})$ and show that a model M can be defined which witnesses that S is invalid. To prove this, we first define a proof-search procedure `Prove` that bottom-up applies rules from $\mathsf{LBIQ}(\mathcal{ID})$ to $w : \vec{x} \vdash w : \varphi(\vec{x})$. Second, we show how a model M can be extracted from failed proof-search. We now describe the proof-search procedure `Prove` and let $\prec$ be a well-founded, strict linear order over the set Ter of terms.

`Prove`. Let us take $w : \vec{x} \vdash w : \varphi(\vec{x})$ as input and continue to the next step. We show some key steps; the complete `Prove` procedure can be found in the online appended version [26].

(ax), $(\bot\text{L})$, and $(\top\text{R})$. Suppose $\mathcal{B}_1, \ldots, \mathcal{B}_n$ are all branches occurring in the current pseudo-derivation and let $S_1, \ldots, S_n$ be the top sequents of each respective branch. For each $1 \leq i \leq n$, we halt the computation of `Prove` on each branch $\mathcal{B}_i$ where S_i is of the form (ax), $(\bot\text{L})$, or $(\top\text{R})$. If `Prove` is halted on each branch $\mathcal{B}_i$, then `Prove` returns `True` because a proof of the input has been constructed. However, if `Prove` did not halt on each branch $\mathcal{B}_i$ with $1 \leq i \leq n$, then let $\mathcal{B}_{j_1}, \ldots, \mathcal{B}_{j_k}$ be the remaining branches for which `Prove` did not halt. For each such branch, copy the top sequent above itself, and continue to the next step.

(ds). Suppose $\mathcal{B}_1, \ldots, \mathcal{B}_n$ are all branches occurring in the current pseudo-derivation and let $S_1, \ldots, S_n$ be the top sequents of each respective branch. For each $1 \leq i \leq n$, we consider $\mathcal{B}_i$ and extend the branch with bottom-up applications of (ds) rules. Let $\mathcal{B}_{k+1}$ be the current branch under consideration, and assume that $\mathcal{B}_1, \ldots, \mathcal{B}_k$ have already been considered. We assume that the top sequent in $\mathcal{B}_{k+1}$ is of the form

$$S_{k+1} = \mathcal{R}, \mathcal{T}, \Gamma, w : p_1(\vec{t}_1), \ldots, w_\ell : p_\ell(\vec{t}_\ell) \vdash \Delta$$

where all atomic input formulae are displayed in S_{k+1} above. We successively consider each atomic input formula and bottom-up apply (ds), yielding a branch extending $\mathcal{B}_{k+1}$ with a top sequent saturated under (ds) applications. After these operations have been performed for each branch $\mathcal{B}_i$ with $1 \leq i \leq n$, we continue to the next step.

(∃L). Suppose $\mathcal{B}_1, \ldots, \mathcal{B}_n$ are all branches occurring in the current pseudo-derivation and let $S_1, \ldots, S_n$ be the top sequents of each respective branch. For each $1 \leq i \leq n$, we consider $\mathcal{B}_i$ and extend the branch with bottom-up applications of (∃L) rules. Let $\mathcal{B}_{k+1}$ be the current branch under consideration, and assume that $\mathcal{B}_1, \ldots, \mathcal{B}_k$ have already been considered. We assume that the top sequent in $\mathcal{B}_{k+1}$ is of the form

$$S_{k+1} = \mathcal{R}, \mathcal{T}, \Gamma, w_1 : \exists x_1 \varphi_1, \ldots, w_m : \exists x_m \varphi_m \vdash \Delta$$

where all existential input formulae $w_i : \exists x_i \varphi_i$ are displayed in S_{k+1} above. We consider each formula $w_i : \exists x_i \varphi_i$ in turn, and bottom-up apply the (∃L) rule. These rule applications extend $\mathcal{B}_{k+1}$ such that

$$\mathcal{R}, \mathcal{T}', \Gamma, w_1 : \varphi_1(y_1/x_1), \ldots, w_n : \varphi_m(y_m/x_m) \vdash \Delta$$

is now the top sequent of the branch with $y_1, \ldots, y_m$ fresh variables and $\mathcal{T}' = \mathcal{T}, w_1 : y_1, \ldots, w_m : y_m$. After these operations have been performed for each branch $\mathcal{B}_i$ with $1 \leq i \leq n$, we continue to the next step.

(∃R). Suppose $\mathcal{B}_1, \ldots, \mathcal{B}_n$ are all branches occurring in the current pseudo-derivation and let $S_1, \ldots, S_n$ be the top sequents of each respective branch. For each $1 \leq i \leq n$, we consider $\mathcal{B}_i$ and extend the branch with bottom-up applications of (∃R) rules. Let $\mathcal{B}_{k+1}$ be the current branch under consideration, and assume that $\mathcal{B}_1, \ldots, \mathcal{B}_k$ have already been considered. We assume that the top sequent in $\mathcal{B}_{k+1}$ is of the form

$$S_{k+1} = \mathcal{R}, \mathcal{T}, \Gamma \vdash w_1 : \exists x_1 \varphi_1, \ldots, w_m : \exists x_m \varphi_m, \Delta$$

where all existential formulae $w_i : \exists x_i \varphi_i$ are displayed in S_{k+1} above. We consider each labeled formula $w_m : \exists x_m \varphi_i$ in turn, and bottom-up apply the (∃R) rule. Let $w_{\ell+1} : \exists x_{\ell+1} \varphi_{\ell+1}$ be the current formula under consideration, and assume that $w_1 : \exists x_1 \varphi_1, \ldots, w_\ell : \exists x_\ell \varphi_\ell$ have already been considered. Recall that $\prec$ is a well-founded, strict linear order over the set Ter of terms. Choose the $\prec$-minimal term $t \in \mathrm{Ter}(X_{w_{\ell+1}})$ that has yet to be picked to instantiate $w_{\ell+1} : \exists x_{\ell+1} \varphi_{\ell+1}$ and bottom-up apply the (∃R) rule, thus adding $w_{\ell+1} : \varphi_{\ell+1}(t/x_{\ell+1})$. We perform these operations for each branch $\mathcal{B}_i$ with $1 \leq i \leq n$.

The remaining rules of $\mathsf{LBIQ}(\mathcal{ID})$ are processed in a similar fashion. The `Prove` procedure will saturate open branches of the pseudo-derivation that is under construction by repeatedly (bottom-up) applying rules from $\mathsf{LBIQ}(\mathcal{ID})$ in a roundabout fashion.

Next, we aim to show that if `Prove` does not return `True`, then a model M, M-interpretation ι, and M-assignment α can be defined such that $M, \iota, \alpha \not\models S$. If `Prove` halts, i.e. `Prove` returns `True`, then a proof of S may be obtained by "contracting" all redundant inferences from the "(ax), ($\bot$L), and ($\top$R)" step of `Prove`. Therefore, in this case, since a proof exists, we have obtained a contradiction to our assumption. As a consequence, we have that `Prove` does not halt, that is, `Prove` generates an infinite tree with finite branching. By König's lemma, an infinite branch must exist in this infinite tree, which we denote by $\mathcal{B}$. We define a model $M = (W, \leq, U, D, I_1, I_2)$ by means of this branch as follows: Let us define the following sets, all of which are obtained by taking the union of each multiset of relational atoms, domain atoms, antecedent labeled formulae, and consequent labeled formulae (resp.) occurring within a sequent in $\mathcal{B}$:

$$\mathcal{R}^{\mathcal{B}} = \bigcup_{(\mathcal{R},\mathcal{T},\Gamma\vdash\Delta)\in\mathcal{B}} \mathcal{R} \qquad \mathcal{T}^{\mathcal{B}} = \bigcup_{(\mathcal{R},\mathcal{T},\Gamma\vdash\Delta)\in\mathcal{B}} \mathcal{T} \qquad \Gamma^{\mathcal{B}} = \bigcup_{(\mathcal{R},\mathcal{T},\Gamma\vdash\Delta)\in\mathcal{B}} \Gamma \qquad \Delta^{\mathcal{B}} = \bigcup_{(\mathcal{R},\mathcal{T},\Gamma\vdash\Delta)\in\mathcal{B}} \Delta$$

We now define: (1) $u \in W$ *iff* $u \in \mathrm{Lab}(\mathcal{R}^{\mathcal{B}}, \mathcal{T}^{\mathcal{B}}, \Gamma^{\mathcal{B}}, \Delta^{\mathcal{B}})$, (2) $\leq = \{(u, v) \mid uRv \in \mathcal{R}\}^*$ where $*$ denotes the reflexive-transitive closure, (3) $t \in U$ *iff* there exists a label $u \in \mathrm{Lab}(\mathcal{R}^{\mathcal{B}}, \mathcal{T}^{\mathcal{B}}, \Gamma^{\mathcal{B}}, \Delta^{\mathcal{B}})$ such that $t \in \mathrm{Ter}(X_u)$, (4) $t \in D(u)$ *iff* $t \in \mathrm{Ter}(X_u)$, and (5) $(t_1, \ldots, t_n) \in I_2(u, p)$ *iff* $v, u \in \mathrm{Lab}(\mathcal{R}^{\mathcal{B}}, \mathcal{T}^{\mathcal{B}}, \Gamma^{\mathcal{B}}, \Delta^{\mathcal{B}})$, $v \twoheadrightarrow^*_{\mathcal{R}^{\mathcal{B}}} u$, and $v : p(t_1, \ldots, t_n) \in \Gamma^{\mathcal{B}}$.

It can be shown that M is indeed a model. Let us define α to be the M-assignment mapping every variable in U to itself and every variable in $\mathrm{Var} \setminus U$ arbitrarily. To finish the proof of completeness, we now argue the following by mutual induction on the complexity of the formula ψ: (1) if $u : \psi \in \Gamma^{\mathcal{B}}$, then $M, u, \alpha \Vdash \psi$, and (2) if $u : \psi \in \Delta^{\mathcal{B}}$, then $M, u, \alpha \nVdash \psi$. Let ι to be the M-interpretation such that $\iota(u) = u$ for $u \in W$ and $\iota(v) \in W$ for $v \notin W$. By the proof above, $M, \iota, \alpha \not\models w : \vec{x} \vdash w : \varphi(\vec{x})$, showing that if a sequent of the form $w : \vec{x} \vdash w : \varphi(\vec{x})$ is not provable in $\mathsf{LBIQ}(\mathcal{ID})$, then it is invalid, that is, every valid sequent of the form $w : \vec{x} \vdash w : \varphi(\vec{x})$ is provable in $\mathsf{LBIQ}(\mathcal{ID})$. ◀

▶ Remark 23. We remark that cut admissibility follows from the soundness of the (*cut*) rule (see Figure 3) and the completeness theorem above. However, this method of proof has two downsides: first, the restriction in the completeness theorem above implies that cut admissibility only holds for proofs with an end sequent of the form $w : \vec{x} \vdash w : \varphi(\vec{x})$. Second, this (semantic) method of proof does not define an algorithm showing how instances of (*cut*) can be permuted upward and eliminated from a given proof. In Section 4, we will prove that cut admissibility holds for all proofs and will provide such an algorithm (see Theorem 30).

3.5 Intuitionistic Subsystems

We end this section by discussing two subsystems of $\mathsf{LBIQ}(\mathcal{ID})$ and $\mathsf{LBIQ}(\mathcal{CD})$ arising from restricting the connectives to the intuitionistic fragment. In the former case, we obtain a proof system for the usual first-order intuitionistic logic (with non-constant domains) IQ, and in the latter, we obtain a proof system for intuitionistic logic with constant domains IQC.

▶ **Corollary 24** (Conservativity). *Let φ be an intuitionistic formula (i.e. a formula with no occurrences of $\multimap$). Then, φ is valid in* IQ *(*IQC*) iff $\vdash w : \varphi$ is provable in* $\mathsf{LBIQ}(\mathcal{ID})$ *(respectively,* $\mathsf{LBIQ}(\mathcal{CD})$*).*

The proof of Corollary 24 is straightforward from Definition 11. We show here a stronger *proof-theoretic* conservativity result: we can in fact extract a purely intuitionistic fragment out of $\mathsf{LBIQ}(\mathcal{ID})$, where every sequent in the fragment is interpretable in the semantics without the existence predicate. We prove this via syntactic means, by showing how we can translate intuitionistic proofs in $\mathsf{LBIQ}(\mathcal{ID})$ to proofs in Gentzen's LJ [11, 12]. A key idea is to first define a formula interpretation of a polytree sequent, and then show that every inference rule corresponds to a valid implication in LJ. We start by defining a notion of intuitionistic (polytree) sequent.

▶ **Definition 25.** *A sequent $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ is an* intuitionistic sequent *iff $\mathcal{R}$ is a tree rooted at node u such that*

- *every formula in S is an intuitionistic formula (i.e. it contains no occurrences of $\multimap$),*
- *for every labeled formula $w : \varphi$ in S and variable $x \in VT(\varphi)$, x is available for w, and*
- *if $w : x$ and $z : x$ are in $\mathcal{T}$, then $w = z$.*

By $\mathsf{NIQ}(\mathcal{ID})$ we denote the restriction to intuitionistic sequents of the proof system $\mathsf{LBIQ}(\mathcal{ID})$ without the (*ds*) rule. The next lemma states an important property of $\mathsf{LBIQ}(\mathcal{ID})$, called the *separation property*, which was first discussed in the context of tense logics [14].

▶ **Lemma 26** (Separation). *An intuitionistic sequent S is provable in* $\mathsf{NIQ}(\mathcal{ID})$ *iff it is provable in* $\mathsf{LBIQ}(\mathcal{ID})$.

Proof (Outline). One direction, from $\mathsf{NIQ}(\mathcal{ID})$ to $\mathsf{LBIQ}(\mathcal{ID})$ is trivial. For the other direction, suppose π is a proof of S in $\mathsf{LBIQ}(\mathcal{ID})$. By induction on the structure of π, it can be shown that there is a proof π' in $\mathsf{LBIQ}(\mathcal{ID})$ in which every sequent in π' is *almost* intuitionistic – it satisfies all the requirements in Definition 25 except possibly the last condition (due to the possible use of the (ds) rule). Then, from π' we can construct another proof π'' of S that does not use (ds), by showing that one can always permute the rule (ds) up until it disappears. Since all the rules of $\mathsf{LBIQ}(\mathcal{ID})$, other than (ds), preserve the property of being an intuitionistic sequent, it then follows that π'' is a proof in $\mathsf{NIQ}(\mathcal{ID})$. ◀

To translate a proof in $\mathsf{NIQ}(\mathcal{ID})$ to LJ, we need to interpret a polytree sequent as a formula. This turns out to be somewhat problematic, due to the difficulty in interpreting the scopes of domain atoms, when interpreting them as universally quantified variables. Fortunately, in the case of intuitionistic sequents, the scopes of such variables follow a straightforward lexical scoping (i.e. their scopes are over formulae in the subtrees). To define the translation, we first relax the requirement on the domain atoms in intuitionistic sequents: a *quasi-intuitionistic sequent* is defined as in Definition 25, except that in the second clause, x is either available for w, or it does not occur in $\mathcal{T}$. Obviously an intutionistic sequent is also a quasi-intuitionistic sequent. Given a quasi-intuitionistic sequent S and a label w, we write S_w to denote the quasi-intuitionistic sub-sequent of S that is rooted in w, i.e. the sequent obtained from S by removing any relational atoms, domain atoms, and labeled formulae that mention a world v not reachable from w. Given a multiset of labeled formulae Γ, we denote with Γ_u the labeled formulae in Γ that are labeled with u.

▶ **Definition 27.** *Let $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ be a quasi-intuitionistic sequent. We define its formula interpretation $F(S)$ recursively on the height of the sequent tree and suppose S is rooted at u.*

- *If S is a flat sequent, then $F(X) = \forall \vec{x}(\bigwedge \Gamma \to \bigvee \Delta)$ where $\vec{x}$ are all the variables in $\mathcal{T}$;*
- *otherwise, if u has n successors $w_1, \ldots, w_n$, then*

$$F(S) = \forall \vec{x}(\bigwedge \Gamma_u \to (\bigvee \Delta_u \vee F(S_{w_1}) \vee \cdots \vee F(S_{w_n}))).$$

The following proof-theoretic conservativity result can then be proved using a standard translation technique for relating nested sequents and traditional Gentzen sequent calculi [6].

▶ **Proposition 28.** *Let S be an intuitionistic sequent. S is provable in* $\mathsf{NIQ}(\mathcal{ID})$ *iff $F(S)$ is provable in* LJ.

Proof (Outline). The proof is tedious, but not difficult and follows a general strategy to translate nested sequent proofs (which, recall, are notational variants of polytree sequent proofs) to traditional sequent proofs (with cuts) from the literature, see e.g., the translation from nested sequent to traditional sequent proofs for full intuitionistic linear logic [6]. For every inference rule in $\mathsf{NIQ}(\mathcal{ID})$ of the form:

$$\frac{S_1 \quad \cdots \quad S_n}{S}$$

we show that the formula $F(S_1) \wedge \cdots \wedge F(S_n) \to F(S)$ is provable in LJ. Then, given any proof in $\mathsf{NIQ}(\mathcal{ID})$, we simulate every inference step with its corresponding implication, followed by a cut. ◀

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}, \mathcal{T}, w{:}x, \Gamma \vdash \Delta}\,(wv) \qquad \frac{\mathcal{R}, \mathcal{T}, w{:}x, u{:}x, \Gamma \vdash \Delta}{\mathcal{R}, \mathcal{T}, w{:}x, \Gamma \vdash \Delta}\,(id)^{\dagger_1} \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, \Sigma \vdash \Delta, \Pi}\,(iw)$$

$$\frac{}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \vdash \Delta, u{:}\varphi}\,(gax)^{\dagger_1} \qquad \frac{\mathcal{R}, wRv, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}, uRv, \mathcal{T}, \Gamma \vdash \Delta}\,(br_f)^{\dagger_2} \qquad \frac{\mathcal{R}, vRu, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}, vRw, \mathcal{T}, \Gamma \vdash \Delta}\,(br_b)^{\dagger_3}$$

$$\frac{\mathcal{R}, \mathcal{T}, w{:}x, \Gamma \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\,(cd) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi, w{:}\varphi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\varphi \vdash \Delta}\,(ctr_l) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi, w{:}\varphi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi}\,(ctr_r)$$

$$\frac{\mathcal{R}, wRu, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}(w/u), \mathcal{T}(w/u), \Gamma(w/u) \vdash \Delta(w/u)}\,(mrg) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi \qquad \mathcal{R}, \mathcal{T}, \Gamma, u{:}\varphi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\,(cut)^{\dagger_1}$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}{\mathcal{R}, \mathcal{T}(t/x), \Gamma(t/x) \vdash \Delta(t/x)}\,(t/x) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\bot}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\,(\bot\mathrm{R}) \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\top \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\,(\top\mathrm{L})$$

$$\frac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\Pi}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, u{:}\Pi}\,(lwr)^{\dagger_1} \qquad \frac{\mathcal{R}, \mathcal{T}, \Gamma, u{:}\Sigma \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, w{:}\Sigma \vdash \Delta}\,(lft)^{\dagger_1}$$

Side Conditions:

$\dagger_1 := w \twoheadrightarrow^*_{\mathcal{R}} u$
$\dagger_2 := w \twoheadrightarrow^*_{\mathcal{R}} u$ and $u \not\twoheadrightarrow^*_{\mathcal{R}} v$
$\dagger_3 := w \twoheadrightarrow^*_{\mathcal{R}} u$ and $w \not\twoheadrightarrow^*_{\mathcal{R}} v$

Figure 3 Admissible rules.

As far as we know, for intuitionistic logic with constant domains IQC, there is no formalization in the traditional Gentzen sequent calculus that admits cut-elimination. There is, however, a formalization in prefixed tableaux by Fitting [9], which happens to be a syntactic variant of the intuitionistic fragment of LBIQ($\mathcal{CD}$) (shown in [19]).

4 Cut-Elimination

In this section, we show that LBIQ($\mathcal{ID}$) and LBIQ($\mathcal{CD}$) satisfy a sizable number of favorable properties culminating in syntactic cut-elimination. We explain here some key steps; the full details are available in the online appended version [26].

LBIQ($\mathcal{ID}$) and LBIQ($\mathcal{CD}$) can be seen as first-order extensions of Postniece's deep-nested sequent calculus for bi-intuitionistic logic DBiInt [31, 13]. Cut-elimination for DBiInt [13] was proven in two stages. First, cut-elimination was proven for a "shallow" version of the nested sequent calculus LBiInt, which can be seen as a variant of a display calculus [1]. The cut-elimination proof for this shallow calculus follows from Belnap's generic cut-elimination for display calculi [1]. Second, cut-free proofs in the shallow calculus are shown to be translatable to proofs in the deep-nested calculus. We do not have the corresponding shallow versions of LBIQ($\mathcal{ID}$) and LBIQ($\mathcal{CD}$), so we cannot rely on Belnap's generic cut-elimination. It may be possible to define shallow versions of our calculi, and then follow the same methodology outlined in [13] to prove cut-elimination, but we find that a direct cut-elimination proof is simpler, e.g., it avoids the need for proving the admissibility of certain structural rules called the display postulates [1], which lets one transition from shallow to deep inference systems.

Since our polytree sequents are a restriction of ordinary labeled sequents, another possible approach to cut-elimination is to apply the methodology for labeled sequent calculi [28]. A main issue in adapting this methodology is ensuring that the proof transformations needed

in cut-elimination preserve the polytree structure of sequents. A key proof transformation in a typical cut-elimination proof for labeled calculi is label substitution: given a proof π_1 and labels u and w, one constructs another proof π_2 by replacing u with w everywhere in π_1 and adjusts the inference rules accordingly. This is typically needed in the reduction of a cut where the last rules in both branches of the cut apply to the cut formula, and where one of the rules introduces (reading the rule bottom up) a new label and a new relational atom (e.g., ($\to$R)). Such a substitution operation may not preserve polytree structures. Another notable difference between our calculi and traditional labeled calculi is the absence of structural rules manipulating relational atoms. These differences mean that cut-elimination techniques for labeled sequent calculi cannot be immediately applied in our setting.

Instead, our cut-elimination proof builds on an approach by Pinto and Uustalu [29, 30], which deals with a polytree sequent calculus for *propositional* bi-intuitionistic logic. We thus provide a series of proof transformations, culminating in the elimination of cuts, which shares similarities with their work in the propositional case and expands in the first-order direction. These transformations are captured in proofs of the admissibility of rules shown in Figure 3. We illustrate some key transformations and why they are needed, through an example of a cut where ($\to$L) and ($\to$R) are applied to the cut formula. The formal details are available in the proof of Theorem 30.

Suppose we have the instance of cut shown below left, where π_1 is shown below right and π_2 is shown below bottom with $w \twoheadrightarrow^*_{\mathcal{R}} u$.

$$\dfrac{\dfrac{\dfrac{\pi_1}{\mathcal{R}, \mathcal{T}, \Gamma \vdash w : \varphi \to \psi, \Delta} \quad \dfrac{\pi_2}{\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi \vdash \Delta}}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\, cut \quad \dfrac{\dfrac{\pi_1'}{\mathcal{R}, \mathcal{T}, wRw', \Gamma, w' : \varphi \vdash w' : \psi, \Delta}}{\mathcal{R}, \mathcal{T}, \Gamma \vdash w : \varphi \to \psi, \Delta}\,(\to\text{R})}{\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi \vdash \Delta}\,(\to\text{L})$$

$$\dfrac{\dfrac{\pi_3}{\mathcal{R}, \mathcal{T}, w : \varphi \to \psi, \Gamma \vdash \Delta, u : \varphi} \quad \dfrac{\pi_4}{\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi, u : \psi \vdash \Delta}}{\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi \vdash \Delta}\,(\to\text{L})$$

A typical cut reduction strategy would be to cut π_1 with π_3 and π_4 (both with cut formula $\varphi \to \psi$), producing the proofs π_5 and π_6 of $\mathcal{R}, \Gamma \vdash \Delta, u : \varphi$ and $\mathcal{R}, \Gamma, u : \psi \vdash \Delta$, respectively. Next, one would cut π_5 with π_1' (with cut formula φ), producing a proof π_7, and then cut π_7 with π_6 (with cut formula ψ). There are a couple of issues with this strategy: (1) the cut formulae in the last two instances of cut have mismatched labels, i.e., w' on one side and u on the other ; (2) the label w' and the relational atom wRw' are not present in the conclusions of π_5 and π_6, so the contexts of the premises of the cuts do not match.

To fix these issues, we first need to transform the proof π_1' into two proofs π_5 and π_6, shown below left and right, respectively. As shown in the cut-elimination proof, a transformation that we use in this case is one that is represented by the rule (iw). This ensures that the contexts match the contexts of the concluding sequents in π_3 and π_4.

$$\dfrac{\dfrac{\dfrac{\pi_1'}{\mathcal{R}, wRw', \mathcal{T}, \Gamma, w' : \varphi \vdash w' : \psi, \Delta}\,(\to\text{R})}{\mathcal{R}, \mathcal{T}, \Gamma \vdash w : \varphi \to \psi, \Delta}\,(iw)}{\mathcal{R}, \mathcal{T}, \Gamma \vdash w : \varphi \to \psi, u : \varphi, \Delta}\,(iw) \qquad \dfrac{\dfrac{\dfrac{\pi_1'}{\mathcal{R}, wRw', \mathcal{T}, \Gamma, w' : \varphi \vdash w' : \psi, \Delta}\,(\to\text{R})}{\mathcal{R}, \mathcal{T}, \Gamma \vdash w : \varphi \to \psi, \Delta}\,(iw)}{\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi, u : \psi \vdash \Delta}\,(iw)$$

We then cut π_3 and π_4 with π_5 and π_6, respectively, which yields proofs π_7 and π_8 of $\mathcal{R}, \Gamma \vdash u : \varphi, \Delta$ and $\mathcal{R}, \Gamma, u : \psi \vdash \Delta$, respectively. At this stage, we want to cut π_7 with π_1', and then cut the resulting proof with π_8. However, this cut cannot be performed until the label w' and its associated relational atom are removed from the conclusion of π_1'. Simply substituting u for w' may break the polytree shape of the sequent, e.g., if there is a v such that wRv and vRu are in $\mathcal{R}$, then replacing u for w' in wRw' would break the polytree

shape of the sequent. So the relational atoms in the sequent also need to be modified. A transformation that we use in this case is represented by the rule (br_f), which shifts the relational atom wRw' "forward" from w to u given that $w \twoheadrightarrow^*_{\mathcal{R}} u$. We also need another transformation to "merge" the label u with the label w', deleting the relational atom in the process, represented as the (mrg) rule.

$$\dfrac{\dfrac{\pi_7}{\mathcal{R}, \mathcal{T}, \Gamma \vdash u : \varphi, u : \psi, \Delta}\,(iw) \qquad \dfrac{\dfrac{\dfrac{\pi'_1}{\mathcal{R}, wRw', \mathcal{T}, \Gamma, w' : \varphi \vdash w' : \psi, \Delta}\,(br_f)}{\mathcal{R}, uRw', \mathcal{T}, \Gamma, w' : \varphi \vdash w' : \psi, \Delta}\,(mrg)}{\mathcal{R}, \mathcal{T}, \Gamma, u : \varphi \vdash u : \psi, \Delta}\,cut}{\mathcal{R}, \mathcal{T}, \Gamma \vdash u : \psi, \Delta}$$

If we cut the above proof with π_8, we obtain a proof of $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$. Here we gloss over the termination arguments, but the details are available in the proof of Theorem 30.

The above example illustrates one among several proof transformations needed in cut-elimination. These transformations make use of the auxiliary rules in Figure 3. The bulk of the cut-elimination proof consists of showing these rules (height-preserving/hp-) admissible and the rules of $\mathsf{LBIQ}(\mathcal{ID})$ and $\mathsf{LBIQ}(\mathcal{CD})$ height-preserving invertible (i.e., hp-invertible). (NB. We take the notions of (hp-)admissibility and (hp-)invertibility to be defined as usual.) All (hp-)admissible rules preserve the polytree structure of sequents, and with the exception of (gax), (cut), and (cd), all rules in Figure 3 are *hp-admissible* in *both* calculi. The (gax) and (cut) rules are strictly *admissible* in both calculi, while (cd) is hp-admissible in only $\mathsf{LBIQ}(\mathcal{CD})$ as the availability conditions are not imposed on rules, rendering domain atoms unnecessary (see Remark 14). We now discuss some of the most interesting rules of Figure 3.

Let us first explain the rules (br_f) and (br_b). For some labels w, u, and v, assume $w \twoheadrightarrow^*_{\mathcal{R}} u$ and $u \not\twoheadrightarrow^*_{\mathcal{R}} v$ for $\mathcal{R} := \mathcal{R}', wRv$. Then, we know that (1) u and v are on two different paths passing through w of the polytree generated from $\mathcal{R}$, and (2) there is no vertex between v and w since otherwise a cycle would be present in $\mathcal{R}$. In this scenario, the rule (br_f) (for *br*anch *f*orward) allows one to move the polytree "rooted" at v forward by connecting it to u instead of w as shown left in the below figure. The rule (br_b) has a similar functionality; for some labels w, u, and v, assume $w \twoheadrightarrow^*_{\mathcal{R}} u$ and $w \not\twoheadrightarrow^*_{\mathcal{R}} v$ for $\mathcal{R} := \mathcal{R}', vRu$. Then, the rule (br_b) (for *br*anch *b*ackward) lets one move the polytree "rooted" at v backward by connecting it to w instead of u as shown right in the below figure.

Figure 4 The left and right diagrams demonstrate the functionality of (br_f) and (br_b), respectively.

The (mrg) rule merges a label and its direct successor and corresponds to the rules *nodemergeD* and *nodemergeU* of Pinto and Uustalu [29]. The rule (id) reflects the redundancy of a variable labeled by two labels such that one is reachable by the other. Model-theoretically, this redundancy follows from the fact that if x is interpreted at w, and u is reachable from w (in a model), then x is interpreted at u as well, showing the domain atom $u : x$ superfluous in the premise. Note that when the labels w and u are identical, then the rule represents a contraction on domain atoms; as $w \twoheadrightarrow^*_{\mathcal{R}} w$ always holds, we have that identical domain atoms can always be contracted in sequents. The rules (lwr) and (lft) allow us to modify the labels of formulae in a sequent by looking at its underlying polytree. More precisely, reading (lwr) and (lft) bottom-up, if $w \twoheadrightarrow^*_{\mathcal{R}} u$ we can both *lower* the label of $u : \Pi$ on the right of the sequent to w, and *lift* the label of $w : \Sigma$ on the left of the sequent to u.

▶ **Lemma 29.** *All non-initial rules in* LBIQ($\mathcal{C}$) *are hp-invertible.*

Finally, we can prove the admissibility of the (*cut*) rule. As our proof proceeds via local transformations of proofs, the proof is constructive and yields a cut-elimination algorithm.

▶ **Theorem 30** (Cut-elimination)**.** *The* (*cut*) *rule is admissible in* LBIQ($\mathcal{C}$)*.*

Proof. We proceed by a primary induction (PIH) on the complexity of the cut formula, and a secondary induction (SIH) on the sum of the heights of the proofs of the premises of (*cut*). Assume that we have proofs of the following form, with $w \twoheadrightarrow^*_{\mathcal{R}} u$.

$$\frac{\pi_1}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi}\,(r_1) \qquad \frac{\pi_2}{\mathcal{R}, \mathcal{T}, \Gamma, u{:}\varphi \vdash \Delta}\,(r_2)$$

We argue that there is a proof of $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ by a case distinction on (r_1) and (r_2), the last rules applied in the above proofs. We focus on some interesting cases; the remaining cases can be found in the online appended version [26].

$(r_1) = (ax)$. Then $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w : \varphi$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v_0 : p(\vec{t}) \vdash \Delta_0, v_1 : p(\vec{t})$ where $v_0 \twoheadrightarrow^*_{\mathcal{R}} v_1$. If $v_1 : p(\vec{t})$ is $w : \varphi$, then we have that $\mathcal{R}, \mathcal{T}, \Gamma, u : \varphi \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v_0 : p(\vec{t}), u : p(\vec{t}) \vdash \Delta$ where $\Gamma = \Gamma_0, v_0 : p(\vec{t})$. Given that $v_0 \twoheadrightarrow^*_{\mathcal{R}} v_1$ and $v_1 \twoheadrightarrow^*_{\mathcal{R}} u$, we can apply the hp-admissibility of (lft) on the latter to obtain a proof of $\mathcal{R}, \mathcal{T}, \Gamma_0, v_0 : p(\vec{t}), v_0 : p(\vec{t}) \vdash \Delta$. Consequently, we obtain a proof of $\mathcal{R}, \mathcal{T}, \Gamma_0, v_0 : p(\vec{t}) \vdash \Delta$, i.e. of $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$, using the hp-admissibility of (ctr_l). If $v_1 : p(\vec{t})$ is not $w : \varphi$, then we have that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v_0 : p(\vec{t}) \vdash \Delta_0, v_1 : p(\vec{t})$ where $v_0 \twoheadrightarrow^*_{\mathcal{R}} v_1$. The latter is provable by the admissibility of (gax).

$(r_1) = (ds)$. Then $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v{:}p(\vec{t}) \vdash \Delta, w{:}\varphi$ and we have a proof of $\mathcal{R}, \mathcal{T}, v{:}VT(\vec{t}), \Gamma_0, v{:}p(\vec{t}) \vdash \Delta, w{:}\varphi$. Consequently, we know that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v : p(\vec{t}) \vdash \Delta$. We also have that $\mathcal{R}, \mathcal{T}, \Gamma, u : \varphi \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v{:}p(\vec{t}), u{:}\varphi \vdash \Delta$. We can repeatedly apply the hp-admissibility of (wv) on the proof of the latter to obtain a proof of $S := \mathcal{R}, \mathcal{T}, v{:}VT(\vec{t}), \Gamma_0, v{:}p(\vec{t}), u{:}\varphi \vdash \Delta$. Then, we proceed as follows:

$$\dfrac{\dfrac{\mathcal{R}, \mathcal{T}, v{:}VT(\vec{t}), \Gamma_0, v{:}p(\vec{t}) \vdash \Delta, w{:}\varphi \qquad S}{\mathcal{R}, \mathcal{T}, v{:}VT(\vec{t}), \Gamma_0, v{:}p(\vec{t}) \vdash \Delta}\,(ds)}{\mathcal{R}, \mathcal{T}, \Gamma_0, v{:}p(\vec{t}) \vdash \Delta}\ \text{SIH}$$

Note that the instance of SIH is justified because the sum of the heights of the proofs of the premises has decreased.

$(r_1) = (\prec\!\!\text{L})$. Then $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w{:}\varphi$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v{:}\psi \prec \chi \vdash \Delta, w{:}\varphi$ and we a have proof of $\mathcal{R}, v_0 R v, \mathcal{T}, \Gamma_0, v_0{:}\psi \vdash \Delta, w{:}\varphi, v_0{:}\chi$. Consequently, we know that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v : \psi \prec \chi \vdash \Delta$. We also have that $\mathcal{R}, \mathcal{T}, \Gamma, u : \varphi \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma_0, v : \psi \prec \chi, u : \varphi \vdash \Delta$. We apply Lemma 29 on the proof of the latter sequent to obtain a proof of $\mathcal{R}, v_0 R v, \mathcal{T}, \Gamma_0, v_0 : \psi, u : \varphi \vdash \Delta, v_0 : \chi$, which we call S. Thus, we proceed as shown below. Note that the instance of SIH is justified as the sum of the heights of the proofs of the premises is smaller than that of the original cut.

$$\dfrac{\dfrac{\mathcal{R}, v_0 R v, \mathcal{T}, \Gamma_0, v_0{:}\psi \vdash \Delta, w{:}\varphi, v_0{:}\chi \qquad S}{\mathcal{R}, v_0 R v, \mathcal{T}, \Gamma_0, v_0{:}\psi \vdash \Delta, v_0{:}\chi}\,(\prec\!\!\text{L})}{\mathcal{R}, \mathcal{T}, \Gamma_0, v{:}\psi \prec \chi \vdash \Delta}\ \text{SIH}$$

$(r_1) = (\exists\mathrm{R})$. Then there are two cases to consider: in the left premise $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w:\varphi$ of (cut), either (1) $w:\varphi$ is not the principal formula $v:\exists x\psi$, or (2) $w:\varphi$ is the principal formula. In case (1), we have a proof of $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta_1, v:\exists x\psi, v:\psi(t/x), w:\varphi$, which we call S, and $\mathcal{R}, \mathcal{T}, \Gamma, u:\varphi \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma, u:\varphi \vdash \Delta_1, v:\exists x\psi$. We proceed as follows.

$$\dfrac{\dfrac{S \qquad \dfrac{\mathcal{R}, \mathcal{T}, \Gamma, u:\varphi \vdash \Delta_1, v:\exists x\psi}{\mathcal{R}, \mathcal{T}, \Gamma, u:\varphi \vdash \Delta_1, v:\exists x\psi, v:\psi(t/x)}\ \text{Lem.29}}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta_1, v:\exists x\psi, v:\psi(t/x)}\ \text{SIH}}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta_1, v:\exists x\psi}\ (\exists\mathrm{R})$$

In case (2), we have proof a of $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, v:\exists x\psi, v:\psi(t/x)$, and $\mathcal{R}, \mathcal{T}, \Gamma, u:\varphi \vdash \Delta$ is of the form $\mathcal{R}, \mathcal{T}, \Gamma, u:\exists x\psi \vdash \Delta$. In this case, we need to consider the shape of (r_2). If $u:\exists x\psi$ is not principal in (r_2), then we apply the hp-invertibility of (r_2) (Lemma 29) to the left premise of (cut) and use SIH to cut the result with the premise of (r_2), applying (r_2) afterward to reach our goal. If $u:\exists x\psi$ is principal in (r_2), then the premise of (r_2) is of the shape $\mathcal{R}, \mathcal{T}, v:y, \Gamma, v:\psi(y/x) \vdash \Delta$ where y is fresh. Then, we proceed as follows where π is the first proof given and $x_0, \ldots, x_n$ are all the variables appearing in t.

$$\dfrac{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, v:\exists x\psi, v:\psi(t/x) \qquad \dfrac{\mathcal{R}, \mathcal{T}, \Gamma, u:\exists x\psi \vdash \Delta}{\mathcal{R}, \mathcal{T}, \Gamma, u:\exists x\psi \vdash \Delta, v:\psi(t/x)}\ (iw)}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, v:\psi(t/x)}\ \text{SIH}$$

$$\dfrac{\pi \qquad \dfrac{\dfrac{\mathcal{R}, \mathcal{T}, v:y, \Gamma, v:\psi(y/x) \vdash \Delta}{\mathcal{R}, \mathcal{T}, v:x_0, \ldots, v:x_n, \Gamma, v:\psi(t/x) \vdash \Delta}\ (t/y)}{\mathcal{R}, \mathcal{T}, \Gamma, v:\psi(t/x) \vdash \Delta}\ (id)}{\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta}\ \text{PIH}$$

Note that the step involving (id) is justified as t is available for v, meaning for each $x_i \in VT(t)$, there exists a domain atom $u_i : x_i$ such that $u_i \twoheadrightarrow^*_{\mathcal{R}} v$, showing (id) applicable. ◀

5 Concluding Remarks

Our analysis indicates that there may be two interesting and possibly distinct first-order extensions of bi-intuitionistic logic that may be worth exploring. The first is to consider a logic with decreasing domains, i.e., if $w \leq u$ then $D(u) \subseteq D(w)$ in the Kripke model. Semantically, this logic is easy to define, but its proof theory is not at all obvious. We are looking into the possibility of formalizing a notion of "non-existence predicate," which is dual to the existence predicate, suggested by Restall [34]. This non-existence predicate may play a similar (but dual) role to the existence predicate in $\mathsf{LBIQ}(\mathcal{ID})$. The other extension is motivated from a proof-theoretic perspective. As mentioned in Remark 19, it seems that one can obtain a subsystem of $\mathsf{LBIQ}(\mathcal{ID})$ without the domain-shift rule (ds) that satisfies cut-elimination. As discussed in Section 3, the (ds) rule is crucial to ensure the completeness of $\mathsf{BIQ}(\mathcal{ID})$ in the presence of the exclusion operator, and so, a natural question to ask is what the semantics of such a logic would look like.

References

1 Nuel D. Belnap. Display logic. *Journal of philosophical logic*, 11(4):375–417, 1982. doi:10.1007/BF00284976.

2 Kai Brünnler. Deep sequent systems for modal logic. *Archive for Mathematical Logic*, 48(6):551–577, 2009. doi:10.1007/s00153-009-0137-3.

3 Robert A. Bull. Cut elimination for propositional dynamic logic without *. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 38(2):85–100, 1992. doi:10.1002/MALQ.19920380107.

4 Marcos A. Castilho, Luis Farinas del Cerro, Olivier Gasquet, and Andreas Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3, 4):281–297, 1997. doi:10.3233/FI-1997-323404.

5 Agata Ciabattoni, Tim Lyon, Revantha Ramanayake, and Alwen Tiu. Display to labelled proofs and back again for tense logics. *ACM Transactions on Computational Logic*, 22(3):1–31, 2021. doi:10.1145/3460492.

6 Ranald Clouston, Jeremy E. Dawson, Rajeev Goré, and Alwen Tiu. Annotation-free sequent calculi for full intuitionistic linear logic - extended version. *CoRR*, abs/1307.0289, 2013. arXiv:1307.0289.

7 Tristan Crolard. A formulae-as-types interpretation of subtractive logic. *Journal of Logic and Computation*, 14(4):529–570, 2004. doi:10.1093/logcom/14.4.529.

8 Melvin Fitting. Tableau methods of proof for modal logics. *Notre Dame Journal of Formal Logic*, 13(2):237–247, 1972. doi:10.1305/NDJFL/1093894722.

9 Melvin Fitting. Nested sequents for intuitionistic logics. *Notre Dame Journal of Formal Logic*, 55(1):41–61, 2014. doi:10.1215/00294527-2377869.

10 D. Gabbay, V. Shehtman, and D. Skvortsov. *Quantification in Non-classical Logics*. Studies in Logic and Foundations of Mathematics. Elsevier, Amsterdam, London, 2009.

11 Gerhard Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

12 Gerhard Gentzen. Untersuchungen über das logische Schließen. II. *Mathematische Zeitschrift*, 39(1):405–431, 1935.

13 Rajeev Goré, Linda Postniece, and Alwen Tiu. Cut-elimination and proof-search for bi-intuitionistic logic using nested sequents. In Carlos Areces and Robert Goldblatt, editors, *Advances in Modal Logic 7*, pages 43–66, Nancy, France, 2008. College Publications. URL: http://www.aiml.net/volumes/volume7/Gore-Postniece-Tiu.pdf.

14 Rajeev Goré, Linda Postniece, and Alwen Tiu. On the correspondence between display postulates and deep inference in nested sequent calculi for tense logics. *Logical Methods in Computer Science*, 7(2):1–38, 2011. doi:10.2168/LMCS-7(2:8)2011.

15 Rajeev Goré and Ian Shillito. Bi-intuitionistic logics: A new instance of an old problem. In *Advances in Modal Logic 13*, pages 269–288, Helsinki, Finland, 2020. College Publications. URL: http://www.aiml.net/volumes/volume13/Gore-Shillito.pdf.

16 Andrzej Grzegorczyk. A philosophically plausible formal interpretation of intuitionistic logic. *Indagationes Mathematicae*, 26(5):596–601, 1964.

17 Ryo Ishigaki and Kentaro Kikuchi. Tree-sequent methods for subintuitionistic predicate logics. In Nicola Olivetti, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 4548 of *Lecture Notes in Computer Science*, pages 149–164, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-73099-6_13.

18 Ryo Kashima. Cut-free sequent calculi for some tense logics. *Studia Logica*, 53(1):119–135, 1994. doi:10.1007/BF01053026.

19 Tim Lyon. On the correspondence between nested calculi and semantic systems for intuitionistic logics. *Journal of Logic and Computation*, 31(1):213–265, December 2020. doi:10.1093/logcom/exaa078.

20 Tim Lyon. *Refining Labelled Systems for Modal and Constructive Logics with Applications*. PhD thesis, Technische Universität Wien, 2021.

21 Tim Lyon, Alwen Tiu, Rajeev Goré, and Ranald Clouston. Syntactic interpolation for tense logics and bi-intuitionistic logic via nested sequents. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic*, volume 152 of *LIPIcs*, pages 28:1–28:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CSL.2020.28.

22 Tim S. Lyon. Nested sequents for intermediate logics: the case of Gödel-Dummett logics. *Journal of Applied Non-Classical Logics*, 33(2):121–164, 2023. doi:10.1080/11663081.2023.2233346.

23 Tim S. Lyon. Nested sequents for intermediate logics: The case of Gödel-Dummett logics, 2024. Updated version, on arXiv. URL: https://arxiv.org/abs/2306.07550, doi:10.48550/arXiv.2306.07550.

24 Tim S. Lyon and Lucía Gómez Álvarez. Automating reasoning with standpoint logic via nested sequents. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pages 257–266, August 2022. doi:10.24963/kr.2022/26.

25 Tim S. Lyon and Jonas Karge. Constructive interpolation and concept-based beth definability for description logics via sequents. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 3484–3492. International Joint Conferences on Artificial Intelligence Organization, August 2024. Main Track. URL: https://www.ijcai.org/proceedings/2024/386, doi:10.24963/ijcai.2024/386.

26 Tim S. Lyon, Ian Shillito, and Alwen Tiu. Taking bi-intuitionistic logic first-order: A proof-theoretic investigation via polytree sequents, 2024. Appended version on arXiv. URL: https://arxiv.org/abs/2404.15855, doi:10.48550/arXiv.2404.15855.

27 E. G. K. López-Escobar. On the interpolation theorem for the logic of constant domains. *Journal of Symbolic Logic*, 46(1):87–88, 1981. doi:10.2307/2273260.

28 Sara Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34(5-6):507, 2005.

29 Luís Pinto and Tarmo Uustalu. Relating sequent calculi for bi-intuitionistic propositional logic. In Steffen van Bakel, Stefano Berardi, and Ulrich Berger, editors, *Proceedings Third International Workshop on Classical Logic and Computation*, volume 47 of *EPTCS*, pages 57–72, 2010. doi:10.4204/EPTCS.47.7.

30 Luís Pinto and Tarmo Uustalu. A proof-theoretic study of bi-intuitionistic propositional sequent calculus. *Journal of Logic and Computation*, 28(1):165–202, 2018. doi:10.1093/LOGCOM/EXX044.

31 Linda Postniece. Deep inference in bi-intuitionistic logic. In Hiroakira Ono, Makoto Kanazawa, and Ruy de Queiroz, editors, *Logic, Language, Information and Computation*, pages 320–334, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-02261-6_26.

32 Cecylia Rauszer. Applications of Kripke models to Heyting-Brouwer logic. *Studia Logica*, 36(1):61–71, 1977. doi:10.1007/BF02121115.

33 Cecylia Rauszer. An algebraic and Kripke-style approach to a certain extension of intuitionistic logic, dissertations mathematicae, 1980.

34 Greg Restall. Constant domain quantified modal logics without boolean negation. *Australasian Journal of Logic*, 3:45–62, 2005. doi:10.26686/ajl.v3i0.1772.

35 Dana Scott. Identity and existence in intuitionistic logic. In *Applications of Sheaves: Proceedings of the Research Symposium on Applications of Sheaf Theory to Logic, Algebra, and Analysis, Durham, July 9–21, 1977*, pages 660–696. Springer, 2006.

36 Ian Shillito. *New Foundations for the Proof Theory of Bi-Intuitionistic and Provability Logics Mechanized in Coq*. PhD thesis, Australian National University, Canberra, 2023.

37 Alex K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, University of Edinburgh. College of Science and Engineering. School of Informatics, 1994.

38 John G. Stell, Renate A. Schmidt, and David Rydeheard. A bi-intuitionistic modal logic: Foundations and automation. *Journal of Logical and Algebraic Methods in Programming*, 85(4):500–519, 2016. Relational and algebraic methods in computer science. doi:10.1016/j.jlamp.2015.11.003.

39 Gaisi Takeuti. *Proof theory*, volume 81. Courier Corporation, 2013.

40 Alwen Tiu, Egor Ianovski, and Rajeev Goré. Grammar logics in nested sequent calculus: Proof theory and decision procedures. In Thomas Bolander, Torben Braüner, Silvio Ghilardi, and Lawrence S. Moss, editors, *Advances in Modal Logic 9*, pages 516–537. College Publications, 2012. URL: http://www.aiml.net/volumes/volume9/Tiu-Ianovski-Gore.pdf.

41 Luca Viganò. *Labelled Non-Classical Logics*. Springer Science & Business Media, 2000.

A Soundness

▶ **Theorem 20** (Soundness). *Let S be a sequent. If S is provable in* LBIQ($\mathcal{ID}$) *(*LBIQ($\mathcal{CD}$)*), then S is (*CD-*)valid.*

Proof. We argue the claim by induction on the height of the given derivation and consider the LBIQ($\mathcal{ID}$) case as the LBIQ($\mathcal{CD}$) case is similar.

Base case. It is straightforward to show that any instance of ($\bot$L) or ($\top$R) is valid; hence, we focus on (ax) and show that any instance thereof is valid. Let us consider the following instance of (ax), where $w \twoheadrightarrow^*_{\mathcal{R}} u$ due to the side condition imposed on (ax), that is, there exist $v_1, \ldots, v_n \in \mathrm{Lab}(\mathcal{R})$ such that $wRv_1, \ldots, v_nRu \in \mathcal{R}$.

$$\frac{}{\mathcal{R}, \mathcal{T}, \Gamma, w : p(\vec{t}) \vdash \Delta, u : p(\vec{t})}\ (ax)$$

Let us suppose $\mathcal{R}, \mathcal{T}, \Gamma, w : p(\vec{t}) \vdash \Delta, u : p(\vec{t})$ is invalid, i.e., a model $M = (W, \leq, U, D, I_1, I_2)$, M-interpretation ι, and M-assignment α exist such that the following hold: $\iota(w) \leq \iota(v_1), \ldots, \iota(v_n) \leq \iota(u)$, $M, \iota(w), \alpha \Vdash p(\vec{t})$, and $M, \iota(u), \alpha \nVdash p(\vec{t})$. By the monotonicity condition (M) (see Definition 3), it must be that $M, \iota(u), \alpha \Vdash p(\vec{t})$, giving a contradiction. Thus, every instance of (ax) must be valid.

Inductive step. We prove the inductive step by contraposition, showing that if the conclusion of the last inference in the given proof is invalid, then at least one premise of the final inference must be invalid. We make a case distinction based on the last rule applied in the given derivation.

(*ds*). Suppose $\mathcal{R}, \mathcal{T}, \Gamma, w : p(\vec{t}) \vdash \Delta$ is invalid with $\vec{t} = t_1, \ldots, t_n$. Then, there exists a model M, M-interpretation ι, and M-assignment α such that $M, \iota(w), \alpha \Vdash p(\vec{t})$. Therefore, $(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_n)) \in I_2(w, p)$, and since $I_2(w, p) \subseteq D(w)^n$, we have that $\overline{\alpha}(t_i) \in D(\iota(w))$ for $1 \leq i \leq n$. By the (C_1) and (C_2) conditions, we know that $M, \iota, \alpha \models w : VT(\vec{t})$. Therefore, $\mathcal{R}, \mathcal{T}, w : VT(\vec{t}), \Gamma, w : p(\vec{t}) \vdash \Delta$ is invalid as well.

($\wedge$L). If we assume that $\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \wedge \psi \vdash \Delta$ is invalid, then there exists a model M, M-interpretation ι, and M-assignment α such that $M, \iota(w), \alpha \Vdash \varphi \wedge \psi$, implying that $M, \iota(w), \alpha \Vdash \varphi$ and $M, \iota(w), \alpha \Vdash \psi$, showing that the premise $\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi, w : \psi \vdash \Delta$ is invalid as well.

($\wedge$R). Let us suppose that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w : \varphi \wedge \psi$ is invalid. Then, there exists an model M, M-interpretation ι, and M-assignment α such that $M, \iota(w), \alpha \nVdash \varphi \wedge \psi$. Hence, either $M, \iota(w), \alpha \nVdash \varphi$ or $M, \iota(w), \alpha \nVdash \psi$. In the first case, the left premise of ($\wedge$R) is invalid, and in the second case, the right premise of ($\wedge$R) is invalid.

($\vee$L). Similar to the ($\wedge$R) case.

($\vee$R). Similar to the ($\wedge$L) case.

($\to$L). Assume $\mathcal{R}, \mathcal{T}, \Gamma, w : \varphi \to \psi \vdash \Delta$ is invalid and $w \twoheadrightarrow^*_{\mathcal{R}} u$, i.e. a sequence $wRv_1, \ldots, v_nRu$ of relational atoms exist in $\mathcal{R}$. By our assumption, there exists a model M, M-interpretation ι, and M-assignment such that $\iota(w) \leq \iota(v_1), \ldots, \iota(v_n) \leq \iota(u)$ and $M, \iota(w), \alpha \Vdash \varphi \to \psi$. Because $M, \iota(w), \alpha \Vdash \varphi \to \psi$ and $\leq$ is transitive, we know that either $M, \iota(u), \alpha \nVdash \varphi$ or $M, \iota(u), \alpha \Vdash \psi$. In the first case, the left premise of ($\to$L) is invalid, and in the second case, the right premise of ($\to$L) is invalid.

($\to$R). Assume that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w : \varphi \to \psi$ is invalid. Then, there exists a model M, an M-interpretation ι, and an M-assignment α such that $M, \iota(w), \alpha \nVdash \varphi \to \psi$. Hence, there exists a world u such that $\iota(w) \leq u$, $M, u, \alpha \Vdash \varphi$, and $M, u, \alpha \nVdash \psi$. Let $\iota'(v) = \iota(v)$ for all labels $v \neq u$ and $\iota'(u) = u$ otherwise. Then, M, ι', and α falsify the premise of ($\to$R), showing it invalid.

($-\!\!\prec$L). Similar to the ($\to$R) case above.

($-\!\!\prec$R). Similar to the ($\to$L) case above.

($\exists$L). Suppose that $S = \mathcal{R}, \mathcal{T}, \Gamma, w : \exists x\varphi \vdash \Delta$ is invalid. Then, there exists a model M, an M-interpretation ι, and an M-assignment α such that $M, \iota(w), \alpha \Vdash \exists x\varphi$. Therefore, there exists an $a \in D(\iota(w))$ such that $M, \iota(w), \alpha[a/y] \Vdash \varphi(y/x)$ with y not occurring in S. Then, as y is fresh, M, ι, and $\alpha[a/y]$ falsify the premise of ($\exists$L), showing it invalid.

($\exists$R). Suppose that $S = \mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w : \exists x\varphi$ is invalid. Then, there exists a model M, and M-interpretation ι, and M-assignment such that $M, \iota(w), \alpha \nVdash \exists x\varphi$. By the side condition on ($\exists$R), we know that $\mathsf{A}(t, X_w, \mathcal{R}, \mathcal{T})$, meaning there exist labels $u_1, \ldots, u_n \in \mathrm{Lab}(S)$ such that $u_1 : x_1, \ldots, u_n : x_n \in \mathcal{T}$, $VT(t) = \{x_1, \ldots, x_n\}$, and $u_1 \twoheadrightarrow^*_{\mathcal{R}} w, \ldots, u_n \twoheadrightarrow^*_{\mathcal{R}} w$. It follows that $\iota(u_1) \leq \iota(w), \ldots, \iota(u_n) \leq \iota(w)$ and $\overline{\alpha}(x_1) \in D(\iota(u_1)), \ldots, \overline{\alpha}(x_n) \in D(\iota(u_n))$. By the increasing domain condition (ID), we have that $\overline{\alpha}(x_1) \in D(\iota(w)), \ldots, \overline{\alpha}(x_n) \in D(\iota(w))$. Therefore, by the ($C_1$) and ($C_2$) conditions, we know that $\overline{\alpha}(t) \in D(\iota(w))$, showing that $M, \iota(w), \alpha \nVdash \varphi(t/x)$, and thus, the premise is invalid.

($\forall$L). Suppose that $S = \mathcal{R}, \mathcal{T}, \Gamma, w : \forall x\varphi \vdash \Delta$ is invalid. Then, there exists a model M, and M-interpretation ι, and M-assignment α such that $M, \iota(w), \alpha \Vdash \forall x\varphi$. By the side condition on ($\forall$L), we know that $w \twoheadrightarrow^*_{\mathcal{R}} u$ and $\mathsf{A}(t, X_w, \mathcal{R}, \mathcal{T})$. By the latter fact, there exist labels $v_1, \ldots, v_n \in \mathrm{Lab}(S)$ such that $v_1 : x_1, \ldots, v_n : x_n \in \mathcal{T}$, $VT(t) = \{x_1, \ldots, x_n\}$, and $v_1 \twoheadrightarrow^*_{\mathcal{R}} w, \ldots, v_n \twoheadrightarrow^*_{\mathcal{R}} w$. It follows that $\iota(v_1) \leq \iota(w), \ldots, \iota(v_n) \leq \iota(w)$ and $\overline{\alpha}(x_1) \in D(\iota(v_1)), \ldots, \overline{\alpha}(x_n) \in D(\iota(v_n))$. By the increasing domain condition (ID), we have that $\overline{\alpha}(x_1) \in D(\iota(w)), \ldots, \overline{\alpha}(x_n) \in D(\iota(w))$. Therefore, by the ($C_1$) and ($C_2$) conditions and our assumption, we know that $\overline{\alpha}(t) \in D(\iota(w))$, showing that $M, \iota(w), \alpha \Vdash \varphi(t/x)$. By the fact that $w \twoheadrightarrow^*_{\mathcal{R}} u$, we know $\iota(w) \leq \iota(u)$ and $\overline{\alpha}(t) \in D(\iota(u))$, showing that $M, \iota(u), \alpha \Vdash \varphi(t/x)$ by Proposition 6. Thus, the premise is invalid.

($\forall$R). Let us assume that $\mathcal{R}, \mathcal{T}, \Gamma \vdash \Delta, w : \forall x\varphi$ is invalid. Then, there exists a model M, an M-interpretation ι, and an M-assignment α such that $M, \iota(w), \alpha \nVdash \forall x\varphi$. Thus, there exists a world $u \in W$ such that $\iota(w) \leq u$, $a \in D(u)$, and $M, u, \alpha[a/y] \nVdash \varphi(y/x)$. We define $\iota'(v) = \iota(v)$ if $v \neq u$ and $\iota'(u) = u$. Then, M, ι', and $\alpha[a/y]$ falsify the premise $\mathcal{R}, w \leq u, \mathcal{T}, u : y, \Gamma \vdash \Delta, u : \varphi(y/x)$, showing it invalid. ◀

Unifying Sequent Systems for Gödel-Löb Provability Logic via Syntactic Transformations

Tim S. Lyon
Technische Universität Dresden, Germany

Abstract

We demonstrate the inter-translatability of proofs between the most prominent sequent-based formalisms for Gödel-Löb provability logic. In particular, we consider Sambin and Valentini's sequent system $\mathsf{GL}_{\mathsf{seq}}$, Shamkanov's non-wellfounded and cyclic sequent systems GL_{∞} and $\mathsf{GL}_{\mathsf{circ}}$, Poggiolesi's tree-hypersequent system CSGL, and Negri's labeled sequent system G3GL. Shamkanov provided proof-theoretic correspondences between $\mathsf{GL}_{\mathsf{seq}}$, GL_{∞}, and $\mathsf{GL}_{\mathsf{circ}}$, and Goré and Ramanayake showed how to transform proofs between CSGL and G3GL, however, the exact nature of proof transformations between the former three systems and the latter two systems has remained an open problem. We solve this open problem by showing how to restructure tree-hypersequent proofs into an end-active form and introduce a novel *linearization technique* that transforms such proofs into linear nested sequent proofs. As a result, we obtain a new proof-theoretic tool for extracting linear nested sequent systems from tree-hypersequent systems, which yields the first cut-free linear nested sequent calculus LNGL for Gödel-Löb provability logic. We show how to transform proofs in LNGL into a certain normal form, where proofs repeat in stages of modal and local rule applications, and which are translatable into $\mathsf{GL}_{\mathsf{seq}}$ and G3GL proofs. These new syntactic transformations, together with those mentioned above, establish full proof-theoretic correspondences between $\mathsf{GL}_{\mathsf{seq}}$, GL_{∞}, $\mathsf{GL}_{\mathsf{circ}}$, CSGL, G3GL, and LNGL while also giving (to the best of the author's knowledge) the first constructive proof mappings between structural (viz. labeled, tree-hypersequent, and linear nested sequent) systems and a cyclic sequent system.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Modal and temporal logics; Theory of computation → Constructive mathematics

Keywords and phrases Cyclic proof, Gödel-Löb logic, Labeled sequent, Linear nested sequent, Modal logic, Non-wellfounded proof, Proof theory, Proof transformation, Tree-hypersequent

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.42

Funding Work supported by European Research Council, Consolidator Grant DeciGUT (771779).

1 Introduction

Provability logics are a class of modal logics where the $\Box$ operator is read as "it is provable that" in some arithmetical theory. One of the most prominent provability logics is Gödel-Löb logic (GL), which arose out of the work of Löb, who formulated a set of conditions on the provability predicate of Peano Arithmetic (PA). The logic GL can be axiomatized as an extension of the basic modal logic K by the single axiom $\Box(\Box\varphi \to \varphi) \to \Box\varphi$, called *Löb's axiom*. It is well-known that the axioms of GL are sound and complete relative to transitive and conversely-wellfounded relational models [38]. In a landmark result, Solovay [41] remarkably showed that GL is complete for PA's provability logic, i.e., GL proves everything that PA can prove about its own provability predicate.

The logic GL enjoys a rich structural proof theory, possessing a number of cut-free sequent-style systems. Sequent systems in the style of Gentzen were originally provided by Sambin and Valentini in the early 1980s [36, 37]; see also Avron [2]. (NB. In this work, we take a *Gentzen system* to be a proof system whose rules operate over *Gentzen sequents*, i.e., expressions of the form $\varphi_1, \ldots, \varphi_n \vdash \psi_1, \ldots, \psi_k$ such that φ_i and ψ_j are logical formulae.)

Since then, a handful of alternative systems have been introduced, each of which either generalizes the structure of sequents or generalizes the notion of proof. The labeled sequent system G3GL was provided by Negri [31] and uses *labeled sequents* in proofs, which are binary graphs whose nodes are Gentzen sequents. In a similar vein, the tree-hypersequent system CSGL was provided by Poggiolesi [35] and uses *tree-hypersequents* in proofs, which are trees whose nodes are Gentzen sequents. The use of *(types of) graphs* of Gentzen sequents in proofs (as in G3GL and CSGL) allows for the systems to possess properties beyond those of the original Gentzen systems [36, 37]. For example, both G3GL and CSGL enjoy invertibility of all rules, rules are symmetric (i.e., for each logical connective, there is at least one rule that introduces it in the antecedent and at least one rule that introduces it in the consequent of the rule's conclusion), and the close connection between the syntax of such sequents and GL's relational semantics makes such systems suitable for counter-model extraction.

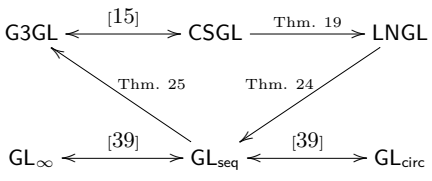
Rather than generalizing the structure of sequents, Shamkanov [39] showed that one could obtain alternative cut-free sequent systems for GL by generalizing the structure of proofs. In particular, by taking the sequent calculus for the modal logic K4 and allowing for *non-wellfounded proofs*, one obtains a non-wellfounded sequent system GL_∞ for GL. Non-wellfounded proofs were introduced to capture (co)inductive reasoning, and are potentially infinite trees of sequents such that (1) every parent node is the conclusion of a rule with its children the corresponding premises and (2) infinite branches satisfy a certain *progress condition*, which ensures soundness (cf. [5, 11, 32, 39]). For GL, non-wellfounded proofs correspond to regular trees (i.e., only contain finitely many distinct sub-trees), which means such proofs can be "folded" into finite trees of sequents such that leaves are "linked" to internal nodes of the tree, giving rise to *cyclic proofs* (cf. [1, 3, 4, 10]). Shamkanov [39] additionally showed that one could obtain a cut-free cyclic sequent system $\mathsf{GL}_{\mathsf{circ}}$ for GL by allowing cyclic proofs in K4's sequent calculus, which was then used to provide the first syntactic proof of the Lyndon interpolation property for GL.

Due to the diversity in GL's proof theory, it is natural to wonder about the relationships between the various systems that have been introduced. Typically, proof systems are related by means of *proof transformations*, which are functions that map proofs from one calculus into another, are sensitive to the structure of the input proof, and operate syntactically by permuting rules, replacing rules, or adding/deleting sequent structure in the input proof to yield the output proof. Studying proof transformations between sequent systems is a beneficial enterprise as it lets one transfer results from one system to another, thus alleviating the need of independent proofs in each system (e.g., [9, 15]). Moreover, one can measure the relative sizes or certain characteristics of proofs, giving insight into which systems are better suited for specific (automated) reasoning tasks, and letting one "toogle" between differing formalisms when one is better suited for a task than another (e.g., [24, 23]).

Indeed, the question of the relationship between G3GL and CSGL was asked by Poggiolesi [35] and answered in full by Goré and Ramanayake [15], who provided constructive mappings of proofs between the two systems. Similarly, Shamkanov [39] provided syntactic mappings of proofs between the systems GL_∞ and $\mathsf{GL}_{\mathsf{circ}}$, and the Gentzen system $\mathsf{GL}_{\mathsf{seq}}$ (an equivalent reformulation of Sambin and Valentini's systems [36, 37]). Nevertheless, the inter-translatability of proofs between the former two structural sequent systems (G3GL and CSGL) and the latter three sequent systems ($\mathsf{GL}_{\mathsf{seq}}$, GL_∞, and $\mathsf{GL}_{\mathsf{circ}}$) has yet to be identified, and presents a non-trivial open problem that we solve in this paper. Thus, our first contribution in this paper is to "complete the picture" and establish complete correspondences between the above five mentioned sequent systems by means of syntactic proof transformations.

There is an inherent difficulty in transforming proofs that use structural sequents (e.g., labeled sequents or tree-hypersequents) into proofs that use Gentzen sequents. This is due to the fact that structural sequents are (types of) graphs of Gentzen sequents, and thus, possess a more complicated structure that must be properly "shed" during proof transformations [23, 26]. To overcome this difficulty and define proof transformations from G3GL and CSGL to $\mathsf{GL}_{\mathsf{seq}}$, we rely on three techniques: first, we show how to restructure proofs in CSGL so that they are *end-active* (cf. [21]), meaning rules only affect data at leaves or parents of leaves in tree-hypersequents. Second, we introduce a novel *linearization technique*, whereby we show how to shed the tree structure of tree-hypersequents in end-active proofs, yielding a proof consisting solely of *linear nested sequents* [21], i.e., lines whose nodes are Gentzen sequents. Linear nested sequents were introduced as an alternative (albeit equivalent) formalism to 2-sequents [29, 30] that allows for sequent systems with complexity-optimal proof-search that also retain fundamental admissibility and invertibility properties [21]. The presented linearization technique is new and shows how to extract linear nested sequent systems from tree-hypersequent systems, serving as the second contribution of this paper. We conjecture that this method can be generalized and applied in other settings to provide new linear nested sequent systems for modal and related logics. The technique also yields the first (cut-free) linear nested sequent calculus for GL, which we dub LNGL, and which is the third contribution of this paper. Last, we show that proofs in LNGL can be put into a specific normal form that repeats in stages of modal and local rules. Such proofs are translatable into $\mathsf{GL}_{\mathsf{seq}}$ proofs, which are translatable into G3GL proofs, thus establishing purely syntactic proof transformations between the most prominent sequent systems for GL. These proof transformations and systemic correspondences are summarized in Figure 1 below.

Outline of Paper. In Section 2, we recall the language, semantics, and axioms of Gödel-Löb logic. In Section 3, we discuss Negri's labeled system G3GL [31], Poggiolesi's tree-hypersequent system CSGL [35], and Goré and Ramanayake correspondence result for the two systems [15]. In Section 4, we show how to put proofs in CSGL into an end-active form (Theorem 18) and specify our novel linearization method (Theorem 19), which yields the new linear nested sequent system LNGL for GL. In Section 5, we recall the sequent calculus $\mathsf{GL}_{\mathsf{seq}}$ due to Sambin and Valentini [36, 37], Shamkanov's non-wellfounded system GL_{∞} and cyclic system $\mathsf{GL}_{\mathsf{circ}}$, as well as Shamkanov's correspondence result for the aforementioned three systems [39]. We then show how to transform proofs in LNGL into proofs in $\mathsf{GL}_{\mathsf{seq}}$ (Theorem 24) and how to transform proofs in $\mathsf{GL}_{\mathsf{seq}}$ into proofs in G3GL (Theorem 25). This establishes a six-way correspondence between the systems G3GL, CSGL, $\mathsf{GL}_{\mathsf{seq}}$, GL_{∞}, $\mathsf{GL}_{\mathsf{circ}}$, and LNGL. Last, in Section 6, we conclude and discuss future work.



Name	Type of System
G3GL	Labeled Sequent System
CSGL	Tree-Hypersequent System
LNGL	Linear Nested Sequent System
$\mathsf{GL}_{\mathsf{seq}}$	Gentzen System
GL_{∞}	Non-Wellfounded Sequent System
$\mathsf{GL}_{\mathsf{circ}}$	Cyclic Sequent System

Figure 1 Proof transformations and correspondences between sequent systems for GL.

2 Gödel-Löb Provability Logic

We let $\mathsf{Prop} := \{p, q, r, \ldots\}$ be a countable set of *propositional atoms* and define the language $\mathcal{L}$ to be the set of all formulae generated via the following grammar in BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \Box\varphi$$

where p ranges over Prop. We use φ, ψ, χ, ... to denote formulae in $\mathcal{L}$ and define $\varphi \wedge \psi := \neg(\neg\varphi \vee \neg\psi)$ and $\varphi \to \psi := \neg\varphi \vee \psi$ as usual.

▶ **Definition 1** (Model). *We define a* model *to be a tuple $M = (W, R, V)$ such that*
- *W is a non-empty set of worlds w, u, v, ... (occasionally annotated);*
- *$R \subseteq W \times W$ is transitive and conversely-wellfounded;*[1]
- *$V : \mathsf{Prop} \mapsto 2^W$ is a* valuation function.

▶ **Definition 2** (Semantic Clauses). *We define the* satisfaction *of a formula φ in a model M at world w, written $M, w \models \varphi$, recursively as follows:*
- *$M, w \models p$ iff $w \in V(p)$;*
- *$M, w \models \neg\varphi$ iff $M, w \not\models \varphi$;*
- *$M, w \models \varphi \vee \psi$ iff $M, w \models \varphi$ or $M, w \models \psi$;*
- *$M, w \models \Box\varphi$ iff $\forall u \in W$, if $(w, u) \in R$, then $M, u \models \varphi$;*
- *$M \models \varphi$ iff $\forall w \in W$, $M, w \models \varphi$.*

We write $\models \varphi$ and say that φ is valid *iff for all models M, $M \models \varphi$.* Gödel-Löb logic (GL) *is defined to be the set $\mathsf{GL} \subset \mathcal{L}$ of all valid formulae.*

As shown by Segerberg [38], the logic GL can be axiomatized by extending the axioms of the modal logic K with Löb's axiom $\Box(\Box\varphi \to \varphi) \to \Box\varphi$.

3 Labeled and Tree Sequent Systems

In this section, we review the labeled sequent calculus G3GL by Negri [31] and its correspondence (proven by Goré and Ramanayake [15]) with a notational variant of Poggiolesi's tree-hypersequent system for GL [35]. *Labeled sequents* are binary graphs of traditional Gentzen sequents, which encode the relational semantics of a logic directly in the syntax of sequents. The formalism of labeled sequents has been extensively studied with the inception of the formalism dating back to the work of Kanger [18] and achieving its modern form in the work of Simpson [40]. It has been shown that labeled sequent systems can capture sizable and diverse classes of logics in a cut-free manner while exhibiting fundamental properties such as the admissibility of various structural rules and the invertibility of rules [40, 43, 31].

By contrast, *tree-hypersequents*, which are more traditionally known as *nested sequents*, are trees of Gentzen sequents. The formalism was introduced independently by Kashima [19] and Bull [7] with further influential works provided by Brünnler [6] and Poggiolesi [34, 35]. Such systems arose out of a call for cut-free sequent-style systems for logics not known to possess a cut-free Gentzen system, such as the tense logic $\mathsf{K_t}$ and the modal logic S5. Like labeled sequents, tree-hypersequent systems exhibit fundamental admissibility and invertibility properties, having been defined for large classes of various logics such as tense logics [14], intuitionistic modal logics [42, 25], and first-order non-classical logics [13, 27].

[1] We note that R is conversely-wellfounded *iff* it does not contain any infinite ascending R-chains.

$$\frac{}{\mathcal{R}, \Gamma, x : p \vdash x : p, \Delta}\ \mathsf{id}_1 \qquad \frac{}{\mathcal{R}, \Gamma, x : \Box\varphi \vdash x : \Box\varphi, \Delta}\ \mathsf{id}_2 \qquad \frac{}{\mathcal{R}, xRx, \Gamma \vdash \Delta}\ \mathsf{ir}$$

$$\frac{\mathcal{R}, xRy, yRz, xRz, \Gamma \vdash \Delta}{\mathcal{R}, xRy, yRz, \Gamma \vdash \Delta}\ \mathsf{tr} \qquad \frac{\mathcal{R}, \Gamma, x : \varphi \vdash \Delta \qquad \mathcal{R}, \Gamma, x : \psi \vdash \Delta}{\mathcal{R}, \Gamma, x : \varphi \vee \psi \vdash \Delta}\ \vee\mathsf{L}$$

$$\frac{\mathcal{R}, \Gamma \vdash x : \varphi, \Delta}{\mathcal{R}, \Gamma, x : \neg\varphi \vdash \Delta}\ \neg\mathsf{L} \qquad \frac{\mathcal{R}, \Gamma, x : \varphi \vdash \Delta}{\mathcal{R}, \Gamma \vdash x : \neg\varphi, \Delta}\ \neg\mathsf{R} \qquad \frac{\mathcal{R}, \Gamma \vdash x : \varphi, x : \psi, \Delta}{\mathcal{R}, \Gamma \vdash x : \varphi \vee \psi, \Delta}\ \vee\mathsf{R}$$

$$\frac{\mathcal{R}, xRy, \Gamma, x : \Box\varphi, y : \varphi \vdash \Delta}{\mathcal{R}, xRy, \Gamma, x : \Box\varphi \vdash \Delta}\ \Box\mathsf{L} \qquad \frac{\mathcal{R}, xRy, \Gamma, y : \Box\varphi \vdash y : \varphi, \Delta}{\mathcal{R}, \Gamma \vdash x : \Box\varphi, \Delta}\ \Box\mathsf{R}^{\dagger}$$

Figure 2 Labeled Sequent Calculus G3GL for GL. The $\Box$R rule is subject to a side condition †, namely, the rule is applicable only if the label y is fresh.

As observed by Goré and Ramanayake [15], restricting labeled sequents to be trees, rather than more general, binary graphs (which may be disconnected or include cycles), yields labeled tree sequents (cf. [17]), which are a notational variant of tree-hypersequents/nested sequents. Via this observation, the authors established bi-directional proof transformations between Negri's labeled sequent calculus and Poggiolesi's tree-hypersequent calculus for GL. Proof theoretic correspondences between labeled and nested systems for various other logics have been established in recent years as well; e.g., for tense logics [9], first-order intuitionistic logics [23, 22], and intuitionistic modal logics [25]. Recently, it was proven in a general setting that correspondences between labeled and nested systems are a product of two underlying proof transformation techniques, structural rule elimination and introduction, and that (Horn) labeled and nested systems tend to come in pairs, being dual to one another [28].

Reducing Negri's labeled sequent system to one that uses trees, as opposed to binary graphs, is the first step in establishing syntactic correspondences between the various sequent systems for GL. As shown in the sequel, we will systematically reduce the structure of sequents in proofs: first, going from binary graphs of Gentzen sequents to trees of Gentzen sequents (this section), then from trees of Gentzen sequents to lines of Gentzen sequents (Section 4), and last from lines of Gentzen sequents to Gentzen sequents themselves, which are easily embedded in labeled sequent proofs, completing the circuit of correspondences (Section 5). This yields correspondences between the most widely regarded sequent systems for GL, as depicted in Figure 1.

3.1 Labeled Sequents

We let $\mathrm{Lab} = \{x, y, z, \ldots\}$ be a countably infinite set of *labels*, define a *relational atom* to be an expression of the form xRy with $x, y \in \mathrm{Lab}$, and define a *labeled formula* to be an expression of the form $x : \varphi$ such that $x \in \mathrm{Lab}$ and $\varphi \in \mathcal{L}$. We use upper-case Greek letters $\Gamma, \Delta, \Sigma, \ldots$ to denote finite multisets of labeled formulae. For a set $\mathcal{R}$ of relational atoms and multiset Γ of labeled formulae, we let $\mathrm{Lab}(\mathcal{R})$, $\mathrm{Lab}(\Gamma)$, and $\mathrm{Lab}(\mathcal{R}, \Gamma)$ be the sets of all labels occurring therein. For a multiset Γ of labeled formulae, we define the multiset $\Gamma(x) := \{\varphi \mid x : \varphi \in \Gamma\}$, for a multiset of formulae $\Gamma := \varphi_1, \ldots, \varphi_n$, we define $x : \Gamma := x : \varphi_1, \ldots, x : \varphi_n$, and for multisets Γ and Δ of labeled formulae, we let Γ, Δ denote the multiset union of the two. We define a *labeled sequent* to be an expression of the form $\mathcal{R}, \Gamma \vdash \Delta$ with $\mathcal{R}$ a set of relational atoms and Γ, Δ a multiset of labeled formulae. Given a labeled sequent $\mathcal{R}, \Gamma \vdash \Delta$, we refer to $\mathcal{R}, \Gamma$ as the *antecedent* and Δ as the *consequent*. Below, we clarify the interpretation of labeled sequents by explaining their evaluation over models.

$$\frac{\mathcal{R},\Gamma\vdash\Delta}{\mathcal{R}(x/y),\Gamma(x/y)\vdash\Delta(x/y)}\,(x/y)\qquad \frac{\mathcal{R},\Gamma\vdash\Delta}{\mathcal{R},\mathcal{R}',\Gamma,\Gamma'\vdash\Delta,\Delta'}\,\mathsf{w}\qquad \frac{\mathcal{R},\Gamma,x:\varphi,x:\varphi\vdash\Delta}{\mathcal{R},\Gamma,x:\varphi\vdash\Delta}\,\mathsf{cL}$$

$$\frac{\mathcal{R},\Gamma\vdash x:\varphi,x:\varphi,\Delta}{\mathcal{R},\Gamma\vdash x:\varphi,\Delta}\,\mathsf{cR}\qquad \frac{\mathcal{R},\Gamma\vdash x:\varphi,\Delta\qquad \mathcal{R},\Gamma,x:\varphi\vdash\Delta}{\mathcal{R},\Gamma\vdash\Delta}\,\mathsf{cut}$$

Figure 3 Admissible rules.

▶ **Definition 3** (Labeled Sequent Semantics). *Let $M=(W,R,V)$ be a model. We define an* M-assignment *to be a function $\mu\colon \mathrm{Lab}\to W$. A labeled sequent $\mathcal{R},\Gamma\vdash\Delta$ is* satisfied *on M with M-assignment μ iff if for all $xRy\in\mathcal{R}$ and $x:\varphi\in\Gamma$, $(\mu(x),\mu(y))\in R$ and $M,\mu(x)\models\varphi$, then there exists a $y:\psi\in\Delta$ such that $M,\mu(y)\models\psi$. A labeled sequent is defined to be* valid *iff it is satisfied on all models M with all M-assignments; a labeled sequent is defined to be* invalid *otherwise.*

Negri's labeled sequent calculus G3GL (adapted to our signature) is shown in Figure 2. The labeled calculus consists of three initial rules id_1, id_2, and ir. We refer to the conclusion of an initial rule as a *initial sequent*. The tr rule is a structural rule that bottom-up adds transitive edges to labeled sequents, and the remaining rules form pairs of left and right logical rules, introducing complex logical formulae into either the antecedent or consequent of the rule's conclusion. We note that the $\Box$R rule is subject to a side condition, namely, the label y must be *fresh* in any application of the rule, i.e., the label y is forbidden to occur in the conclusion. We refer to the distinguished formulae in the conclusion (premises) of a rule as the *principal formulae* (*auxiliary formulae*, respectively). For example, $x:\Box\varphi$ is principal in $\Box$R and $xRy,x:\Box\varphi,y:\varphi$ are auxiliary.

▶ Remark 4. Negri's original labeled system G3GL includes the following $\Box$L′ rule rather than the $\Box$L rule. However, the left premise of the $\Box$L′ rule is provable in G3GL using $\Box$L, $\Box$R, and tr [31]. We therefore opt to use the simpler $\Box$L rule in G3GL rather than the $\Box$L′ rule to simplify our work.

$$\frac{\mathcal{R},xRy,\Gamma,x:\Box\varphi\vdash y:\Box\varphi,\Delta\qquad \mathcal{R},xRy,\Gamma,x:\Box\varphi,y:\varphi\vdash\Delta}{\mathcal{R},xRy,\Gamma,x:\Box\varphi\vdash\Delta}\,\Box\mathsf{L}'$$

A *derivation* of a labeled sequent $\mathcal{R},\Gamma\vdash\Delta$ is defined to be a (potentially infinite) tree whose nodes are labeled with labeled sequents such that (1) $\mathcal{R},\Gamma\vdash\Delta$ is the root of the tree and (2) each parent node is the conclusion of a rule with its children the corresponding premises. A *proof* is a finite derivation such that every leaf is an instance of an initial sequent. We use π (potentially annotated) to denote derivations and proofs throughout the remainder of the paper, and use this notation to denote derivations and proofs in other systems as well with the context determining the usage. The *height* of a proof is defined as usual to be equal to the length of a maximal path from the root of the proof to an initial sequent.

▶ Remark 5. We assume w.l.o.g. that every fresh variable used in a proof is *globally fresh*, meaning there is a one-to-one correspondence between $\Box$R applications and their fresh variables. This assumption is helpful, yet benign (cf. [31]).

As shown by Negri [31], the various rules displayed in Figure 3 are *admissible* in G3GL. Note that the (x/y) rule applies a *label substitution* to the premise which replaces every occurrence of the label y in a relational atom or labeled formula by x. We define a rule to be *admissible* (*height-preserving admissible*) *iff* if the premises of the rule have proofs (of height $h_1,\ldots,h_n$), then the conclusion of the rule has a proof (of height $h\leq\max\{h_1,\ldots,h_n\}$). We refer to a height-preserving admissible rule as *hp-admissible*. Moreover, the non-initial rules

of G3GL are *height-preserving invertible*. If we let r_i^{-1} be the i-inverse of the rule r whose conclusion is the i^{th} premise of the n-ary rule r and premise is the conclusion of r, then we say that r is *(height-preserving) invertible iff* r_i^{-1} is (height-preserving) admissible for each $1 \leq i \leq n$. We refer to height-preserving invertible rules as *hp-invertible*. The following theorem is due to Negri [31].

▶ **Theorem 6** (G3GL Properties [31]). *The labeled sequent calculus* G3GL *satisfies the following:*

(1) *Each labeled sequent of the form* $\mathcal{R}, \Gamma, x : \varphi \vdash x : \varphi, \Delta$ *is provable in* G3GL*;*

(2) *All non-initial rules are hp-invertible in* G3GL*;*

(3) *The* (x/y)*,* w*,* cL*, and* cR *rules are hp-admissible in* G3GL*;*

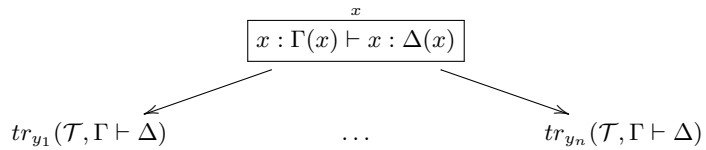
(4) *The* cut *rule is admissible in* G3GL*;*

(5) φ *is valid iff* $\vdash x : \varphi$ *is provable in* G3GL*.*

3.2 Labeled Tree Sequents

A set $\mathcal{T}$ of relational atoms is a *tree iff* the graph $G(\mathcal{T}) = (V, E)$ forms a tree, where $V = \{x \mid x \in \mathrm{Lab}(\mathcal{T})\}$ and $E = \{(x, y) \mid xRy \in \mathcal{T}\}$. A *tree sequent* is defined to be an expression of the form $\mathcal{T}, \Gamma \vdash \Delta$ such that (1) $\mathcal{T}$ is a tree, (2) if $\mathcal{T} \neq \emptyset$, then $\mathrm{Lab}(\Gamma, \Delta) \subseteq \mathrm{Lab}(\mathcal{T})$, and (3) if $\mathcal{T} = \emptyset$, then $|\mathrm{Lab}(\Gamma, \Delta)| = 1$, i.e. all labeled formulae in Γ, Δ share the same label. We note that conditions (1)–(3) ensure that each tree sequent forms a connected graph that is indeed of a tree shape. We use T and annotated versions thereof to denote tree sequents. We define a *flat sequent* to be a tree sequent of the form $\Gamma \vdash \Delta$, that is, a flat sequent is a sequent $\Gamma \vdash \Delta$ without relational atoms and where every labeled formula in Γ, Δ shares the same label. The *root* of a tree sequent $\mathcal{T}, \Gamma \vdash \Delta$ is the label x such that there exists a unique directed path of relational atoms in $\mathcal{T}$ from x to every other label $y \in \mathrm{Lab}(\mathcal{T}, \Gamma, \Delta)$; if $\mathcal{T} = \emptyset$, then the root is the single label x shared by all formulae in Γ, Δ.

Every tree sequent encodes a tree whose vertices are flat sequents. In other words, each tree sequent $T = \mathcal{T}, xRy_1, \ldots, xRy_n, \Gamma \vdash \Delta$ such that x is the root and $y_1, \ldots, y_n$ are all children of x can be graphically depicted as a tree $tr_x(T)$ of the form shown below:



▶ **Definition 7** (Tree Sequent Calculus CSGL). *We define* CSGL := (G3GL \ {ir, tr}) ∪ {4L}*, where the* 4L *is shown below and the rules of the calculus only operate over tree sequents.*

$$\frac{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \Box\varphi \vdash \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash \Delta}\ 4\mathsf{L}$$

The system CSGL is a notational variant of Poggiolesi's eponymous tree-hypersequent system [35], and thus, we identify the two systems with one another. The main difference between the two systems is notational: the system defined above uses tree sequents, which are tree-hypersequents "dressed" as labeled sequents [15]. We also note that Poggiolesi's original tree-hypersequent system CSGL uses a notational variant of the binary rule $\Box\mathsf{L}'$ discussed in Remark 4. However, as with the labeled system G3GL, the left premise of this rule is provable in CSGL. We have therefore opted to use the unary $\Box\mathsf{L}$ rule in CSGL to simplify our work and note that this change is benign.

▶ Remark 8. Derivations, proofs, the height of a proof, (hp-)admissibility, the i-inverse of a rule, and (hp-)invertibility are defined for CSGL analogous to how such notions are defined for G3GL. We will apply these terms and concepts in the expected way to other sequent systems as well to avoid repeating similar definitions.

The system CSGL differs from G3GL in that CSGL only allows for tree sequents in proofs, lacks the structural rules ir and tr, and includes the 4L rule. We refer to 4L and $\Box$L as *propagation rules* (cf. [12, 8]) since the rules bottom-up propagate data forward along relational atoms, and we refer to $\neg$L, $\neg$R, $\vee$L, and $\vee$R as *local rules* since they only affect formulae locally at a single label. For any rule, we call the label x labeling the principal formula in the conclusion the *principal label*, for the 4L, $\Box$L, and $\Box$R rules, we refer to the label y labeling the auxiliary formula(e) in the premise(s) as the *auxiliary label*, and for local rules, the *auxiliary label* is taken to be the same as the principal label since they are identical. Note that we define a label x in a tree sequent $\mathcal{T}, \Gamma \vdash \Delta$ to be a *leaf iff* x is a leaf in $\mathcal{T}$, and we define a label x to be a *pre-leaf iff* for all $y \in \mathrm{Lab}(\mathcal{T})$, if $xRy \in \mathcal{T}$, then y is a leaf.

Since the tree sequent calculus CSGL is isomorphic to Poggiolesi's tree-hypersequent system, the two systems share the same properties. We note that in the setting of tree sequents the (x/y) and w rules are less general than for labeled sequents. In particular, such rules are assumed to preserve the "tree shape" of tree sequents when applied. Nevertheless, the restricted forms of these rules are still hp-admissible in CSGL.

▶ **Theorem 9** (CSGL Properties [35]). *The tree sequent calculus* CSGL *satisfies the following:*
(1) *Each tree sequent of the form* $\mathcal{T}, \Gamma, x : \varphi \vdash x : \varphi, \Delta$ *is provable in* CSGL*;*
(2) *All non-initial rules are hp-invertible in* CSGL*;*
(3) *The* (x/y)*,* w*,* cL*, and* cR *rules are hp-admissible in* CSGL*;*
(4) *The* cut *rule is admissible in* CSGL*;*
(5) φ *is valid iff* $\vdash x : \varphi$ *is provable in* CSGL*.*

Proofs in G3GL and CSGL are inter-translatable with one another. This correspondence was established by Goré and Ramanayake [15] and is based on a couple observations. First, the ir rule does not occur in G3GL proofs where the end sequent is a tree sequent. It is not difficult to see why this is the case: if one takes a proof of a tree sequent, then bottom-up applications of rules from G3GL will not allow for directed cycles to enter a sequent in a proof. This follows from the fact that the conclusion of the proof is a tree sequent, which is free of directed cycles, and each rule of G3GL either preserves relational atoms bottom-up, adds a single relational atom from a label x to a fresh label y in the case of the $\Box$R rule, or adds an undirected cycle in the case of tr. Since the conclusion of ir contains a directed cycle xRx, such a sequent will never occur in such a proof.

▶ **Observation 10** ([15]). *The* ir *rule does not occur in any* G3GL *proof of a tree sequent.*

Second, Goré and Ramanayake [15] show that instances of tr can be eliminated from G3GL proofs not containing ir and replaced by instances of 4L. This elimination procedure can be used to map G3GL proofs such that the end sequent is a tree sequent to CSGL proofs. Conversely, if we let arbitrary labeled sequent appear in CSGL proofs, it can be shown that 4L can be eliminated from CSGL proofs and replaced by instances of tr. These elimination results are proven syntactically, showing that proof transformations exist between CSGL and G3GL for proofs of tree sequents as summarized in the theorem below. We refer to the reader to [15] for the details.

▶ **Theorem 11** ([15]). *A tree sequent* T *is provable in* G3GL *iff it is provable in* CSGL*.*

$$\frac{}{\mathcal{G} /\!\!/ \Gamma, p \vdash p, \Delta}\,\mathsf{id}_1 \qquad \frac{}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Box\varphi, \Delta}\,\mathsf{id}_2 \qquad \frac{\mathcal{G} /\!\!/ \Gamma, \varphi \vdash \Delta \qquad \mathcal{G} /\!\!/ \Gamma, \psi \vdash \Delta}{\mathcal{G} /\!\!/ \Gamma, \varphi \vee \psi \vdash \Delta}\,\vee\mathsf{L}$$

$$\frac{\mathcal{G} /\!\!/ \Gamma \vdash \varphi, \psi, \Delta}{\mathcal{G} /\!\!/ \Gamma \vdash \varphi \vee \psi, \Delta}\,\vee\mathsf{R} \qquad \frac{\mathcal{G} /\!\!/ \Gamma \vdash \varphi, \Delta}{\mathcal{G} /\!\!/ \Gamma, \neg\varphi \vdash \Delta}\,\neg\mathsf{L} \qquad \frac{\mathcal{G} /\!\!/ \Gamma, \varphi \vdash \Delta}{\mathcal{G} /\!\!/ \Gamma \vdash \neg\varphi, \Delta}\,\neg\mathsf{R}$$

$$\frac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \Pi}\,4\mathsf{L} \qquad \frac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \Pi}\,\Box\mathsf{L} \qquad \frac{\mathcal{G} /\!\!/ \Gamma \vdash \Delta /\!\!/ \Box\varphi \vdash \varphi}{\mathcal{G} /\!\!/ \Gamma \vdash \Box\varphi, \Delta}\,\Box\mathsf{R}$$

Figure 4 Linear Nested Sequent Calculus LNGL for GL.

4 Linearizing Tree Sequents in Proofs

We now show how to extract a *linear nested sequent calculus* from CSGL, dubbed LNGL (see Figure 4). To the best of the author's knowledge, this is the first linear nested sequent calculus for Gödel-Löb provability logic. Linear nested sequents were introduced by Lellmann [21] and are a finite representation of Masini's 2-sequents [29]. Such systems operate over *lines* of Gentzen sequents and have been used to provide cut-free systems for intermediate and modal logics [20, 21, 29, 30].

The extraction of linear nested sequent proofs from tree sequent proofs takes place in three phases. In the first phase, we show how to transform any CSGL proof into an *end-active* proof, i.e., a proof such that principal and auxiliary formulae only occur at (pre-)leaves in tree sequents (cf. [21]). In the second phase, we define our novel linearization technique, where we identify specific paths in tree sequents and "prune" sub-trees, yielding a linear nested sequent proof as the result. This technique is an additional contribution of this paper, and we conjecture that this technique can be used in other settings to extract linear nested sequent systems from tree sequent/nested sequent systems. In the third phase, we show how to "reshuffle" a linear nested sequent proof so that the proof proceeds in repetitive stages of local rules, propagation rules, and $\Box$R rules, which we refer to as a proof in *normal form*. This transformation is motivated by one provided in [33] for so-called *basic nested systems*, which transforms proofs in a similar manner to extract Gentzen sequent proofs. Our transformation is distinct however as it works within the context of linear nested sequents. The simpler data structure used in linear nested sequents and the "end-active" shape of the rules in LNGL simplifies the process of reshuffling proofs into normal form.

A *linear nested sequent* is an expression of the form $\mathcal{G} := \Gamma_1 \vdash \Delta_1 /\!\!/ \cdots /\!\!/ \Gamma_n \vdash \Delta_n$ such that Γ_i and Δ_i are multisets of formulae from $\mathcal{L}$ for $1 \leq i \leq n$. We use $\mathcal{G}, \mathcal{H}, \ldots$ to denote linear nested sequents and note that such sequents admit a formula interpretation:

$$f(\Gamma \vdash \Delta) := \bigwedge \Gamma \to \bigvee \Delta \qquad f(\Gamma \vdash \Delta /\!\!/ \mathcal{G}) := \bigwedge \Gamma \to (\bigvee \Delta \vee \Box f(\mathcal{G}))$$

We define a linear nested sequent $\mathcal{G}$ to be (in)valid *iff* $f(\mathcal{G})$ is (in)valid. The linear nested sequent calculus LNGL consists of the rules shown in Figure 4. We take the $\neg$L, $\neg$R, $\vee$L, and $\vee$R rules to be *local rules* and the 4L and $\Box$L rules to be *propagation rules* in LNGL. For $1 \leq i \leq n$, we refer to $\Gamma_i \vdash \Delta_i$ as the i-component (or, as a *component* more generally) of the linear nested sequent $\mathcal{G} = \Gamma_1 \vdash \Delta_1 /\!\!/ \cdots /\!\!/ \Gamma_n \vdash \Delta_n$, we refer to $\Gamma_n \vdash \Delta_n$ as the *end component*, and we define the *length* of $\mathcal{G}$ to be $||\mathcal{G}|| := n$, i.e., the length of a linear nested sequent is equal to the number of its components. Comparing LNGL to CSGL, one can see that LNGL is the calculus CSGL restricted to lines of Gentzen sequents and where rules only operate in the last two components. Making use of the formula translation, it is a basic exercise to show that if the conclusion of any rule is invalid, then at least one premise is invalid, i.e., LNGL is sound.

▶ **Theorem 12.** *If $\mathcal{G}$ is provable in* LNGL*, then $\mathcal{G}$ is valid.*

When transforming CSGL proofs into LNGL proofs later on, it will be helpful to use the weakening rule w shown in the lemma below. Observe that any application of w to id_1 or id_2 yields an initial sequent, and w permutes above every other rule of LNGL. As an immediate consequence, we have that w is hp-admissible in LNGL.

▶ **Lemma 13.** *The following weakening rule* w *is hp-admissible in* LNGL*.*

$$\frac{\mathcal{G} /\!\!/ \Gamma \vdash \Delta /\!\!/ \mathcal{H}}{\mathcal{G} /\!\!/ \Gamma, \Sigma \vdash \Pi, \Delta /\!\!/ \mathcal{H}}\,\mathsf{w}$$

Furthermore, we have that the 4L and $\Box$L rules are hp-invertible in LNGL since the premises of each rule may be obtained from the conclusion by w.

▶ **Lemma 14.** *The* 4L *and* $\Box$L *rules are hp-invertble in* LNGL*.*

From Tree Sequents to Linear Nested Sequents

To extract LNGL from CSGL, we first establish a set of rule permutation results, i.e., we show that rules of a certain form in CSGL can always be permuted below other rules of a specific form. We note that a rule r *permutes below* a rule r' whenever an application of r followed by r' in a proof can be replaced by an application of r' (potentially preceded by an application of an i-inverse of r) followed by an application of r to derive the same conclusion. To make this definition more concrete, we show (1) the permutation of a unary rule r below a binary rule r' below top-left, (2) the permutation of a binary rule r below a unary rule r' below top-right, (3) the permutation of a unary rule r below a unary rule r' below bottom-left, and (4) the permutation of a binary rule r below a binary rule r' below bottom-right. In (1) and (4), we note that the case where r is applied to the right premise of r' is symmetric. Furthermore, recall that r_1^{-1} and r_2^{-1} are the 1- and 2-inverses of r, respectively. (NB. For the definition of the *i-inverse* of a rule, see Section 3.1.) We use (annotated versions of) the symbol S below to indicate not only tree sequents, but linear nested sequents since we consider permutations of rules in LNGL later on as well.

$$\frac{\dfrac{S_0}{S_2}\,\mathsf{r} \qquad S_1}{S_3}\,\mathsf{r}' \quad\rightsquigarrow\quad \frac{S_0 \qquad \dfrac{S_1}{S'}\,\mathsf{r}^{-1}}{\dfrac{S}{S_3}\,\mathsf{r}}\,\mathsf{r}' \qquad \frac{S_0 \qquad S_1}{\dfrac{S_2}{S_3}\,\mathsf{r}'}\,\mathsf{r} \quad\rightsquigarrow\quad \frac{\dfrac{S_0}{S}\,\mathsf{r}' \qquad \dfrac{S_1}{S'}\,\mathsf{r}'}{S_3}\,\mathsf{r}$$

$$\frac{\dfrac{S_0}{S_1}\,\mathsf{r}}{S_2}\,\mathsf{r}' \quad\rightsquigarrow\quad \frac{\dfrac{S_0}{S}\,\mathsf{r}'}{S_2}\,\mathsf{r} \qquad \frac{S_0 \qquad S_1}{\dfrac{S_2 \qquad\qquad S_3}{S_4}\,\mathsf{r}'}\,\mathsf{r} \quad\rightsquigarrow\quad \frac{\dfrac{S_0 \qquad \dfrac{S_3}{S}\,\mathsf{r}_1^{-1}}{S'}\,\mathsf{r}' \qquad \dfrac{S_1 \qquad \dfrac{S_3}{S''}\,\mathsf{r}_2^{-1}}{S'''}\,\mathsf{r}'}{S_4}\,\mathsf{r}$$

The various admissible rule permutations we describe are based on the notion of *end-activity*, which is a property of rule applications where principal and auxiliary formulae only occur at (pre-)leaves in sequents. End-activity was first discussed by Lellmann [21] in the context of mapping Gentzen sequent proofs into linear nested sequent proofs for non-classical logics.

▶ **Definition 15** (End Active)**.** *A* CSGL *proof is* end-active *iff the following hold:*

(1) *The principal label in every instance of* id_1 *and* id_2 *is a leaf;*

(2) *The principal label of each local rule is a leaf;*

(3) *The principal and auxiliary label of a propagation rule is a pre-leaf and leaf, respectively.*

A rule r *is* end-active *iff it satisfies its respective condition above; otherwise, the rule is* non-end-active. *We note that we always take the* $\Box\mathsf{R}$ *rule to be end-active.*

As stated in the following lemma, the end-activity of sequential rule applications determines a set of permutation relationships between the rules of CSGL. The lemma is straightforward to prove, though tedious due to the number of cases; its proof can be found in the appendix.

▶ **Lemma 16.** *The following permutations hold in* CSGL*:*

(1) *If* r *is a non-end-active local rule and* r' *is non-initial and end-active, then* r *permutes below* r' *and* r' *remains end-active after this permutation;*

(2) *if* r *is a non-end-active propagation rule and* r' *is non-initial and end-active, then* r *permutes below* r' *and* r' *remains end-active after this permutation.*

Let us define a *final-active* proof in CSGL to be a proof such that the last inference is end-active. Using the above lemma, every final-active proof π in CSGL can be transformed into an end-active proof as follows: first, observe that the last inference in π is end-active by assumption. By successively considering bottom-most instances of non-end-active local and propagation rules r in π, we can repeatedly apply Lemma 16 to permute r lower in the proof because all rules below r are guaranteed to be end-active. By inspecting the rules of CSGL, we know that the trees within the tree sequents in π will never "grow" and tend to "shrink" as they get closer to the conclusion of the proof, meaning, each non-end-active rule r will eventually become end-active through successive downward permutations.[2] This process will eventually terminate and yield a proof where all non-initial rules are end-active for the following two reasons: (1) As stated in the lemma above, permuting a non-end-active local or propagation rule r' below an end-active rule r preserves the end-activity of the rule r. (2) Although downward permutations may require the i-inverse of a rule to be applied above the permuted inferences, the hp-invertibility of all non-initial rules in CSGL (see Theorem 9) ensures that the height of the proof does not grow after a downward rule permutation.

After all such downward permutations have been performed, the resulting proof is *almost* end-active with the exception that initial rules may not be end-active. For example, as shown below left, it may be the case that id_1 is non-end-active and followed by a rule r. Since r is guaranteed to be end-active at this stage, we know that the auxiliary label of r is distinct from y, meaning, the conclusion will be an instance of id_1 as shown below right.

$$\dfrac{\dfrac{}{\mathcal{T}, \Gamma, y : p \vdash y : p, \Delta}\,\mathsf{id}_1 \qquad (\mathcal{T}', \Gamma', y : p \vdash y : p, \Delta')}{\mathcal{T}'', \Gamma'', y : p \vdash y : p, \Delta''}\,\mathsf{r} \qquad \dfrac{}{\mathcal{T}'', \Gamma'', y : p \vdash y : p, \Delta''}\,\mathsf{id}_1$$

By replacing such rule applications r by id_1 (or id_2) instances, effectively "pushing" initial rules down in the proof, we will eventually obtain initial rules such that the label y is auxiliary in the subsequent rule application, which will then be a leaf since all non-initial rules of the proof are end-active. Thus, every final-active proof in CSGL can be transformed into an end-active proof.

▶ Remark 17. As a corollary, we note that every proof π in CSGL of a sequent $\vdash x : \varphi$ can be transformed into an end-active proof as well. This is due to the fact that the last inference in π must be end-active since the proof ends with $\vdash x : \varphi$, i.e., π is final-active in this case.

[2] Observe that $\neg\mathsf{L}$, $\neg\mathsf{R}$, $\vee\mathsf{L}$, $\vee\mathsf{R}$, $\Box\mathsf{L}$, and $\mathsf{4L}$ only affect the formulae associated with the label of a tree sequent, whereas $\Box\mathsf{R}$ top-down removes a relational atom from a tree sequent. Therefore, the number of relational atoms in tree sequents never increases as we move down paths in CSGL proofs from initial sequents to the conclusion.

▶ **Theorem 18.** *Each final-active proof in* CSGL *can be transformed into an end-active proof.*

▶ **Theorem 19.** *Each end-active proof in* CSGL *can be transformed into a proof in* LNGL.

Proof. We prove that if there exists an end-active proof in CSGL of a tree sequent $\mathcal{T}, \Gamma \vdash \Delta$, then there exists a path $x_1, \ldots, x_n$ of labels from the root x_1 to a leaf x_n in $\mathcal{T}, \Gamma \vdash \Delta$ such that $\Gamma(x_1) \vdash \Delta(x_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(x_n) \vdash \Delta(x_n)$ is provable in LNGL. We argue this by induction on the height of the end-active proof π in CSGL.

Base case. Suppose π consists of a single application of id_1 or id_2, as shown below:

$$\frac{}{\mathcal{T}, \Gamma, x : p \vdash x : p, \Delta}\ \mathsf{id}_1 \qquad \frac{}{\mathcal{T}, \Gamma, x : \Box\varphi \vdash x : \Box\varphi, \Delta}\ \mathsf{id}_2$$

We know that each initial sequent is end-active, i.e., the label x is a leaf in both tree sequents. Therefore, since each sequent is a tree sequent, there exists a path $y_1, \ldots, y_n = x$ of labels from the root y_1 to the leaf x. By using this path, we obtain respective instances of id_1 and id_2 as shown below:

$$\frac{}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(x), p \vdash p, \Delta(x)}\ \mathsf{id}_1 \qquad \frac{}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(x), \Box\varphi \vdash \Box\varphi, \Delta(x)}\ \mathsf{id}_2$$

Inductive step. For the inductive hypothesis (IH), we assume that the claim holds for every end-active proof in CSGL of height $h' \leq h$, and aim to show that the claim holds for proofs of height $h+1$. We let π be of height $h+1$ and argue the cases where π ends with $\Box\mathsf{L}$ or $\Box\mathsf{R}$ as the remaining cases are shown similarly. Some additional cases are given in the appendix.

$\Box$L. Let us suppose that π ends with an instance of $\Box\mathsf{L}$ as shown below.

$$\frac{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \varphi \vdash \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash \Delta}\ \Box\mathsf{L}$$

By IH, we know there exists a path $y_1, \ldots, y_n$ of labels from the root y_1 to the leaf y_n in the premise of $\Box\mathsf{L}$ such that $\mathcal{G} = \Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)$ is provable in LNGL. Since $\Box\mathsf{L}$ is end-active, we have three cases to consider: (1) neither x nor y occur along the path in the premise, (2) only x occurs along the path in the premise, or (3) both x and y occur along the path in the premise. In cases (1) and (2), we translate the entire $\Box\mathsf{L}$ inference as the linear nested sequent $\mathcal{G}$. In case (3), $\mathcal{G}$ has the form of the premise shown below with $\Gamma(y_{n-1}) = \Sigma_1, \Box\varphi$ and $\Gamma(y_n) = \Sigma_2, \varphi$. A single application of $\Box\mathsf{L}$ gives the desired result.

$$\frac{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma_1, \Box\varphi \vdash \Delta(y_{n-1}) \mathbin{/\!\!/} \Sigma_2, \varphi \vdash \Delta(y_n)}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma_1, \Box\varphi \vdash \Delta(y_{n-1}) \mathbin{/\!\!/} \Sigma_2 \vdash \Delta(y_n)}\ \Box\mathsf{L}$$

$\Box$R. Let us suppose that π ends with an instance of $\Box\mathsf{R}$ as shown below.

$$\frac{\mathcal{T}, xRy, \Gamma, y : \Box\varphi \vdash y : \varphi, \Delta}{\mathcal{T}, \Gamma \vdash x : \Box\varphi, \Delta}\ \Box\mathsf{R}$$

By IH, we know there exists a path $y_1, \ldots, y_n$ of labels from the root y_1 to the leaf y_n in the premise of $\Box\mathsf{R}$ such that $\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)$ is provable in LNGL. We have three cases to consider: either (1) neither x nor y occur along the path, (2) only x occurs along the path, or (3) both x and y occur along the path. In case (1), we translate the entire $\Box\mathsf{R}$ instance as the single linear nested sequent $\mathcal{G}$. In case (2), we know that $x = y_i$ for some $1 \leq i \leq n$. To obtain the desired conclusion, we apply the hp-admissible w rule as shown

below. Observe that the conclusion of the w application below corresponds to the linear nested sequent obtained from the path $y_1, \ldots, y_n$ in the conclusion of the $\Box$R instance above.

$$\dfrac{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_i) \vdash \Delta(y_i) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_i) \vdash \Box\varphi, \Delta(y_i) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)}\,\mathsf{w}$$

Last, in case (3), we know that $x = y_{n-1}$ and $y = y_n$ due to the freshness condition imposed on the $\Box$R rule. In this case, $\mathcal{G}$ has the form of the premise shown below, meaning, a single application of the $\Box$R rule gives the linear nested sequent corresponding to the path $y_1, \ldots, y_n$ in the conclusion of the $\Box$R instance above.

$$\dfrac{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_{n-1}) \vdash \Delta(y_{n-1}) \mathbin{/\!\!/} \Box\varphi \vdash \varphi}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_{n-1}) \vdash \Box\varphi, \Delta(y_{n-1})}\,\Box\mathsf{R}$$ ◀

The following is an immediate consequence of Theorems 12, 18, 19 and Remark 17.

▶ **Corollary 20** (LNGL Soundness and Completeness). *φ is valid iff $\vdash \varphi$ is provable in* LNGL.

Last, we show that every LNGL proof can be put into a *normal form* (see Definition 21 and Theorem 18 below) such that (reading the proof bottom-up) $\Box$R instances are preceded by 4L instances, which are preceded by $\Box$L instances, which are preceded by local rule instances (or, initial rules). We will utilize this normal form in the next section to show that every LNGL proof can be transformed into a Gentzen sequent proof (Theorem 24). We let B be a set of LNGL rules and define a *block* to be a derivation that only uses rules from B. We use the following notation to denote blocks, showing that the set B of rules derives $\mathcal{G}$ from $\mathcal{G}_1, \ldots, \mathcal{G}_n$, and refer to $\mathcal{G}_1, \ldots, \mathcal{G}_n$ as the *premises* of the block B.

$$\dfrac{\mathcal{G}_1, \ldots, \mathcal{G}_n}{\mathcal{G}}\,\mathrm{B}$$

▶ **Definition 21** (Normal Form). *A proof in* LNGL *is in* normal form *iff each bottom-up $\Box$R application is derived from a block* B *of* 4L *rules, whose premise is derived from a block* B′ *of* $\Box$L *rules, whose premise is derived from a block* B″ *of local rules, as indicated below.*

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Sigma_1 \vdash \Pi_1 \quad \ldots \quad \mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Sigma_n \vdash \Pi_n}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Gamma', \Gamma'', \Box\varphi \vdash \varphi}\,\mathrm{B''}}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Gamma', \Box\varphi \vdash \varphi}\,\mathrm{B'}}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Box\varphi \vdash \varphi}\,\mathrm{B}}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Box\varphi, \Delta}\,\Box\mathsf{R}$$

We refer to block of rules of the above form as a complete block*, and refer to the portion of a complete block consisting of only* $\Box$R, B, *and* B′ *as a* modal block.

As proven in the next section (Theorem 24), every normal form proof in LNGL can be transformed into a proof in Sambin and Valentini's Gentzen calculus $\mathsf{GL}_{\mathsf{seq}}$. Therefore, we need to show that every proof in LNGL can be put into normal form. We prove this by making an observation about the structure of proofs in LNGL. Observe that local and propagation rules in LNGL only affect the end component of linear nested sequents and preserve the length of such sequents, whereas the $\Box$R rule increases the length of a linear nested sequent by 1 when applied bottom-up. This implies that any LNGL proof π bottom-up proceeds in repetitive stages, as we now describe. Let π be a proof in LNGL with conclusion $\mathcal{G}$ such that $||\mathcal{G}|| = n$. The conclusion $\mathcal{G}$ is derived with a block B of local and propagation rules that only affect the n-component in inferences with the premises of the block B being initial rules or derived by applications of $\Box$R rules. These applications of $\Box$R rules will have premises

$$\frac{}{\Gamma, \varphi \vdash \varphi, \Delta}\,\mathsf{id} \qquad \frac{\Gamma \vdash \varphi, \Delta}{\Gamma, \neg\varphi \vdash \Delta}\,\neg\mathsf{L} \qquad \frac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \neg\varphi, \Delta}\,\neg\mathsf{R} \qquad \frac{\Gamma, \varphi \vdash \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta}\,\vee\mathsf{L}$$

$$\frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta}\,\vee\mathsf{R} \qquad \frac{\Box\Gamma, \Gamma, \Box\varphi \vdash \varphi}{\Sigma, \Box\Gamma \vdash \Box\varphi, \Delta}\,\Box_{\mathsf{GL}} \qquad \frac{\Box\Gamma, \Gamma \vdash \varphi}{\Sigma, \Box\Gamma \vdash \Box\varphi, \Delta}\,\Box_4$$

Figure 5 Sequent calculus rules.

of length $n+1$ and will be preceded by blocks B_i of local and propagation rules that only affect the $(n+1)$-component in inferences. The premises of the B_i blocks will then either be initial rules or derived by applications of $\Box\mathsf{R}$ rules that have premises of length $n+2$, which are preceded by blocks of local and propagation rules that only affect the $(n+2)$-component in inferences, and so on. Every proof in LNGL will have this repetitive structure.

Let $\Box\mathsf{R}$ be applied in an LNGL proof π with premise $\mathcal{G} /\!\!/ \Gamma \vdash \Delta /\!\!/ \Box\varphi \vdash \varphi$ of length n. We say that an instance of a local or propagation rule r in π is *length-consistent* with $\Box\mathsf{R}$ *iff* the length of the conclusion of r is equal to n. Based on the discussion above, we can see that for any $\Box\mathsf{R}$ application in a proof π, all length-consistent local and propagation rules will occur in a block B above the $\Box\mathsf{R}$ application with B free of other $\Box\mathsf{R}$ rules. It is not difficult to show that B can be transformed into a *complete block* by (1) successively permuting $\mathsf{4L}$ rules down into a block above $\Box\mathsf{R}$, and (2) successively permuting $\Box\mathsf{L}$ rules down above the $\mathsf{4L}$ block. After the permutations from (1) and (2) have been carried out, the premise of the $\Box\mathsf{L}$ block will be derived by length-consistent local rule applications, showing that $\Box\mathsf{R}$ is preceded by a complete block. As these permutations can be performed for every $\Box\mathsf{R}$ rule in a proof, every proof can be put into normal form.

▶ **Theorem 22.** *Every proof in* LNGL *can be transformed into a proof in normal form.*

5 Sequent Systems and Correspondences

5.1 Gentzen, Cyclic, and Non-Wellfounded Systems

We use Γ, Δ, Σ, ... to denote finite multisets of formulae within the context of sequent systems. For a multiset $\Gamma := \varphi_1, \ldots, \varphi_n$, we define $\Box\Gamma := \Box\varphi_1, \ldots, \Box\varphi_n$. A *sequent* is defined to be an expression of the form $\Gamma \vdash \Delta$. The sequent calculus $\mathsf{GL}_{\mathsf{seq}}$ for GL consists of the rules id, $\neg\mathsf{L}$, $\neg\mathsf{R}$, $\vee\mathsf{L}$, $\vee\mathsf{R}$, and $\Box_{\mathsf{GL}}$ shown in Figure 5 and is an equivalent variant of the Gentzen calculi GLSC and GLS introduced by Sambin and Valentini for GL [36, 37].[3] The system $\mathsf{GL}_{\mathsf{seq}}$ is sound and complete for GL, admits syntactic cut-elimination, and the weakening and contraction rules w, cL, and cR (shown below) are admissible (cf. [16, 39]).

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Sigma \vdash \Pi, \Delta}\,\mathsf{w} \qquad \frac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi \vdash \Delta}\,\mathsf{cL} \qquad \frac{\Gamma \vdash \varphi, \varphi, \Delta}{\Gamma \vdash \varphi, \Delta}\,\mathsf{cR} \qquad \frac{\Gamma \vdash \varphi, \Delta \qquad \Gamma, \varphi \vdash \Delta}{\Gamma \vdash \Delta}\,\mathsf{cut}$$

Shamkanov [39] showed that equivalent non-wellfounded and cyclic sequent systems could be obtained for GL by taking the sequent calculus for the modal logic $\mathsf{K4}$ and generalizing the notion of proof. The sequent calculus $\mathsf{K4}_{\mathsf{seq}}$ is obtained by replacing the $\Box_{\mathsf{GL}}$ rule in $\mathsf{GL}_{\mathsf{seq}}$ with the $\Box_4$ rule shown in Figure 5. Let us now recall Shamkanov's non-wellfounded sequent calculus GL_∞ and cyclic sequent calculus $\mathsf{GL}_{\mathsf{circ}}$ for GL. We present Shamkanov's systems in

[3] $\mathsf{GL}_{\mathsf{seq}}$ differs from Sambin and Valentini's original systems in that multisets are used instead of sets, rules for superfluous logical connectives (e.g., conjunction $\wedge$ and implication $\rightarrow$) have been omitted as these are definable in terms of other rules, and the weakening rules have been absorbed into id and $\Box_{\mathsf{GL}}$.

a *two-sided format*, i.e., using two-sided sequents $\Gamma \vdash \Delta$ rather than one-sided sequents of the form Γ. This makes the correspondence between Shamkanov's systems and $\mathsf{GL}_{\mathsf{seq}}$ clearer as well as saves us from having to introduce a new language for GL since one-sided sequents use formulae in negation normal form. Translating proofs with two-sided sequents to proofs with one-sided sequents and vice-versa can be easily obtained by standard techniques, and so, this minor modification causes no problems.

A *derivation* of a sequent $\Gamma \vdash \Delta$ is defined to be a (potentially infinite) tree whose nodes are labeled with sequents such that (1) $\Gamma \vdash \Delta$ is the root of the tree, and (2) each parent node is taken to be the conclusion of a rule in $\mathsf{K4}_{\mathsf{seq}}$ with its children the corresponding premises. A *non-wellfounded proof* is a derivation such that all leaves are initial sequents. GL_∞ is the non-wellfounded sequent system obtained by letting the set of provable sequents be determined by non-wellfounded proofs.

A *cyclic derivation* is a pair $\pi = (\kappa, c)$ such that κ is a finite derivation in $\mathsf{K4}_{\mathsf{seq}}$ and c is a function with the following properties: (1) c is defined on a subset of the leaves of κ, (2) the image $c(x)$ lies on the path from the root of κ to x and does not coincide with x, and (3) both x and $c(x)$ are labeled by the same sequent. If the function c is defined at a leaf x, then we say that a *back-link* exists from x to $c(x)$. A *cyclic proof* is a cyclic derivation $\pi = (\kappa, c)$ such that every leaf x is labeled by an instance of id or there exists a back-link from x to the node $c(x)$. $\mathsf{GL}_{\mathsf{circ}}$ is the cyclic sequent system obtained by letting the set of provable sequents be determined by cyclic proofs.

Shamkanov established a three-way correspondence between $\mathsf{GL}_{\mathsf{seq}}$, GL_∞, and $\mathsf{GL}_{\mathsf{circ}}$, providing syntactic transformations mapping proofs between the three systems.[4]

▶ **Theorem 23** ([39]). *$\Gamma \vdash \Delta$ is provable in $\mathsf{GL}_{\mathsf{seq}}$ iff $\Gamma \vdash \Delta$ is provable in GL_∞ iff $\Gamma \vdash \Delta$ is provable in $\mathsf{GL}_{\mathsf{circ}}$.*

5.2 Completing the Correspondences

▶ **Theorem 24.** *If π is a normal form proof of $\vdash \varphi$ in LNGL, then π can be transformed into a proof of $\vdash \varphi$ in $\mathsf{GL}_{\mathsf{seq}}$.*

Proof. We show how to transform the normal form proof π of $\vdash \varphi$ in LNGL into a proof π' of $\vdash \varphi$ in $\mathsf{GL}_{\mathsf{seq}}$ in a bottom-up manner. For the conclusion $\vdash \varphi$ of the proof π, we take $\vdash \varphi$ to be the conclusion of π'. We now make a case distinction on bottom-up applications of rules applied in π. For each rule $\neg\mathsf{L}$, $\neg\mathsf{R}$, $\vee\mathsf{L}$, or $\vee\mathsf{R}$, we translate each premise of the rule as its end component. For example, the $\vee\mathsf{L}$ rule will be translated as shown below.

$$\frac{\mathcal{G} /\!\!/ \Gamma, \varphi \vdash \Delta \qquad \mathcal{G} /\!\!/ \Gamma, \psi \vdash \Delta}{\mathcal{G} /\!\!/ \Gamma, \varphi \vee \psi \vdash \Delta} \vee\mathsf{L} \qquad \frac{\Gamma, \varphi \vdash \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta} \vee\mathsf{L}$$

Suppose now that we encounter a $\Box\mathsf{R}$ rule while bottom-up translating the proof π into a proof in $\mathsf{GL}_{\mathsf{seq}}$. Since π is in normal form, we know that $\Box\mathsf{R}$ is preceded by a modal block (see Definition 21), that is, $\Box\mathsf{R}$ is (bottom-up) preceded by a block B_{4L} of $\mathsf{4L}$ rules, which is preceded by a block $\mathrm{B}_{\Box\mathsf{L}}$ of $\Box\mathsf{L}$ rules, i.e., the modal block has the shape shown below. We suppose that $\Box\Sigma_1$ are the principal formulae of the $\mathsf{4L}$ applications, $\Box\Sigma_2$ are those formulae principal in both $\mathsf{4L}$ and $\Box\mathsf{L}$ applications, and $\Box\Sigma_3$ are those formulae principal only in $\Box\mathsf{L}$ applications.

[4] We note that Shamkanov's proof transformation from $\mathsf{GL}_{\mathsf{seq}}$ to GL_∞ relies on the admissibility of the cut rule in $\mathsf{GL}_{\mathsf{seq}}$. This is not problematic however since $\mathsf{GL}_{\mathsf{seq}}$ admits syntactic cut-elimination.

$$\dfrac{\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3 \vdash \Delta /\!\!/ \Box\Sigma_1, \Box\Sigma_2, \Sigma_2, \Sigma_3, \Box\varphi \vdash \varphi}{\mathcal{G} /\!\!/ \Gamma, \Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3 \vdash \Delta /\!\!/ \Box\Sigma_1, \Box\Sigma_2, \Box\varphi \vdash \varphi}\,\mathrm{B}_{\Box\mathsf{L}}}{\mathcal{G} /\!\!/ \Gamma, \Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3 \vdash \Delta /\!\!/ \Box\varphi \vdash \varphi}\,\mathrm{B}_{\mathsf{4L}}}{\mathcal{G} /\!\!/ \Gamma, \Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3 \vdash \Box\varphi, \Delta}\,\Box\mathsf{R}$$

We bottom-up translate the entire block as shown below, where the conclusion is obtained from the end component of the modal block's conclusion. Note that we may apply the w rule because it is admissible in $\mathsf{GL}_{\mathsf{seq}}$.

$$\dfrac{\dfrac{\Box\Sigma_1, \Box\Sigma_2, \Sigma_2, \Sigma_3, \Box\varphi \vdash \varphi}{\Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3, \Sigma_1, \Sigma_2, \Sigma_3, \Box\varphi \vdash \varphi}\,\mathsf{w}}{\Gamma, \Box\Sigma_1, \Box\Sigma_2, \Box\Sigma_3 \vdash \Box\varphi, \Delta}\,\Box_{\mathsf{GL}}$$

Last, suppose an instance of id_1 or id_2 is reached in the translation as shown below left.

$$\dfrac{}{\mathcal{G} /\!\!/ \Gamma, p \vdash p, \Delta}\,\mathsf{id}_1 \qquad \dfrac{}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Box\varphi, \Delta}\,\mathsf{id}_2 \qquad \dfrac{}{\Gamma, p \vdash p, \Delta}\,\mathsf{id} \qquad \dfrac{}{\Gamma, \Box\varphi \vdash \Box\varphi, \Delta}\,\mathsf{id}$$

In each case, we translate the linear nested sequent as its end component, yielding the respective Gentzen sequents shown above right, both of which are instances of id. ◀

Last, the following theorem completes the circuit of proof transformations and establishes syntactic correspondences between $\mathsf{G3GL}$, CSGL, LNGL, $\mathsf{GL}_{\mathsf{seq}}$, GL_{∞} and $\mathsf{GL}_{\mathsf{circ}}$.

▶ **Theorem 25.** *If* $\Gamma \vdash \Delta$ *is provable in* $\mathsf{GL}_{\mathsf{seq}}$*, then* $x : \Gamma \vdash x : \Delta$ *is provable in* $\mathsf{G3GL}$*.*

Proof. By induction on the height of the proof π in $\mathsf{GL}_{\mathsf{seq}}$. The base case immediately follows from Theorem 6-(1), and the $\neg\mathsf{L}$, $\neg\mathsf{R}$, $\vee\mathsf{L}$, and $\vee\mathsf{R}$ cases of the inductive step straightforwardly follow by applying IH and then the corresponding rule in $\mathsf{G3GL}$. Therefore, we need only show the case where π ends with an application of $\Box_{\mathsf{GL}}$, as shown below left.

$$\dfrac{\Box\Gamma, \Gamma, \Box\varphi \vdash \varphi}{\Sigma, \Box\Gamma \vdash \Box\varphi, \Delta}\,\Box_{\mathsf{GL}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : \Box\Gamma, x : \Gamma, x : \Box\varphi \vdash x : \varphi}{yRx, y : \Box\Gamma, x : \Box\Gamma, x : \Gamma, x : \Box\varphi \vdash x : \varphi}\,\mathsf{w}}{yRx, y : \Box\Gamma, x : \Box\Gamma, x : \Box\varphi \vdash x : \varphi}\,\Box\mathsf{L}}{yRx, y : \Box\Gamma, x : \Box\varphi \vdash x : \varphi}\,\mathsf{4L}}{y : \Box\Gamma \vdash y : \Box\varphi}\,\Box\mathsf{R}}{x : \Box\Gamma \vdash x : \Box\varphi}\,(x/y)}{x : \Sigma, x : \Box\Gamma \vdash x : \Box\varphi, x : \Delta}\,\mathsf{w}$$

To obtain the desired proof, we first apply the hp-admissible w rule (Theorem 6), followed by applications of the $\Box\mathsf{L}$ rule and admissible $\mathsf{4L}$ rule (cf. [15]). Applying the $\Box\mathsf{R}$ rule, followed by applications of the hp-admissible (x/y) and w rules (Theorem 6), gives the desired conclusion. ◀

6 Concluding Remarks

There are various avenues for future research: first, it would be interesting to look into the properties of the new linear nested sequent calculus LNGL, investigating additional admissible structural rules, how the system can be amended to allow for the hp-invertibility of all rules, and also looking into syntactic cut-elimination. Second, by employing a methodology for extracting nested sequent systems from relational semantics [28], we can integrate this approach with the linearization technique to develop a general method for extracting (cut-free)

linear nested systems from the semantics of various modal, intuitionistic, and related logics. Third, it seems worthwhile to see if the proof transformation techniques discussed in this paper can be applied to structural cyclic systems (e.g., cyclic labeled sequent systems for classical and intuitionistic Gödel-Löb logic [11]) to remove extraneous structure and extract simpler (cyclic) Gentzen systems.

References

1 Bahareh Afshari and Graham E. Leigh. Cut-free completeness for modal mu-calculus. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017. doi:10.1109/LICS.2017.8005088.

2 Arnon Avron. On modal systems having arithmetical interpretations. *Journal of Symbolic Logic*, 49(3):935–942, 1984. doi:10.2307/2274147.

3 James Brotherston. Cyclic proofs for first-order logic with inductive definitions. In Bernhard Beckert, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 78–92, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. doi:10.1007/11554554_8.

4 James Brotherston. Formalised inductive reasoning in the logic of bunched implications. In Hanne Riis Nielson and Gilberto Filé, editors, *Static Analysis*, pages 87–103, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-74061-2_6.

5 James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, October 2010. doi:10.1093/logcom/exq052.

6 Kai Brünnler. Deep sequent systems for modal logic. *Archive for Mathematical Logic*, 48(6):551–577, 2009. doi:10.1007/s00153-009-0137-3.

7 Robert A. Bull. Cut elimination for propositional dynamic logic without *. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 38(2):85–100, 1992. doi:10.1002/MALQ.19920380107.

8 Marcos A Castilho, Luis Farinas del Cerro, Olivier Gasquet, and Andreas Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3, 4):281–297, 1997. doi:10.3233/FI-1997-323404.

9 Agata Ciabattoni, Tim Lyon, Revantha Ramanayake, and Alwen Tiu. Display to labelled proofs and back again for tense logics. *ACM Transactions on Computational Logic*, 22(3):1–31, 2021. doi:10.1145/3460492.

10 Anupam Das and Marianna Girlando. Cyclic hypersequent system for transitive closure logic. *Journal of Automated Reasoning*, 67(3):27, 2023. doi:10.1007/s10817-023-09675-1.

11 Anupam Das, Iris van der Giessen, and Sonia Marin. Intuitionistic Gödel-Löb Logic, à la Simpson: Labelled Systems and Birelational Semantics. In Aniello Murano and Alexandra Silva, editors, *32nd EACSL Annual Conference on Computer Science Logic (CSL 2024)*, volume 288 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:18, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2024.22.

12 Melvin Fitting. Tableau methods of proof for modal logics. *Notre Dame Journal of Formal Logic*, 13(2):237–247, 1972. doi:10.1305/NDJFL/1093894722.

13 Melvin Fitting. Nested sequents for intuitionistic logics. *Notre Dame Journal of Formal Logic*, 55(1):41–61, 2014. doi:10.1215/00294527-2377869.

14 Rajeev Goré, Linda Postniece, and Alwen Tiu. Cut-elimination and proof-search for bi-intuitionistic logic using nested sequents. In Carlos Areces and Robert Goldblatt, editors, *Advances in Modal Logic 7*, pages 43–66. College Publications, 2008. URL: http://www.aiml.net/volumes/volume7/Gore-Postniece-Tiu.pdf.

15 Rajeev Goré and Revantha Ramanayake. Labelled tree sequents, tree hypersequents and nested (deep) sequents. In Thomas Bolander, Torben Braüner, Silvio Ghilardi, and Lawrence S. Moss, editors, *Advances in Modal Logic 9*, pages 279–299. College Publications, 2012. URL: http://www.aiml.net/volumes/volume9/Gore-Ramanayake.pdf.

16 Rajeev Goré and Revantha Ramanayake. Valentini's cut-elimination for provability logic resolved. *The Review of Symbolic Logic*, 5(2):212–238, 2012. doi:10.1017/S1755020311000323.

17 Ryo Ishigaki and Kentaro Kikuchi. Tree-sequent methods for subintuitionistic predicate logics. In Nicola Olivetti, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 4548 of *Lecture Notes in Computer Science*, pages 149–164, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-73099-6_13.

18 Stig Kanger. *Provability in logic*. Almqvist & Wiksell, 1957.

19 Ryo Kashima. Cut-free sequent calculi for some tense logics. *Studia Logica*, 53(1):119–135, 1994. doi:10.1007/BF01053026.

20 Roman Kuznets and Björn Lellmann. Interpolation for intermediate logics via hyper- and linear nested sequents. In Guram Bezhanishvili, Giovanna D'Agostino, George Metcalfe, and Thomas Studer, editors, *Advances in Modal Logic 12*, pages 473–492. College Publications, 2018. URL: http://www.aiml.net/volumes/volume12/Kuznets-Lellmann.pdf.

21 Björn Lellmann. Linear nested sequents, 2-sequents and hypersequents. In Hans De Nivelle, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 9323 of *Lecture Notes in Computer Science*, pages 135–150, Cham, 2015. Springer International Publishing. doi:10.1007/978-3-319-24312-2_10.

22 Tim Lyon. On the correspondence between nested calculi and semantic systems for intuitionistic logics. *Journal of Logic and Computation*, 31(1):213–265, December 2020. doi:10.1093/logcom/exaa078.

23 Tim Lyon. *Refining labelled systems for modal and constructive logics with applications*. PhD thesis, TU Wien, 2021. URL: https://arxiv.org/abs/2107.14487, arXiv:2107.14487.

24 Tim Lyon, Alwen Tiu, Rajeev Goré, and Ranald Clouston. Syntactic interpolation for tense logics and bi-intuitionistic logic via nested sequents. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*, volume 152 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2020.28.

25 Tim S. Lyon. Nested sequents for intuitionistic modal logics via structural refinement. In Anupam Das and Sara Negri, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 409–427, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-86059-2_24.

26 Tim S. Lyon, Agata Ciabattoni, Didier Galmiche, Dominique Larchey-Wendling, Daniel Méry, Nicola Olivetti, and Revantha Ramanayake. Internal and external calculi: Ordering the jungle without being lost in translations. *Found on arXiv*, 2023. URL: https://arxiv.org/abs/2312.03426, doi:10.48550/arXiv.2312.03426.

27 Tim S. Lyon and Eugenio Orlandelli. Nested sequents for quantified modal logics. In Revantha Ramanayake and Josef Urban, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 449–467, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-43513-3_24.

28 Tim S. Lyon and Piotr Ostropolski-Nalewaja. Foundations for an abstract proof theory in the context of horn rules. *Found on arXiv*, 2024. URL: https://arxiv.org/abs/2304.05697, doi:10.48550/arXiv.2304.05697.

29 Andrea Masini. 2-sequent calculus: A proof theory of modalities. *Annals of Pure and Applied Logic*, 58(3):229–246, 1992. doi:10.1016/0168-0072(92)90029-Y.

30 Andrea Masini. 2-sequent calculus: Intuitionism and natural deduction. *Journal of Logic and Computation*, 3(5):533–562, 1993. doi:10.1093/LOGCOM/3.5.533.

31 Sara Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34(5):507–544, 2005. doi:10.1007/s10992-005-2267-3.

32 Damian Niwiński and Igor Walukiewicz. Games for the μ-calculus. *Theoretical Computer Science*, 163(1):99–116, 1996. doi:10.1016/0304-3975(95)00136-0.

33 Elaine Pimentel, Revantha Ramanayake, and Björn Lellmann. Sequentialising nested systems. In Serenella Cerrito and Andrei Popescu, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 11714 of *Lecture Notes in Computer Science*, pages 147–165, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-29026-9_9.

34 Francesca Poggiolesi. The method of tree-hypersequents for modal propositional logic. In David Makinson, Jacek Malinowski, and Heinrich Wansing, editors, *Towards Mathematical Philosophy*, volume 28 of *Trends in logic*, pages 31–51. Springer, 2009. doi:10.1007/978-1-4020-9084-4_3.

35 Francesca Poggiolesi. A purely syntactic and cut-free sequent calculus for the modal logic of provability. *The Review of Symbolic Logic*, 2(4):593–611, 2009. doi:10.1017/S1755020309990244.

36 G. Sambin and S. Valentini. A modal sequent calculus for a fragment of arithmetic. *Studia Logica: An International Journal for Symbolic Logic*, 39(2/3):245–256, 1980. URL: http://www.jstor.org/stable/20014984.

37 Giovanni Sambin and Silvio Valentini. The modal logic of provability. the sequential approach. *Journal of Philosophical Logic*, 11(3):311–342, 1982. URL: http://www.jstor.org/stable/30226252, doi:10.1007/BF00293433.

38 Krister Segerberg. *An Essay in Classical Modal Logic*. Uppsala: Filosofiska Föreningen och Filosofiska Institutionen vid Uppsala Universitet, 1971.

39 D. S. Shamkanov. Circular proofs for the Gödel-Löb provability logic. *Mathematical Notes*, 96(3):575–585, 2014. doi:10.1134/S0001434614090326.

40 Alex K Simpson. *The proof theory and semantics of intuitionistic modal logic*. PhD thesis, University of Edinburgh. College of Science and Engineering. School of Informatics, 1994.

41 Robert M. Solovay. Provability interpretations of modal logic. *Israel Journal of Mathematics*, 25(3):287–304, 1976. doi:10.1007/BF02757006.

42 Lutz Straßburger. Cut elimination in nested sequents for intuitionistic modal logics. In Frank Pfenning, editor, *Foundations of Software Science and Computation Structures*, volume 7794 of *Lecture Notes in Computer Science*, pages 209–224, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. doi:10.1007/978-3-642-37075-5_14.

43 Luca Viganò. *Labelled Non-Classical Logics*. Springer Science & Business Media, 2000.

A Proofs for Section 4

▶ **Lemma 16.** *The following permutations hold in* GSL*:*

(1) *If* r *is a non-end-active local rule and* r' *is non-initial and end-active, then* r *permutes below* r' *and* r' *remains end-active after this permutation;*

(2) *if* r *is a non-end-active propagation rule and* r' *is non-initial and end-active, then* r *permutes below* r' *and* r' *remains end-active after this permutation.*

Proof. Follows from Lemmas 26 and 27 below. ◀

▶ **Lemma 26.** *If* r *is a non-end-active local rule and* r' *is non-initial and end-active, then* r *permutes below* r' *and* r' *remains end-active after this permutation.*

Proof. We let r be an instance of $\neg\mathsf{R}$ as the cases where r is either $\neg\mathsf{L}$, $\vee\mathsf{L}$, or $\vee\mathsf{R}$ are shown similarly. We show that r can be permuted down r' and consider a representative number of cases when r' is either $\vee\mathsf{R}$, $4\mathsf{L}$, or $\Box\mathsf{R}$ as the remaining cases are similar.

$\vee$R. By our assumption that $\neg\mathsf{R}$ is non-end-active and $\vee\mathsf{R}$ is end-active, we know that the labels x and y are distinct. Hence, we can permute the $\neg\mathsf{R}$ instance below the $\vee\mathsf{R}$ instance. Observe that $\vee\mathsf{R}$ remains end-active after the permutation.

$$\dfrac{\dfrac{\mathcal{T}, \Gamma, x : \varphi \vdash y : \psi, y : \chi, \Delta}{\mathcal{T}, \Gamma \vdash x : \neg\varphi, y : \psi, y : \chi, \Delta}\,\neg\mathsf{R}}{\mathcal{T}, \Gamma \vdash x : \neg\varphi, y : \psi \vee \chi, \Delta}\,\vee\mathsf{R} \qquad \dfrac{\dfrac{\mathcal{T}, \Gamma, x : \varphi \vdash y : \psi, y : \chi, \Delta}{\mathcal{T}, \Gamma, x : \varphi \vdash y : \psi \vee \chi, \Delta}\,\vee\mathsf{R}}{\mathcal{T}, \Gamma \vdash x : \neg\varphi, y : \psi \vee \chi, \Delta}\,\neg\mathsf{R}$$

4L. By our assumption, we know that z is distinct from y in the inferences shown below left, meaning, we can permute ¬R below 4L as shown below right. Observe that 4L remains end-active after the permutation.

$$\dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, x : \Box\psi, y : \Box\psi, z : \varphi \vdash \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\psi, y : \Box\psi \vdash z : \neg\varphi, \Delta}\,\neg\mathsf{R}}{\mathcal{T}, xRy, \Gamma, x : \Box\psi \vdash z : \neg\varphi, \Delta}\,\mathsf{4L} \qquad \dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, x : \Box\psi, y : \Box\psi, z : \varphi \vdash \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\psi, z : \varphi \vdash \Delta}\,\mathsf{4L}}{\mathcal{T}, xRy, \Gamma, x : \Box\psi \vdash z : \neg\varphi, \Delta}\,\neg\mathsf{R}$$

□R. By our assumption, we know that z is distinct from y in the inferences shown below left, meaning, we can permute ¬R below □R as shown below right. Trivially, the □R rule remains end-active after the permutation.

$$\dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, y : \Box\psi, z : \varphi \vdash y : \psi, \Delta}{\mathcal{T}, xRy, \Gamma, y : \Box\psi \vdash y : \psi, z : \neg\varphi, \Delta}\,\neg\mathsf{R}}{\mathcal{T}, \Gamma \vdash x : \Box\psi, z : \neg\varphi, \Delta}\,\Box\mathsf{R} \qquad \dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, y : \Box\psi, z : \varphi \vdash y : \psi, \Delta}{\mathcal{T}, \Gamma, z : \varphi \vdash x : \Box\psi, \Delta}\,\Box\mathsf{R}}{\mathcal{T}, \Gamma \vdash x : \Box\psi, z : \neg\varphi, \Delta}\,\neg\mathsf{R}$$

◀

▶ **Lemma 27.** *If* r *is a non-end-active propagation rule and* r′ *is non-initial and end-active, then* r *permutes below* r′ *and* r′ *remains end-active after this permutation.*

Proof. We consider the case where r is an instance of □L as the 4L case is similar. We show that r can be permuted down r′ and consider a representative number of cases when r′ is either ¬L, 4L, or □R as the remaining cases are similar.

¬L. By our assumption, we know that z is distinct from y in the inferences below left. We can therefore permute □L below ¬L as shown below right and we observe that ¬L remains end-active.

$$\dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \varphi \vdash z : \psi, \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash z : \psi, \Delta}\,\Box\mathsf{L}}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, z : \neg\psi \vdash \Delta}\,\neg\mathsf{L} \qquad \dfrac{\dfrac{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \varphi \vdash z : \psi, \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \varphi, z : \neg\psi \vdash \Delta}\,\neg\mathsf{L}}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, z : \neg\psi \vdash \Delta}\,\Box\mathsf{L}$$

4L. Let us suppose we have a □L instance followed by a 4L instance. There are two cases to consider: either the principal formula of □L is the same as for 4L, or the principal formulae are distinct. We show the first case as the second case is similar. Then, our inferences are of the form shown below left, where y and z are distinct due to our assumption. We may permute □L below 4L as shown below right and we observe that 4L remains end-active.

$$\dfrac{\dfrac{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi, y : \varphi, z : \Box\varphi \vdash \Delta}{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi, z : \Box\varphi \vdash \Delta}\,\Box\mathsf{L}}{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi \vdash \Delta}\,\mathsf{4L} \qquad \dfrac{\dfrac{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi, y : \varphi, z : \Box\varphi \vdash \Delta}{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi, y : \varphi \vdash \Delta}\,\mathsf{4L}}{\mathcal{T}, xRy, xRz, \Gamma, x : \Box\varphi \vdash \Delta}\,\Box\mathsf{L}$$

□R. Suppose we have an instance of □L followed by an application of □R as shown below.

$$\dfrac{\dfrac{\mathcal{T}, xRy, zRu, \Gamma, x : \Box\varphi, y : \varphi, u : \Box\psi \vdash u : \psi, \Delta}{\mathcal{T}, xRy, zRu, \Gamma, x : \Box\varphi, u : \Box\psi \vdash u : \psi, \Delta}\,\Box\mathsf{L}}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash z : \Box\psi, \Delta}\,\Box\mathsf{R}$$

By our assumption, the labels y and u are distinct, meaning, we can permute □L below □R as shown below. Trivially, □R remains end-active after the permutation is performed.

$$\dfrac{\dfrac{\mathcal{T}, xRy, zRu, \Gamma, x : \Box\varphi, y : \varphi, u : \Box\psi \vdash u : \psi, \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \varphi \vdash z : \Box\psi, \Delta}\,\Box\mathsf{R}}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash z : \Box\psi, \Delta}\,\Box\mathsf{L}$$

◀

▶ **Theorem 19.** *Each end-active proof in* CSGL *can be transformed into a proof in* LNGL.

Proof. We have included additional cases of the inductive step that are not included in the main text.

4L. Let us suppose that π ends with an application of 4L as shown below.

$$\dfrac{\mathcal{T}, xRy, \Gamma, x : \Box\varphi, y : \Box\varphi \vdash \Delta}{\mathcal{T}, xRy, \Gamma, x : \Box\varphi \vdash \Delta}\,4\mathsf{L}$$

By IH, we know there exists a path $y_1, \ldots, y_n$ of labels from the root y_1 to the leaf y_n in the premise of 4L such that $\mathcal{G} = \Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)$ is provable in LNGL. Since 4L is end-active, there are three cases to consider: either (1) neither x nor y occur along the path, (2) only x occurs along the path, or (3) both x and y occur along the path. In each case, the conclusion is obtained by taking the linear nested sequent corresponding to the path $y_1, \ldots, y_n$ in the conclusion of the 4L instance above. In the first and second cases, we translate the entire 4L instance as the single linear nested sequent $\mathcal{G}$. In the third case, we have that $x = y_{n-1}$ and $y = y_n$, meaning, the premise of the 4L instance shown below is provable in LNGL by IH, where $\Gamma(y_{n-1}) = \Sigma_1, \Box\varphi$ and $\Gamma(y_n) = \Sigma_2, \Box\varphi$. As shown below, a single application of 4L yields the desired conclusion.

$$\dfrac{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma_1, \Box\varphi \vdash \Delta(y_{n-1}) \mathbin{/\!\!/} \Sigma_2, \Box\varphi \vdash \Delta(y_n)}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma_1, \Box\varphi \vdash \Delta(y_{n-1}) \mathbin{/\!\!/} \Sigma_2 \vdash \Delta(y_n)}\,4\mathsf{L}$$

∨L. Let us suppose that π ends with an instance of ∨L as shown below.

$$\dfrac{\mathcal{T}, \Gamma, x : \varphi \vdash \Delta \qquad \mathcal{T}, \Gamma, x : \psi \vdash \Delta}{\mathcal{T}, \Gamma, x : \varphi \vee \psi \vdash \Delta}\,\vee\mathsf{L}$$

By IH, we know there exist paths $v = y_1, \ldots, y_n$ and $v = z_1, \ldots, z_k$ of labels from the root v to the leaves y_n and z_k in the premises of ∨L such that $\mathcal{G} = \Gamma(v) \vdash \Delta(v) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(y_n) \vdash \Delta(y_n)$ and $\mathcal{H} = \Gamma(v) \vdash \Delta(v) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Gamma(z_k) \vdash \Delta(z_k)$ are provable in LNGL. There are two cases to consider: either (1) $x \neq y_n$ or $x \neq z_k$, or (2) $x = y_n = z_k$. In the first case, if $x \neq y_n$, then we translate the entire ∨L inference as the single linear nested sequent $\mathcal{G}$, and if $x \neq z_k$, then we translate the entire ∨L inference as $\mathcal{H}$. In the second case, we know that the left premise $\mathcal{G}$ and right premise $\mathcal{H}$ of the ∨L inference below are provable with $\Gamma(y_n) = \Sigma, \varphi$ and $\Gamma(z_k) = \Sigma, \psi$, and so, a single application of ∨L gives the desired result.

$$\dfrac{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma, \varphi \vdash \Delta(y_n) \qquad \Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma, \psi \vdash \Delta(y_n)}{\Gamma(y_1) \vdash \Delta(y_1) \mathbin{/\!\!/} \cdots \mathbin{/\!\!/} \Sigma, \varphi \vee \psi \vdash \Delta(y_n)}\,\vee\mathsf{L}$$ ◀

▶ **Theorem 22.** *Every proof in* LNGL *can be transformed into a proof in normal form.*

Proof. Let π be a proof in LNGL. We consider an arbitrary instance of a □R rule in π and first show that every length-consistent 4L rule above □R can be permuted down into a block B of 4L rules above □R. Afterward, we will show that every length-consistent □L rule can be permuted down into a block B′ of □L rules above B. As a result, all length-consistent local rules will occurs in a block B″ above the premise of the block B′, showing that □R is preceded be a complete block. As these permutations can be performed for every □R instance in π, we obtain a normal form proof as the result.

Let us choose an application of □R in π, as shown below, proceeded by a (potentially empty) block R of 4L rules.

$$\dfrac{\dfrac{\vdots}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Delta \mathbin{/\!\!/} \Box\varphi \vdash \varphi}\,\mathrm{R}}{\mathcal{G} \mathbin{/\!\!/} \Gamma \vdash \Box\varphi, \Delta}\,\Box\mathsf{R}$$

We now select a bottom-most, length-consistent application of a $\mathsf{4L}$ rule above the chosen $\Box\mathsf{R}$ application that does not occur within the block R of $\mathsf{4L}$ rules. We show that $\mathsf{4L}$ can be permuted below every local rule and $\Box\mathsf{L}$ rule until it reaches and joins the R block. We show that $\mathsf{4L}$ can be permuted below $\neg\mathsf{R}$, $\vee\mathsf{L}$, and $\Box\mathsf{L}$ as the remaining cases are similar. Note that we are guaranteed that no other $\Box\mathsf{R}$ applications occur below $\mathsf{4L}$ and above R since then $\mathsf{4L}$ would not be length-consistent with the chosen $\Box\mathsf{R}$ application.

Suppose $\mathsf{4L}$ occurs above a $\neg\mathsf{R}$ application as shown below left. The rules can be permuted as shown below right.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi, \Box\varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vdash \Pi}\,\mathsf{4L}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \neg\psi, \Pi}\,\neg\mathsf{R} \qquad \dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi, \Box\varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi \vdash \neg\psi, \Pi}\,\neg\mathsf{R}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \neg\psi, \Pi}\,\mathsf{4L}$$

Suppose that we have $\mathsf{4L}$ followed by an application of the $\vee\mathsf{L}$ rule.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi, \psi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vdash \Pi}\,\mathsf{4L} \qquad \mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \chi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vee \chi \vdash \Pi}\,\vee\mathsf{L}$$

Invoking the hp-invertibility of $\mathsf{4L}$ (Lemma 14), we can permute $\mathsf{4L}$ below $\vee\mathsf{L}$ as shown below.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi, \psi \vdash \Pi \qquad \dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \chi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi, \chi \vdash \Pi}\,\mathsf{4L}^{-1}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\varphi, \psi \vee \chi \vdash \Pi}\,\vee\mathsf{L}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vee \chi \vdash \Pi}\,\mathsf{4L}$$

Last, we show (below left) one of the cases where $\mathsf{4L}$ is applied above a $\Box\mathsf{L}$ rule. We can permute the rules as shown below right.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\psi, \varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi \vdash \Pi}\,\mathsf{4L}}{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \Pi}\,\Box\mathsf{L} \qquad \dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\psi, \varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \Box\psi \vdash \Pi}\,\Box\mathsf{L}}{\mathcal{G} /\!\!/ \Gamma, \Box\psi, \Box\varphi \vdash \Delta /\!\!/ \Sigma \vdash \Pi}\,\mathsf{4L}$$

We can repeat the above downward permutations of bottom-most, length-consistent $\mathsf{4L}$ rules, so that all length-consistent $\mathsf{4L}$ rules occur in a block B above $\Box\mathsf{R}$ as shown below, where we let R' be a (potentially empty) block of $\Box\mathsf{L}$ rules above the B block of $\mathsf{4L}$ rules.

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\mathcal{H}}\,\mathrm{R}'}{\mathcal{G} /\!\!/ \Gamma \vdash \Delta /\!\!/ \Box\varphi \vdash \varphi}\,\mathrm{B}}{\mathcal{G} /\!\!/ \Gamma \vdash \Box\varphi, \Delta}\,\Box\mathsf{R}$$

Next we show that every length-consistent $\Box\mathsf{L}$ rule occurring above the block R' can be permuted down to the block R'. Let $\Box\mathsf{L}$ occur above the block R' be length-consistent with the chosen $\Box\mathsf{R}$ rule. Notice that we need only consider downward permutations of $\Box\mathsf{L}$ rules with local rules as all $\mathsf{4L}$ rules have already been permuted downward and no other $\Box\mathsf{R}$ rule can occur between $\Box\mathsf{L}$ and R' because then $\Box\mathsf{L}$ would not be length-consistent. We show how to permute the $\Box\mathsf{L}$ rule below a $\vee\mathsf{L}$ instance; the remaining cases are simple and similar.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi, \varphi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vdash \Pi}\,\Box\mathsf{L} \qquad \mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \chi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vee \chi \vdash \Pi}\,\vee\mathsf{L}$$

By using the hp-invertibility of $\Box\mathsf{L}$ (Lemma 14), we can permute $\Box\mathsf{L}$ below the $\vee\mathsf{L}$ rule.

$$\dfrac{\dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi, \psi \vdash \Pi \qquad \dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi, \chi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi, \psi \vee \chi \vdash \Pi}\,\vee\mathsf{L}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vee \chi \vdash \Pi}\,\Box\mathsf{L} \qquad \dfrac{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \chi \vdash \Pi}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \varphi, \chi \vdash \Pi}\,\Box\mathsf{L}^{-1}}{\mathcal{G} /\!\!/ \Gamma, \Box\varphi \vdash \Delta /\!\!/ \Sigma, \psi \vee \chi \vdash \Pi}\,\vee\mathsf{L}$$

By successively permuting all $\Box\mathsf{L}$ rules down into a block above B, we have that $\Box\mathsf{R}$ is preceded by a complete block in the proof. As argued above, this implies that every proof in LNGL can be put into normal form. ◀

Linear Realisability over Nets: Multiplicatives

Adrien Ragot
LIPN – UMR 7030 CNRS, Université Sorbonne Paris Nord, France
Dipartimento di Matematica e Fisica, Università Degli Studi Roma Tre, Italy

Thomas Seiller
CNRS, LIPN – UMR 7030, Université Sorbonne Paris Nord, France

Lorenzo Tortora de Falco
Dipartimento di Matematica e Fisica, Università Degli Studi Roma Tre, Italy
GNSAGA, Istituto Nazionale di Alta Matematica, Roma, Italy

Abstract

We provide a new realisability model based on orthogonality for the multiplicative fragment of linear logic, both in presence of generalised axioms (MLL$^{✠}$) and in the standard case (MLL). The novelty is the definition of cut elimination for generalised axioms. We prove that our model is adequate and complete both for MLL$^{✠}$ and MLL.

2012 ACM Subject Classification Theory of computation → Linear logic; Theory of computation → Program semantics

Keywords and phrases Linear Logic, Proof Nets, Realisability, Orthogonality, Hypergraphs, Rewriting, Correctness

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.43

Funding *Adrien Ragot*: Supported by a VINCI PhD fellowship from the Franco-Italian Université. *Thomas Seiller*: Partially supported by the ANR-22-CE48-0003-01 project DySCo.

Introduction

Since the inception of Linear Logic (LL), proofs are represented as graphs that naturally live in a wider space of agents called proof structures (*nets* in this paper) that can freely interact. These nets were introduced by J.Y. Girard in [7], together with the *desequentialisation*: a simple process transforming proof trees from the sequent calculus of LL into nets. However, not every net is the desequentialisation of a proof: it is *impossible* to extract a proof tree from a net that "contains" cycles or disconnections [5]. Nets can therefore present forms of (what we call) *geometrical incorrectness*, and geometrically correct nets are (representants of) proof trees of LL. More recently, J.Y. Girard proposed *Ludics*, an interpretation of LL given in terms of "desseins": proof trees of the LL sequent calculus with the addition of the daimon (✠) rule, a generalised axiom allowing to prove any sequent. Ludics introduces a new kind of incorrectness that we call *provability incorrectness*: dessein are geometrically correct (they are proof trees) but can be provably incorrect. In the standard theory of proof nets geometrical and provability correctness coincide; it is the presence of daimons that allows to distinguish between provability correctness and geometrical correctness.

Understanding the relationship between correctness and computational behavior is (one of) the goal(s) of *realisability*, which, restricted to LL, will be our focus in this paper. We briefly sum up the existing works on linear realisability[1] by positioning them with respect to Table 1. We also recall if these models enjoy completeness or not. Two lines of research on realisability of LL can be identified.

[1] We use the expression linear realisability in the sense of [20] i.e. realisability models for LL.

Table 1 Presence of incorrectness, restricted to multiplicative linear logic, for realisability models.

	MLL	MLL$^{\text{✠}}$
Proof Nets	no incorrectness	provability incorrectness
Nets	geometrical incorrectness	geometrical and provability incorrectness

One was initiated by V. De Paiva *Dialectica Interpretation* [6] and led to P. Oliva's adequate and complete realisability model of first order LL [14] where realisers are proof trees (with standard axioms) from a decorated sequent calculus of LL. As a consequence realisers are typed and are "by construction" *geometrically and provably correct* (placing this model in the top left corner of Table 1).

The other originates in the work of J.Y. Girard: *Ludics* [10], whose "desseins" are geometrically correct but can be provably incorrect (top right corner of Table 1), which enjoys *completeness*. E. Beffara proposed adequate models in a concurrent π-calculus [2] and conjunctive structure [3]. T. Seiller's *interaction graphs* (inspired by Girard's Geometry of Interaction [8]) model various LL fragments adequately [15–19]. Beffara's and Seiller's approaches exhibit both geometrical and provability incorrectness (bottom-right corner of Table 1), but contain no completeness result.

We give the first complete realisability model of the multiplicative fragment of linear logic in terms of nets, essentially the well–known untyped proof–structures of LL [9] with *daimons*, as in the work of P.L. Curien [4]: this places us in the bottom–right corner of Table 1. The main tool we use in our approach to realisability is LL cut elimination: we interpret formulas as types, sets of nets closed under bi–orthogonality, where the notion of orthogonality is defined via the rewriting rules of nets induced by cut elimination. We prove completeness for MLL$^{\text{✠}}$, multiplicative LL with generalised axioms, meaning our model can capture *geometrical correctness*. As a byproduct we obtain completeness for the standard multiplicative fragment of linear logic (MLL), thus capturing *provability correctness*.

Although not expressed in the terms of realisability, a completeness result for MLL$^{\text{✠}}$ (in the atomic case) using a notion of orthogonality is already apparent in the work of P.L. Curien [4], where the partitions involved in the Danos Regnier criterion [5] are encoded using daimons. More precisely, one can test the geometrical correctness of a net by confronting it against carefully chosen *opponents* (which as in the work of Béchet [1] are geometrically correct nets). However the method in [4] does not allow to derive a completeness result for MLL. By contrast, we use *geometrically incorrect* opponents to prove completeness for MLL (Remark 87).

The novelty is the *cut elimination* of non-homogenous cuts (a generalised axiom against a connective – say a tensor): unlike in Ludics [2] [10] our daimon is the "perfect" opponent/evaluation context; it never stops responding during computation and never prevents proof search to go on (Figure 4 and Remark 24). These new cut elimination steps are key to interactively identify provability correctness and so to obtain our completeness result for MLL (Remark 86). The computational behavior of the daimon also differs from Krivine's continuations involved in *classical realisability* [12]: they restore a previously stored context while our daimon rather behaves like an adaptive evaluation context.

The general aim is to understand the computational content of proofs and of (incorrect) nets, following a "purely interactive approach to logic" (to quote [10]). We follow the approach initiated with Ludics, we present a framework in which proofs and refutations are objects

[2] In Ludics, the daimon means the end of the game, or the end of the proof search.

of the same nature that can freely interact: a proof–object proves a formula A whenever it "defeats" all the refutations of A. The correctness of an object is evaluated using a dynamic criterion (we make an object interact with each of its refutations) rather than a static one (such as a typing discipline).

Outline. In Section 1, we give a detailed introduction of nets that we define as ordered hypergraphs. In Section 2, we recall the elementary notions of multiplicative linear logic, we introduce the ✠-links and we formulate the criterion of Danos Regnier [5] in our setting. In Section 3, we define orthogonality between two nets as "successful interaction" through cut elimination (Definition 42); this leads to the notion of type: a set of nets closed under bi–orthogonality. We then show how to perform the usual multiplicative constructions in the framework of types. In Section 4, we define our realisability model interpreting formulas as types and we prove its adequacy: a net representing a proof of A is a realiser of A (Theorem 64). In Section 5, we relate correctness criteria with orthogonality. The Danos-Regnier criterion applied to a cut-free net with conclusion A yields a set of nets called tests (Definition 74). We prove that the tests of A are proofs of $A^{\perp}$ (Theorem 77) and that the interaction between a net π with conclusion A and its tests allows to determine whether or not π is indeed a proof: we thus extend to our framework a result of Béchet [1]. In Section 6, we prove the completeness of our realisability model: if a net S realises A (in every basis), then S is a proof of A in $\mathsf{MLL}^{✠}$ (Theorem 85). Finally we show that completeness of $\mathsf{MLL}^{✠}$ implies that of MLL (Theorem 88).

1 Untyped nets

We introduce the framework of *nets* in which our construction takes place. Nets are a special kind of *directed hypergraphs* together with an order of *some* of their vertices which will come in play later on to define the notion of orthogonality. These hypergraphs enjoy a natural notion of sum (Definition 6). In subsection 1.2, we define our "realisers" that we call nets and their computational rules, the cut elimination procedure as known for multiplicative proof structures [9] but with a novelty: the generalised axiom or daimon–link (✠) which behave like an adaptative evaluation context.

1.1 Directed hypergraphs

Given a set X we will let $\mathcal{P}_{\leq}(X)$ denote the set of totally ordered finite subsets of X. An element of $\mathcal{P}_{\leq}(X)$ is equivalently a finite sequence of elements of X but, *without repetitions*.

▶ **Definition 1.** *Suppose given a set L of* labels*. A* directed (L-labelled) hypergraph *is a tuple $(V, E, \mathrm{s}, \mathrm{t}, \ell)$ where V is a finite set of* positions *and E is a finite set of* links*, $\mathrm{s} : E \to \mathcal{P}_{\leq}(V)$ is the* source map*, $\mathrm{t} : E \to \mathcal{P}_{\leq}(V)$ is the* target map *and $\ell : E \to L$ is the* labelling *map.*

Given a link $e \in E$, since the finite sets $\mathrm{t}(e)$ and $\mathrm{s}(e)$ are totally ordered, to support readability we will represent them as sequences: they are respectively called the *target* and the *source* sets of e. A *source* (resp. target) of e is an element of its source (resp. target) set $\mathrm{s}(e)$ (resp.$\mathrm{t}(e)$). The set of targets and sources of e is the *domain* of the link e. We will use superscripts to denote sequences of positions ($\overline{p}, \overline{q}, \overline{u}, \ldots$). A link is a *loop* when its target set and source set are not disjoint.

Convention. Along this work we assume all the hypergraphs to be loop–free i.e. containing only links which are not loops.

Given an hypergraph $\mathcal{H}$ with E as its set of links, we denote $\mathrm{s}(\mathcal{H})$ (resp. $\mathrm{t}(\mathcal{H})$) the set of all positions which are source (resp. target) of at least one link:

$$\mathrm{s}(\mathcal{H}) = \bigcup_{e \in E} \mathrm{s}(e), \qquad \mathrm{t}(\mathcal{H}) = \bigcup_{e \in E} \mathrm{t}(e).$$

A *conclusion/output* (resp. a *premise/input*) of a directed hypergraph $\mathcal{H}$ is a position which is the source (resp. target) of no link in $\mathcal{H}$, i.e. an element of $V \setminus \mathrm{s}(\mathcal{H})$ (resp. of $V \setminus \mathrm{t}(\mathcal{H})$). The set of conclusions (resp. premises) of an hypergraph $\mathcal{H}$ is denoted $\mathsf{out}(\mathcal{H})$ (resp. $\mathsf{in}(\mathcal{H})$). A position p is *isolated* in an hypergraph $\mathcal{H}$ if p is both an output and an input of $\mathcal{H}$, i.e. $p \notin \mathrm{s}(\mathcal{H}) \cup \mathrm{t}(\mathcal{H})$. The *size* of a directed hypergraph is the number of its links. There is a unique empty hypergraph $\mathcal{H} = (V, E, \mathrm{s}, \mathrm{t}, \ell)$ with $V = E = \mathrm{s} = \mathrm{t} = \emptyset$.

An *isomorphism* of hypergraphs $f : (V_1, E_1, \mathrm{s}_1, \mathrm{t}_1, \ell_1) \to (V_2, E_2, \mathrm{s}_2, \mathrm{t}_2, \ell_2)$ is a pair of functions (f_V, f_E) such that $f_V : V_1 \to V_2$ and $f_E : E_1 \to E_2$ are bijections, f_E preserve labels i.e. $\ell(f_E(e)) = \ell(e)$, and f_E preserves the target and source of a link, i.e. $\mathrm{s}_2(f_E(e)) = f_V^*(\mathrm{s}_1(e))$ and $\mathrm{t}_2(f_E(e)) = f_V^*(\mathrm{t}_1(e))$, where f_V^* is the natural extension of f_V to sequences of positions. Along this work we work with hypergraphs up to isomorphism.

▶ Notation 2. We denote $\langle \overline{u} \rhd_l \overline{v} \rangle$ the hypergraph $(V, E, \mathrm{s}, \mathrm{t}, \ell)$ such that $E = \{e\}$, $V = \mathrm{s}(e) \cup \mathrm{t}(e)$, $\mathrm{s}(e) = \overline{u}$, $\mathrm{t}(e) = \overline{v}$ and $\ell(e) = l$ (an example of such a single–link hypergraph is found in Figure 1a). In the sequel $\langle \overline{u} \rhd_l \overline{v} \rangle$ will denote both the described hypergraph and its unique link.

▶ Notation 3. We write $u \cdot v$ the concatenation of sequences. Given $u = (u_1, \ldots, u_n)$ a sequence of elements of a set X and an integer $i \in \{1, \ldots, n\}$, we denote by $u_{<i}$ (resp. $u_{>i}$) the sequence $(u_1, \ldots, u_{i-1})$ (resp. $(u_{i+1}, \ldots, u_n)$). Moreover, given two – potentially empty – sequences u and v we denote by $u[i \leftarrow v]$ the sequence $u_{<i} \cdot v \cdot u_{>i}$.

A link is *initial* (resp. *final*) when it has no input (resp. no output). A position is *initial* (resp. *final*) when it is an output (resp. input) of an initial (resp. final) link. In an hypergraph $\mathcal{H}$, a link e is *terminal* when every target of e is a conclusion of $\mathcal{H}$ – thus a final link is a terminal link.

▶ **Example 4.** For instance a link $\langle \rhd_\ell a, b, c \rangle$ is an initial link and the positions a, b and c are initial, on the other hand a link $\langle a, b \rhd_\ell c \rangle$ is not initial and neither are the positions a, b or c.

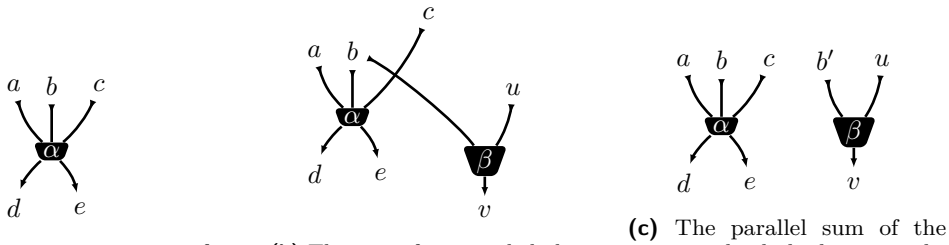
Hypergraphs enjoy a natural notion of sum based on the disjoint union of the set of links.

▶ Notation 5. Given two sets X_0 and X_1 we denote $X_0 \uplus X_1$ the set $X_0 \cup X_1$ whenever X_0 and X_1 are disjoint. Given two functions $f : X_0 \to E$ and $g : X_1 \to E$ with disjoint domains we denote $f \uplus g$ the function which takes an element x of $X_0 \uplus X_1$, and returns $f(x)$ if $x \in X_0$ and $g(x)$ if $x \in X_1$.

▶ **Definition 6.** *Given two hypergraphs $\mathcal{H}_1 = (V_1, E_1, \mathrm{t}_1, \mathrm{s}_1, \ell_1)$ and $\mathcal{H}_2 = (V_2, E_2, \mathrm{t}_2, \mathrm{s}_2, \ell_2)$ such that $E_1 \cap E_2 = \emptyset$. The sum of $\mathcal{H}_1$ and $\mathcal{H}_2$ is defined as:*

$$\mathcal{H}_1 + \mathcal{H}_2 = (V_1 \cup V_2, E_1 \uplus E_2, \mathrm{t}_1 \uplus \mathrm{t}_2, \mathrm{s}_1 \uplus \mathrm{s}_2, \ell_1 \uplus \ell_2).$$

▶ Remark 7. Whenever $\mathcal{H}_1 = (V_1, E_1, \mathrm{t}_1, \mathrm{s}_1, \ell_1)$ and $\mathcal{H}_2 = (V_2, E_2, \mathrm{t}_2, \mathrm{s}_2, \ell_2)$ are such that $E_1 \cap E_2 \neq \emptyset$, we will abusively write their sum as $\mathcal{H}_1 + \mathcal{H}_2 = (V_1 \cup V_2, E_1 \uplus E_2, \mathrm{t}_1 \uplus \mathrm{t}_2, \mathrm{s}_1 \uplus \mathrm{s}_2, \ell_1 \uplus \ell_2)$, since up to renaming the sets of links of two hypergraphs can always be considered disjoint.



(a) The representation of the single–link hypergraph $\langle a, b, c \rhd_\alpha d, e\rangle$ (Notation 2).

(b) The sum of two single link hypergraphs $\langle a, b, c \rhd_\alpha d, e\rangle$ and $\langle b, u \rhd_\beta v\rangle$ (Definition 6).

(c) The parallel sum of the two single link hypergraphs $\langle a, b, c \rhd_\alpha d, e\rangle$ and $\langle b, u \rhd_\beta v\rangle$ (Definition 11).

Figure 1 Hypergraphs can naturally be represented in a graphical way, we illustrate the notation of a hypergraph containing a single link, the sum of hypergraphs and the parallel sum of hypergraphs. In Figure 1c The position b is present in both hypergraphs therefore we rename it in one of the two hypergraphs: thus $\langle a, b, c \rhd_\alpha d, e\rangle \parallel \langle b, u \rhd_\beta v\rangle$ equals $\langle a, b, c \rhd_\alpha d, e\rangle \parallel \langle b', u \rhd_\beta v\rangle$ (that is, upto isomorphism).

▶ Remark 8. Vertices may overlap in a sum (as we take the union of vertex sets rather than the disjoint union). As a consequence, a position may be input (or output) of several distinct links (Figure 1b). We can describe hypergraphs as sums of simple hypergraphs; namely those that contain only one link. Indeed using Notation 2, an hypergraph consisting of two links $\langle \overline{a} \rhd_\ell \overline{b}\rangle$ and $\langle \overline{c} \rhd_{\ell'} \overline{d}\rangle$ is in fact equal to the sum of the single-link hypergraphs $\langle \overline{a} \rhd_\ell \overline{b}\rangle$ and $\langle \overline{c} \rhd_{\ell'} \overline{d}\rangle$. By induction on the number of links, this shows that any hypergraph $\mathcal{H}$ without isolated positions can be written as $\mathcal{H} = \sum_{e\in E} \langle \mathrm{s}(e) \rhd_{\ell(e)} \mathrm{t}(e)\rangle$.

▶ **Example 9.** In the hypergraph $\langle \rhd_{\ell 1} a, b, c\rangle + \langle a \rhd_{\ell 2} d\rangle + \langle \rhd_{\ell 3} e\rangle + \langle e \rhd_{\ell 4}\rangle$ the set of initial positions is $\{a, b, c, e\}$, while e is the only final position of the hypergraph, and it belongs to the domain of the unique final link $\langle e \rhd_{\ell 4}\rangle$.

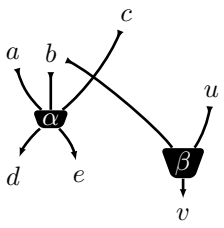
▶ Remark 10. The sum of hypergraphs enjoys the properties of an abelian monoid; associativity, commutativity, and a neutral element which is the empty hypergraph.

We will also use extensively the notion of *parallel composition* or *parallel sum* of hypergraphs, an analogue of the *union–graph* of two simple graphs.

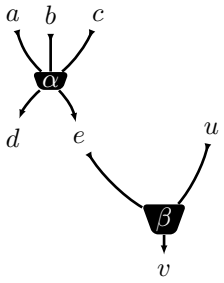
▶ **Definition 11.** *Given* $\mathcal{H}_1 = (V_1, E_1, \mathrm{t}_1, \mathrm{s}_1, \ell_1)$ *and* $\mathcal{H}_2 = (V_2, E_2, \mathrm{t}_2, \mathrm{s}_2, \ell_2)$ *two hypergraphs such that* $V_1 \cap V_2 = E_1 \cap E_2 = \emptyset$*, we define their* parallel sum *as:* $\mathcal{H}_1 \parallel \mathcal{H}_2 = (V_1 \uplus V_2, E_1 \uplus E_2, \mathrm{t}_1 \uplus \mathrm{t}_2, \mathrm{s}_1 \uplus \mathrm{s}_2, \ell_1 \uplus \ell_2)$.

▶ Remark 12. The parallel sum of two hypergraphs $\mathcal{H}_1$ and $\mathcal{H}_2$ corresponds to a regular sum whenever the sets of vertices are disjoint. Just like the sum, parallel composition can always be performed between two hypergraphs (up to a renaming, see Figure 1c).

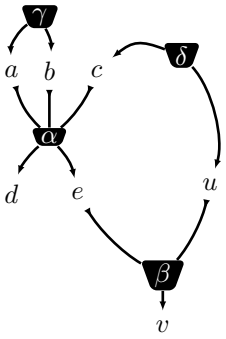
A hypergraph $\mathcal{H} = (V, E, \mathrm{t}, \mathrm{s}, \ell)$ is: (1) *target–surjective* whenever $\mathrm{t}(\mathcal{H}) = V$, (2) *source–disjoint* if the sets $\mathrm{s}(e)$ for $e \in E$ are pairwise disjoint, (3) *target–disjoint* if the sets $\mathrm{t}(e)$ for $e \in E$ are pairwise disjoint (Figure 2). A *module* is an hypergraph which is target–disjoint and source–disjoint, which means that for each position p there exists *at most* one link e such that $\mathrm{s}(e)$ (resp. $\mathrm{t}(e)$) contains p. Any single–link hypergraph is a module. Uncarefully summing two modules does not necessarily result in a module; for instance the single link hypergraphs $e = \langle \rhd_\ell a\rangle$ and $e' = \langle \rhd_{\ell'} a\rangle$ are both modules but their sum isn't as a is the target of the two links e and e'.



(a) The sum of two single link hypergraphs $\langle a, b, c \rhd_\alpha d, e\rangle + \langle b, u \rhd_\beta v\rangle$. The hypergraph is target–disjoint, but because b belongs to the source of both links it is not source–disjoint, it is also not target surjective.



(b) The sum of two single link hypergraphs $\langle a, b, c \rhd_\alpha d, e\rangle + \langle e, u \rhd_\beta v\rangle$. The hypergraph is target–disjoint and source–disjoint, however it is not target surjective.



(c) The sum of four single link hypergraphs $\langle a, b, c \rhd_\alpha d, e\rangle + \langle e, u \rhd_\beta v\rangle + \langle \rhd_\gamma a, b\rangle + \langle \rhd_\delta c, u\rangle$. The hypergraph is target–disjoint, source–disjoint, and target surjective.

Figure 2 Properties of hypergraphs: source–disjoint, target–disjoint and target–surjective hypergraphs.

An *arrangement* of a directed hypergraph $\mathcal{H}$ is a total order $<_{\mathbf{a}}$ on its set of conclusions; equivalently the order may be identified as a bijection $\mathbf{a} : \{1, \dots, card(\mathsf{out}(\mathcal{H}))\} \to \mathsf{out}(\mathcal{H})$. An *ordered hypergraph* is a pair $(\mathcal{H}, \mathbf{a})$ of an hypergraph $\mathcal{H}$ together with an arrangement $\mathbf{a}$ of $\mathcal{H}$. Given an ordered hypergraph $(\mathcal{H}, \mathbf{a})$ with n conclusions for an integer $1 \leq i \leq n$, we denote $\mathbf{a}(i)$ by $\mathcal{H}(i)$ whenever there is no ambiguity. The arrangement $\mathbf{a}$ is denoted $\mathbf{a}(\mathcal{H})$, and we might refer to $\mathcal{H}$ as the *unordered hypergraph underlying* $(\mathcal{H}, \mathbf{a})$.

For $n, m \in \mathbb{N}$ we denote by $[n; m]$ the set of integers i such that $n \leq i \leq m$. Given two functions $f : [1; n] \to E$ and $g : [1; m] \to E$ we denote $f \uplus g : [1; m + n] \to E$ the function such that $f \uplus g(i) = f(i)$ when $1 \leq i \leq n$ and $f \uplus g(i) = g(i - n)$ when $n + 1 \leq i \leq n + m$. This operation is not commutative. The parallel sum of two ordered hypergraph $(\mathcal{H}_1, \mathbf{a}_1)$ and $(\mathcal{H}_2, \mathbf{a}_2)$ naturally yields an ordered hypergraph as $(\mathcal{H}_1 \parallel \mathcal{H}_2, \mathbf{a}_1 \uplus \mathbf{a}_2)$ (note that however this is not a commutative operation).

1.2 Multiplicative nets

Up to this point we have allowed any kind of link to occur in a hypergraph. We now consider untyped multiplicative nets in which only some specific kinds of links occur. We fix the set of labels as the set made of the *daimon* (✠) the tensor ($\otimes$) the par ($\parr$) and the cut (cut) symbol. Furthermore we fix a family of links, namely ✠-labelled links that have no inputs (they are initial links), cut-labelled links that have exactly two inputs and no outputs (they are final links), $\otimes$- and $\parr$-labelled links that have exactly two inputs and one output. As a consequence, the hypergraphs considered will closely resemble to multiplicative linear logic proof structures, with two important points of divergence: the absence of typing and the presence of generalised axioms, a standard MLL axiom link can be seen as daimon link with two conclusions [3].

Formally we fix a countable set Pos of positions and a family of links $\mathcal{L}$ defined as:

$$\mathcal{L} \triangleq \{\langle p_1, p_2 \rhd_\otimes p\rangle,\ \langle p_1, p_2 \rhd_\parr p\rangle,\ \langle p_1, p_2 \rhd_{\mathsf{cut}}\rangle \mid p_1, p_2, p \in \mathsf{Pos}\} \quad \cup \quad \{\langle \rhd_✠ p_1, \dots, p_n\rangle \mid n \in \mathbb{N}, p_1, \dots, p_n \in \mathsf{Pos}\}.$$

[3] To be precise one should say that an *atomic* standard MLL axiom link *is* a daimon link with two conclusions (Remark 29).

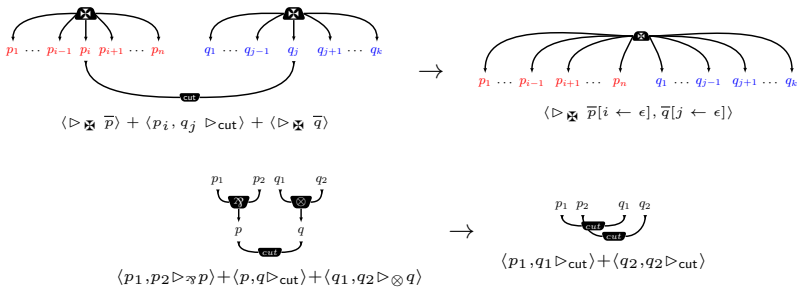


Figure 3 Rewriting defining the homogeneous cut elimination. We provide a representation of each hypergraph involved above its expression. In the step of the glueing cut we assume the two daimons to be distinct i.e. the cut is acyclic. In this figure $\overline{p} = p_1, \ldots, p_n$ while $\overline{q} = q_1, \ldots, q_k$.

▶ **Definition 13.** *A multiplicative module is an ordered hypergraph $M = (|M|, \mathbf{a}(M))$ where $|M|$ is a sum of links of $\mathcal{L}$ which is a module.*

A multiplicative net is a multiplicative module $S = (|S|, \mathbf{a}(S))$ where $|S|$ is target–surjective.

From now on we will omit the word *multiplicative* but a module (resp. net) will always be a multiplicative module (resp. net). For a module M (resp. a net S) we refer to $|M|$ (resp. $|S|$) as the unordered hypergraph underlying M (resp. S). An *unordered* module (resp. net) is the unordered hypergraph underlying a module (resp. net).

▶ Remark 14. For two nets $S_1 = (V_1, E_1, \mathrm{s}_1, \mathrm{t}_1, \ell_1)$ and $S_2 = (V_2, E_2, \mathrm{s}_2, \mathrm{t}_2, \ell_2)$, if $S_1 + S_2$ remains a net then $S_1 + S_2 = S_1 \parallel S_2$. Indeed, by Definition 6, $E_1 \cap E_2 = \emptyset$. Then, by target–disjointness $\mathrm{t}(S_1) \cap \mathrm{t}(S_2) = \emptyset$; and finally because S_1 and S_2 are target surjective we have $V_1 \cap V_2 = \mathrm{t}(S_1) \cap \mathrm{t}(S_2) = \emptyset$, so that Definition 11 applies.

▶ Notation 15. Given an integer n we denote by $✠_n$ any multiplicative net consisting of a single daimon link with n outputs, i.e. isomorphic to $\langle \triangleright_{✠} p_1, \ldots, p_n \rangle$.

▶ **Definition 16.** *Given a multiplicative net S the* type *of a cut link $c = \langle p, q \triangleright_{\mathsf{cut}} \rangle$ occurring in S is the multiset of the two labels of the links of output p and q; for readability we write these multisets as ordered pairs. Thus there are six* types *of cuts (up to symmetry). More precisely, we distinguish:* multiplicative *cuts, of type* $(\otimes/⅋)$*;* clash *cuts, of type* $(\otimes/\otimes)$ *or* $(⅋/⅋)$*;* glueing *cuts, of type* $(✠/✠)$*;* non–homogeneous *cuts, of type* $(\otimes/✠)$ *or* $(⅋/✠)$*, which are respectively called* reversible *and* irreversible *cuts. In a net S, a cut $\langle p, q \triangleright_{\mathsf{cut}} \rangle$ is* cyclic *whenever p and q are targets of the same link.*

▶ Remark 17. Each cut link occurring in a net S has a type since a net is target–surjective. However in a module this isn't true: for instance in the module $\langle p, q \triangleright_{\mathsf{cut}} \rangle$ consisting of a single cut link, the type of the cut link is not defined.

▶ Remark 18. The inputs of a cut link $\langle p, q \triangleright_{\mathsf{cut}} \rangle$ are ordered, making the two links $\langle p, q \triangleright_{\mathsf{cut}} \rangle$ and $\langle q, p \triangleright_{\mathsf{cut}} \rangle$ distinct. However (up to isomorphism) this plays no role during cut elimination.

Multicative nets comes with their notion of computation called *cut elimination*: it is a rewriting on nets and more precisely it rewrites a redex (that is a sub–net made of a single cut link and two non–cut links) into redexes or daimons (in the very specific case of glueing cuts). Up to isomorphism, how a redex is rewritten depends solely on its type (Definition 16).

▶ **Definition 19.** *The relation of* homogeneous cut elimination *on unordered nets is denoted by $\to_h$ and it is the rewriting relation defined as the contextual closure (with respect to the sum) of the relation defined in Figure 3.*

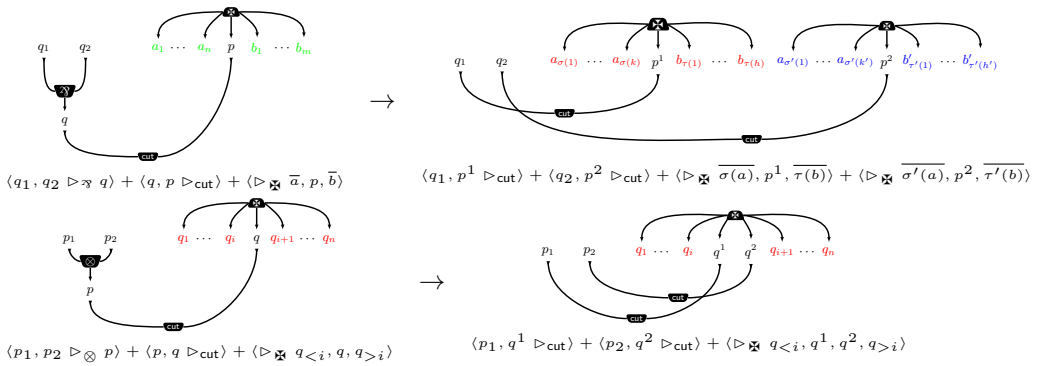


Figure 4 Rules defining the non–homogeneous cut elimination. In the elimination of the ($\parr$/$\pentagram$) cut - first row - $\overline{a} = (a_1, \ldots, a_n)$ and $\overline{b} = (b_1, \ldots, b_m)$ while $\sigma(\overline{a}) = (a_{\sigma(1)}, \ldots, a_{\sigma(k)})$, $\sigma'(\overline{a}) = (a_{\sigma'(1)}, \ldots, a_{\sigma'(k')})$, $\tau(\overline{b}) = (b_{\tau(1)}, \ldots, b_{\tau(h)})$, $\tau'(\overline{b}) = (b_{\tau'(1)}, \ldots, b_{\tau'(h')})$ (with $n = k + k'$ and $m = h + h'$) are sequences that define a partition of $\{a_1, \ldots, a_n, b_1, \ldots, b_n\}$ more precisely $\{a_1, \ldots, a_n\} = \{a_{\sigma(1)}, \ldots, a_{\sigma(k)}, a_{\sigma'(1)}, \ldots, a_{\sigma'(k')}\}$ and $\{b_1, \ldots, b_n\} = \{b_{\tau(1)}, \ldots, b_{\tau(h)}, b_{\tau'(1)}, \ldots, b_{\tau'(h')}\}$, and $\sigma, \sigma', \tau, \tau'$ are permutations. Furthermore p^1, p^2, q^1, q^2 are fresh positions. The figure is slightly misleading: q_1 and q_2 may be elements of $\overline{a}$ or $\overline{b}$ (in the first row) while p_1 and p_2 may be elements of $q_1, \ldots, q_{i-1}, q_{i+1}, \ldots, q_n$ (in the second row), these cases are illustrated in Figure 13. This has an important consequence: a cut can belong to a cycle and still be reducible (Remark 28).

▶ Remark 20. The (homogeneous) cut elimination procedure on unordered nets leave the conclusions unchanged. As a consequence the homogeneous cut elimination can be lifted from unordered nets to nets: whenever two unordered nets are such that $S \to S'$, for any arrangement $\mathbf{a}$ of S we have $(S, \mathbf{a}) \to (S', \mathbf{a})$.

The following result is easily established, in particular since the number of links strictly decreases during homogeneous cut elimination.

▶ **Proposition 21.** *Homogeneous cut elimination is confluent and strongly normalizing.*

▶ **Definition 22.** *The non homogeneous reduction is denoted $\to_{nh}$ and it is defined on unordered nets as the contextual closure of the relation given in Figure 4.*

▶ Remark 23. The non–homogeneous reduction preserves the conclusion of the nets, hence it can be lifted to ordered nets – as in remark 20.

▶ Remark 24. In the framework of Multiplicative Linear Logic (Section 2, Figure 6c), non homogeneous cut elimination simulates proof search in the sequent calculus:

$$\dfrac{\dfrac{\vdash \Gamma, A \parr B}{\vdash \Gamma, A \parr B}\pentagram \quad \dfrac{\dfrac{\vdash A^\perp, A}{\vdash A^\perp \otimes B^\perp, A, B}\otimes \quad \dfrac{\vdash B^\perp, B}{\vdash A^\perp \otimes B^\perp, A \parr B}\parr}{\vdash \Gamma, A \parr B}\mathsf{cut} \quad \to^* \quad \dfrac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\parr \qquad \dfrac{\dfrac{\vdash \Gamma, A \otimes B}{\vdash \Gamma, A \otimes B}\pentagram \quad \dfrac{\dfrac{\vdash A^\perp, A}{\vdash A^\perp, B^\perp, A \otimes B}\otimes \quad \dfrac{\vdash B^\perp, B}{\vdash A^\perp \parr B^\perp, A \otimes B}\parr}{\vdash \Gamma, A \otimes B}\mathsf{cut} \quad \to^* \quad \dfrac{\vdash \Gamma_1, A}{\vdash \Gamma, A \otimes B}\pentagram \quad \dfrac{\vdash \Gamma_2, B}{}\pentagram}{\vdash \Gamma, A \otimes B}\otimes$$

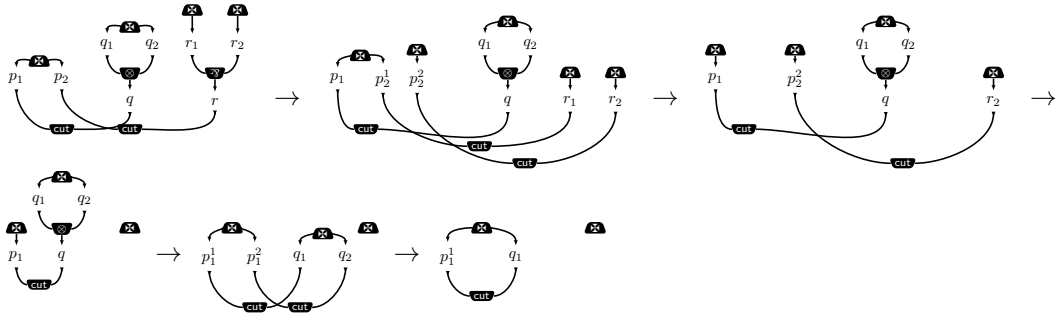
This also illustrates the non determinism of the ($\pentagram$/$\parr$) reduction step which corresponds to proof search on a formula of the form $A \otimes B$: going from bottom to top the $\otimes$–introduction rule splits the context Γ, which is a non deterministic process. A consequence of non determinism is the loss of confluence for cut elimination (but not of strong normalisation, Proposition 30); since splitting the context is irreversible, a net can have different normal forms, like the second net of figure 5b (from left to right) which coincides with the second net of figure 5c: this same net reduces, following the two figures, to two different normal forms.

▶ Remark 25. A cyclic cut is a glueing cut. Indeed, given a cyclic cut link $\langle p, q \rhd_{\mathsf{cut}} \rangle$ in a net, because p and q belong to the target of a same link e and the only links which may have several targets are daimon links it follows that e is a daimon link.

▶ Remark 26. The side condition of Figure 3 entails that a cyclic cut is not reducible: for example the net $\langle \rhd_{✠} p, q \rangle + \langle p, q \rhd_{\mathsf{cut}} \rangle$ is a net in normal form.

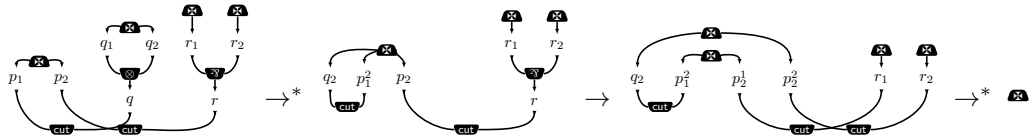
▶ Remark 27. A cut link which is not reducible is either a clashing cut or a cyclic glueing cut. Notice, however, that while clashing cuts never disappear during cut elimination, cyclic cuts may disappear (see Figure 10b).

▶ Remark 28. In the standard framework of MLL proof structures the cut elimination of an axiom against a cut is defined as the identification of the two extreme positions, therefore eliminating such a cut may create *loops* (Section 1). To avoid loops from occurring during cut elimination an ad hoc condition is usually added (see for example [13]). In our framework, this condition is the rather natural side condition of Figure 3.

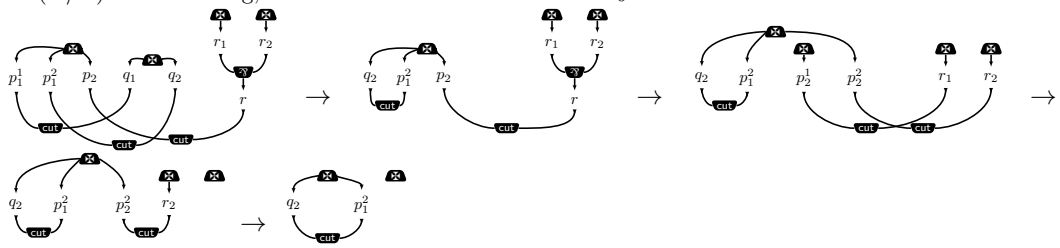


(a) Eliminating first the *irreversible cut* (✠/⅋) produces a net[a] which cannot normalize in $✠_0$.

[a] The (✠/⅋) reduction step is not deterministic but in this very special case any choice yields the same net.



(b) Eliminating the reversible cut (✠/⊗) produces a cycle which can be eliminated by the elimination of the (✠/⅋) cut remaining, hence that net can normalize in $✠_0$.



(c) Non determinism also comes from the choice of how we reduce (⅋/✠) cuts, different choices leading to different normal forms: the "wrong" choice results in a net which cannot normalize to $✠_0$.

Figure 5 Non homogeneous cut eliminations contains two sources of non–determinism.

▶ Remark 29. Notice that whenever daimons are binary and typed by dual atomic formulas the cut elimination procedure for MLL$^{✠}$ defined in Definition 19 is exactly the standard cut elimination procedure for MLL [7], [13].

The rewriting rule, denoted $\to$, associated with cut elimination is the union of the homogeneous and non–homogeneous cut elimination i.e. $\to_h \cup \to_{nh}$. We write $S \xrightarrow{c} S'$, when S' is obtained from S by eliminating the cut c. We write by $S \to_{\mathsf{mult}} S'$ (resp. $S \to_{\neg\mathsf{mult}} S'$) whenever $S \xrightarrow{c} S'$ and c is multiplicative (resp. not multiplicative). Given two binary relations R_1 and R_2 on a set X we denote by $R_1 \cdot R_2$ their composition, i.e. for two $x, y \in X$ $xR_1 \cdot R_2 y$ if and only if there exists z such that $xR_1 z$ and $zR_2 y$.

▶ **Proposition 30.** *Cut elimination is strongly normalising, furthermore:*
1. $\to^*$ *can be factorised as* $\to^*_{\mathsf{mult}} \cdot \to^*_{\neg\mathsf{mult}}$.
2. *If c is a $(\parr/\maltese)$ cut in S; if $S \xrightarrow{c} \cdot \to^* S'$ then $S \to^* \cdot \xrightarrow{c} S'$.*
3. *If c is not a $(\parr/\maltese)$ cut in S; if $S \to^* \cdot \xrightarrow{c} S'$ then $S \xrightarrow{c} \cdot \to^* S'$.*

$$\begin{aligned} A, B &\triangleq X \in \mathsf{Var} \\ &\mid A \parr B \mid A \otimes B \\ \mathcal{H}_1, \mathcal{H}_2 &\triangleq A \in \mathsf{Form} \\ &\mid \mathcal{H}_1, \mathcal{H}_2 \mid \mathcal{H}_1 \parallel \mathcal{H}_2 \end{aligned}$$

(a) Grammar defining Form (first two rows), and grammar defining Hseq (last two rows).

$$(A \parr B)^\perp = A^\perp \otimes B^\perp \qquad (A \otimes B)^\perp = A^\perp \parr B^\perp$$

(b) De Morgan laws lifting the involution $(\cdot)^\perp$ from Var to Form.

$$\frac{}{\Gamma}\,{\scriptstyle \maltese} \qquad \frac{\Gamma, A, B}{\Gamma, A \parr B}\,{\scriptstyle \parr} \qquad \frac{\Gamma, A \quad \Delta, B}{\Gamma, \Delta, A \otimes B}\,{\scriptstyle \otimes} \qquad \frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta}\,{\scriptstyle \mathsf{cut}} \qquad \frac{\Gamma, A, B, \Delta}{\Gamma, B, A, \Delta}\,{\scriptstyle \mathsf{ex}} \qquad \frac{}{A, A^\perp}\,{\scriptstyle \mathsf{ax}}$$

(c) Rules used for constructing the proof trees. The rules $(\maltese, \parr, \otimes, \mathsf{cut}, \mathsf{ex})$ define the $\mathsf{MLL}^{\maltese}$ fragment. Substituting the $(\maltese)$–daimon rule with the (ax)–axiom rule results in the fragment MLL, that is $(\mathsf{ax}, \parr, \otimes, \mathsf{cut}, \mathsf{ex})$.

Figure 6 Grammar of formulas and (hyper)sequent, de Morgan laws and inference rules.

$$\langle \overline{c} \rhd_\ell \overline{a}, p_1, \overline{b} \rangle + \langle p_1, p_2 \rhd_\parr p \rangle \to^l_\parr \langle \overline{c} \rhd_\ell \overline{a}, p, \overline{b} \rangle \qquad \langle \overline{c} \rhd_\ell \overline{a}, p_2, \overline{b} \rangle + \langle p_1, p_2 \rhd_\parr p \rangle \to^r_\parr \langle \overline{c} \rhd_\ell \overline{a}, p, \overline{b} \rangle$$

Figure 7 The two cases (left and right) defining the switching rewriting. The left reduction $\to^l_\parr$ destroys p_1 and makes p_2 a conclusion; while the right reduction $\to^r_\parr$ destroys p_2 and makes p_1 a conclusion.

$$\frac{}{\Gamma}\,{\scriptstyle \maltese} \qquad \frac{\prod \pi_1 \quad A, \Gamma \qquad \prod \pi_2 \quad A^\perp, \Delta}{\Gamma, \Delta}\,{\scriptstyle \mathsf{cut}} \qquad \frac{\prod \pi_1 \quad A, \Gamma \qquad \prod \pi_2 \quad B, \Delta}{\Gamma, \Delta, A \otimes B}\,{\scriptstyle \otimes} \qquad \frac{\prod \pi_0 \quad A, B, \Gamma}{A \parr B, \Gamma}\,{\scriptstyle \parr} \qquad \frac{\prod \pi_0 \quad \Gamma, B, A, \Delta}{\Gamma, A, B, \Delta}\,{\scriptstyle \mathsf{ex}}$$

$$\langle \rhd_\maltese p_1, \ldots, p_n \rangle \qquad S_1 + S_2 + \langle S_1(1), S_2(1) \rhd_{\mathsf{cut}} \rangle \qquad S_1 + S_2 + \langle S_1(1), S_2(1) \rhd_\otimes p \rangle \qquad S_0 + \langle S_0(1), S_0(2) \rhd_\parr p \rangle \qquad (S_0, a)$$

Figure 8 Induction defining the relation $\equiv_{\mathcal{R}}$. The proof in the first row is represented by a net below it in the second row. The position p is always supposed fresh. In each case and for each $0 \leq i \leq 2$, S_i is a net which represent π_i i.e. $S_i \equiv_{\mathcal{R}} \pi_i$. In the case of the exchange rule we explicitly mention the arrangement i.e. the order of the conclusion and assume $(S_0, a') \equiv_{\mathcal{R}} \pi_0$ and $a(i) = a'(i)$ whenever $i \leq |\Gamma|$ or $|\Gamma| + 2 < i$. On the other hand, $a'(|\Gamma| + 1) = a(|\Gamma| + 2)$ and $a'(|\Gamma| + 2) = a(|\Gamma| + 1)$.

2 Multiplicative Linear Logic and proof nets

We define the well–known notion of proof net [7] in our setting: in the presence of the generalised axiom $(\maltese)$, proof nets are similar to the *paraproof nets* of Curien [4] (which come from Girard Ludics [10]). We then formulate the Danos–Regnier criterion [5]: testing the acyclicity and connectedness of (several) graphs allows to determine whether a net is a (para)proof net or not [4].

We fix a countable set Var of *propositional variables*. The set Var comes with an (explicit) involution $(\cdot)^{\perp}$; for each atomic variable X there exists its *dual* atomic variable $X^{\perp}$ in Var. The set Form of *formulas* of multiplicative linear logic is defined by the grammar in Figure 6a. The involution $(\cdot)^{\perp}$ is lifted from Var to Form as in Figure 6b. The set Hseq of *hypersequents* is defined by the grammar in Figure 6a, a *sequent* is an hypersequent without the *parallel* "$\|$" constructor. The introduction of hypersequents is naturally suggested by the constructions on types (Section 3): indeed as the interpretation of the $\parr$–connective is based on the interpretation of the ","–connective, the interpretation of the $\otimes$–connective relies on that of the "$\|$"–connective (Definition 49 and Definition 58). Technically hypersequents are necessary in *our* proof of the completeness theorem (Theorem 88).

A *proof* of MLL (resp. $\mathsf{MLL}^{\maltese}$) is a tree constructed using the rules $(\mathsf{ax}, \parr, \otimes, \mathsf{cut}, \mathsf{ex})$ (resp. $(\maltese, \parr, \otimes, \mathsf{cut}, \mathsf{ex})$) of Figure 6c.

▶ **Definition 31.** *A net S* represents[4] *a proof π of $\mathsf{MLL}^{\maltese}$, denoted $\pi \equiv_{\mathcal{R}} S$ or $S \equiv_{\mathcal{R}} \pi$, whenever the relation defined in Figure 8 holds. A net represents a proof of MLL whenever it represents a proof of $\mathsf{MLL}^{\maltese}$ where every sequent conclusion of a $(\maltese)$–rule has shape $A, A^{\perp}$ for $A \in \mathsf{Form}$. A representation of a proof π is a net S which represents π. A* proof net *of $\mathsf{MLL}^{\maltese}$ (resp. MLL) is a net which represents a proof of $\mathsf{MLL}^{\maltese}$ (resp. MLL): we say that S is* correct. *A net S is* correctly typeable[5] *by a sequent Γ whenever it represents a proof of Γ in $\mathsf{MLL}^{\maltese}$.*

▶ Notation 32. Let $\mathfrak{P}$ denote MLL or $\mathsf{MLL}^{\maltese}$ and let S be a net. We write $S \vdash_{\mathfrak{P}} \Gamma$ whenever there exists a proof π in $\mathfrak{P}$ such that S is the representation of π. Furthermore we denote $\{\!|\Gamma : \mathfrak{P}|\!\}$ the set of all the nets S such that $S \vdash_{\mathfrak{P}} \Gamma$.

A *substitution* is a map $\theta : \mathsf{Var} \to \mathsf{Form}$ such that $\theta(X^{\perp}) = \theta(X)^{\perp}$ for each $X \in \mathsf{Var}$. A substitution can be lifted to formulas and hypersequents by induction: $\theta(A \otimes B) = \theta(A) \otimes \theta(B)$; $\theta(A \parr B) = \theta(A) \parr \theta(B)$; $\theta(A \parallel B) = \theta(A) \parallel \theta(B)$; $\theta(A, B) = \theta(A), \theta(B)$. Given two hypersequents, we denote $\Delta \leq \Gamma$ whenever there exists a substitution θ such that $\theta\Delta = \Gamma$.

▶ **Proposition 33.** *Let Γ and Δ be two sequents and suppose $\Delta \leq \Gamma$. For any net S: (1) if $S \vdash_{\mathsf{MLL}^{\maltese}} \Delta$ then $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$ and (2) if $S \vdash_{\mathsf{MLL}} \Delta$ then $S \vdash_{\mathsf{MLL}} \Gamma$.*

▶ **Definition 34.** *The* switching *rewriting is defined on unordered nets as the contextual closure of the rules in Figure 7. A* switching *of a net S is a normal form of S for the switching rewriting: we often denote it σS.*

▶ Remark 35. The switching rewriting strongly normalizes since every step reduces the number of links of the net. The rewriting is also non-deterministic and non-confluent, every normal form is a par–free net. The switching rewriting can be lifted to (ordered) nets; with the notations of Figure 7 whenever an unordered net $|S|$ with n conclusions is such that $|S| \to^{l}_{\parr} |S'|$ we define $(|S|, \mathbf{a}) \to^{l}_{\parr} (|S'|, \mathbf{a}')$ where $\mathbf{a}'(i) = \mathbf{a}(i)$ for each $1 \leq i \leq n$ and $\mathbf{a}'(n+1) = p_2$ i.e. the new conclusion is made last conclusion (similarly we can define it for the case $\to^{r}_{\parr}$).

[4] In the standard Linear Logic terminology π is a sequentialisation of the proof net S.

[5] Notice that with the expressions "correctly typeable" we mean here that the net is both correct (it represents a proof) and that we can label its conclusions with the formulas of Γ.

▶ **Definition 36.** *The* undirected multigraph[6] *induced by two partitions P and Q of a set X is* (V, E, brd) *denoted* $\mathsf{G}(P, Q)$ *where: (1)* $V = \{1\} \times P \cup \{2\} \times Q$ *the vertices are the classes of P and Q (as a disjoint union); (2)* $E = X$*; (3) For any edge x in X;* $\mathsf{brd}(x) = \{(1, P_x), (2, Q_x)\}$ *where $P_x \in P$ is such that $x \in P_x$ and $Q_x \in Q$ is such $x \in Q_x$.*

Two partitions P and Q of a set X are orthogonal *if the multigraph* $\mathsf{G}(P, Q)$ *is acyclic and connected.*

▶ **Definition 37.** *In a net S denote $p \geq_S q$ the relation which holds whenever there exists a link e such that $p \in \mathrm{s}(e)$ and $q \in \mathrm{t}(e)$. Denote $\geq_S^*$ its reflexive and transitive closure; a position p is* above *a position q whenever $p \geq_S^* q$. Given a position q we denote $q \uparrow^i S$ the set of* initial *positions which are above q in S.*

▶ Remark 38. Given a *cut–free* net S with conclusions $p_1, \ldots, p_n$ the sets $p_1 \uparrow^i S, \ldots, p_n \uparrow^i S$ form a partition of the initial positions of S. We denote this partition $\uparrow^i S$.

▶ Notation 39. Let S be a net and let $\{d_1, \ldots, d_n\}$ be the set of daimon links of S. The partition $\{\mathrm{t}(d_1), \ldots, \mathrm{t}(d_n)\}$ on the set of initial positions of S is denoted by $\mathsf{P}_{\text{✠}}(S)$.

Reformulated in the context of hypergraphs we get the following theorem from [5].

▶ **Theorem 40** ([4, 5])**.** *Given a cut–free net S, the following assertions are equivalent:*

1. *S is a proof net of* $\mathsf{MLL}^{\text{✠}}$*;*
2. *For every switching σS of S, the partitions $\mathsf{P}_{\text{✠}}(S)$ and $\uparrow^i \sigma S$ of the set of initial positions of S are orthogonal;*
3. *Every switching σS of S is acyclic and connected*[7]*.*

3 Interaction of nets, orthogonality, and types

We define how nets can *interact* and if the interaction of two nets leads to the ✠-link with no outputs ($✠_0$) we say they are *orthogonal*. This recalls classical realisability proposed by J.-L. Krivine [12], where (the closure by antireduction of) the set $\{✠_0\}$ will play the role of the *pole*. Notice, however, that our setting is fully symmetrical: both the elements of truth values and falsity values are nets.

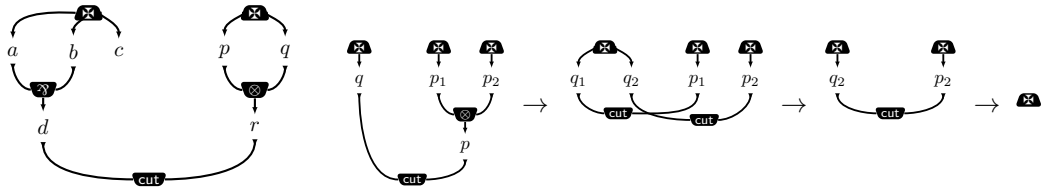
The notion of ordered hypergraph and *arrangement* introduced in Section 1 will now explicitly come into play as it is necessary for defining the interactions of nets (see Figure 9a). We will denote by $\#S$ the number of outputs of a net S. Given a partial function $f : \mathbb{N} \to E$ with a finite domain of cardinality n and ordered as $i_1 < i_2 < \cdots < i_n$, the *collapse* of f, denoted $f{\downarrow}$, is the total function with domain $[1; n]$ such that $f{\downarrow}(m) = f(i_m)$ for any integer $1 \leq m \leq n$.

▶ **Definition 41.** *Let $S = (|S|, \mathbf{a}(S))$ and $T = (|T|, \mathbf{a}(T))$ be two nets and $k = min(\#S, \#T)$, we define their* interaction $S :: T = (|S :: T|, \mathbf{a}(S :: T))$ *as:*

$$|S :: T| \triangleq |S| + |T| + \sum_{1 \leq i \leq min(\#S, \#T)} \langle S(i), T(i) \rhd_{\mathsf{cut}} \rangle \qquad \mathbf{a}(S :: T) \triangleq \begin{cases} \emptyset & \text{when } \#S = \#T \\ \mathbf{a}(S)\ \restriction_{[k+1; \#S]}{\downarrow} & \text{when } \#S > \#T \\ \mathbf{a}(T)\ \restriction_{[k+1; \#T]}{\downarrow} & \text{when } \#S < \#T \end{cases}$$

[6] Recall that a multigraph is a graph where two vertices may be connected by several edges (not to be confused with the notion of hypergraph of Definition 1). The function brd maps each edge to its endpoints.

[7] We refer to the graph naturally induced by the net σS.



(a) Representation of the interaction $S :: T$ of two nets $S = \langle \triangleright_{\text{✠}}\ a, b, c\rangle + \langle a, b \triangleright_{⅋}\ d\rangle$ and $T = \langle \triangleright_{\text{✠}}\ p, q\rangle + \langle p, q \triangleright_{\otimes}\ r\rangle$.

(b) The cut elimination procedure applied to $S' :: T'$ leads to ✠_0, showing that $S' \perp T'$. In this figure $S' = \langle \triangleright_{\text{✠}}\ q\rangle$ and $T' = \langle \triangleright_{\text{✠}}\ p_1\rangle + \langle \triangleright_{\text{✠}}\ p_2\rangle + \langle p_1, p_2 \triangleright_{\otimes}\ p\rangle$.

Figure 9 The interaction of two nets (Definition 41) and two orthogonal nets (Definition 42).

▶ **Definition 42.** *Two nets S_1 and S_2 are* orthogonal *if $S_1 :: S_2 \to^* \text{✠}_0$[8]: when this holds we write $S_1 \perp S_2$. For a net S and a set of nets Λ, if for every $\lambda \in \Lambda$ we have $S \perp \lambda$ we write $S \perp \Lambda$.*

▶ Remark 43. Since cut links are asymmetric, namely $\langle p, q \triangleright_{\mathsf{cut}}\rangle$ and $\langle q, p \triangleright_{\mathsf{cut}}\rangle$ are distinct nets, the interactions $S :: T$ and $T :: S$ are not the same net. However, this has no consequence on cut elimination because the reduction steps do not depend on the order of the inputs of a cut link. Thus $S :: T$ reduces to ✠_0 if and only if $T :: S$ does, and as expected the relation of orthogonality is symmetric.

▶ **Definition 44.** *Given a set A of multiplicative nets, we define the* orthogonal *of A as $A^\perp = \{P \mid \forall R \in A, P \perp R\}$. A* type $\mathbf{A}$ *is a set of multiplicative nets such that $\mathbf{A}^{\perp\perp} = \mathbf{A}$.*[9]

▶ Remark 45. Since cut elimination preserves the conclusions of a net and ✠_0 has no output, two orthogonal nets have the same number of conclusions. Thus, for every type $\mathbf{A}$, for every $R \in \mathbf{A}$ and for every $S \in \mathbf{A}^\perp$, the nets R and S have the same number of conclusions: we denote by $\#\mathbf{A}$ the number of conclusions of the nets in $\mathbf{A}$. Obviously $\#\mathbf{A} = \#\mathbf{A}^\perp$.

▶ Remark 46. Clash cuts are preserved during cut elimination, thus a net containing such a cut cannot reduce to ✠_0. Hence, there cannot be two nets S and S' respectively in $\mathbf{A}$ and $\mathbf{A}^\perp$ such that their ith conclusions $S(i)$ and $S'(i)$ are both outputs of a ⅋–link (or $\otimes$–link): their interaction $S :: S'$ contains a clash cut and thus the nets cannot be orthogonal.

▶ Remark 47. A net S which is orthogonal to the daimon link with a single output (i.e. ✠_1) has a single conclusion which can be the output of a daimon link, a tensor link or a par link. For instance the three cut–free nets $\langle \triangleright_{\text{✠}}\ p\rangle$, $\langle \triangleright_{\text{✠}}\ p_1\rangle + \langle \triangleright_{\text{✠}}\ p_2\rangle + \langle p_1, p_2 \triangleright_{\otimes}\ p\rangle$ and $\langle \triangleright_{\text{✠}}\ p_1, p_2\rangle + \langle p_1, p_2 \triangleright_{⅋}\ p\rangle$ are all orthogonal to ✠_1 (one case is proved in Figure 9b).

The following proposition is a key step for proving propositions 51 and 54.

▶ **Proposition 48.** *Given three net S and T and R such that $\#S \geq \#T + \#R$: the interaction $S :: (T \parallel R)$ is equal to $(S :: T) :: R$.*

In the following definition 49 the side condition $\#S \geq \#\mathbf{A}$ ensures that whenever a net S in $A \succ B$ interacts with a net of $T \in \mathbf{A}^\perp$ the remaining conclusions of $S :: T$ are conclusions of S, this will allow to activate Proposition 48.

[8] Note that we require the *existence* of such a reduction, not all reductions need to behave this way.
[9] Equivalently, a type is a set $\mathbf{A}$ such that $\mathbf{A} = B^\perp$ for some set B, see, for instance, [11].

▶ **Definition 49.** *Given two sets of nets* $\mathbf{A}$ *and* $\mathbf{B}$ *their* functional composition *denoted* $\mathbf{A} \succ \mathbf{B}$*, and their* parallel composition *denoted* $\mathbf{A} \parallel \mathbf{B}$ *are defined as follows:*

$$\mathbf{A} \succ \mathbf{B} \triangleq \{S \mid \text{for any } T \in \mathbf{A}^\perp, S :: T \in \mathbf{B} \text{ and } \#S \geq \#\mathbf{A}\} \qquad \mathbf{A} \parallel \mathbf{B} \triangleq \{S \parallel T \mid S \in \mathbf{A}, T \in \mathbf{B}\}^{\perp\perp}$$

▶ Remark 50 (Density of the parallel composition). For any two types $\mathbf{A}$ and $\mathbf{B}$ we have $(\mathbf{A} \parallel^- \mathbf{B})^\perp = (\mathbf{A} \parallel \mathbf{B})^\perp$, where $\mathbf{A} \parallel^- \mathbf{B} = \{S \parallel T \mid S \in \mathbf{A}, T \in \mathbf{B}\}$.

▶ **Proposition 51** (Duality)**.** *Given two types* $\mathbf{A}$ *and* $\mathbf{B}$*:* $(\mathbf{A} \parallel \mathbf{B})^\perp = \mathbf{A}^\perp \succ \mathbf{B}^\perp$ *and* $(\mathbf{A} \succ \mathbf{B})^\perp = \mathbf{A}^\perp \parallel \mathbf{B}^\perp$.

▶ Remark 52. The duality of the constructions (Proposition 51) ensures that the set of types is closed under the $\parallel$ and $\succ$ operations. Moreover, the intersection of two types is still a type. This is not the case for the union which needs to be closed under bi–orthogonal.

▶ Remark 53. For two types $\mathbf{A}$ and $\mathbf{B}$ the unordered nets of $\mathbf{A} \parallel \mathbf{B}$ and of $\mathbf{B} \parallel \mathbf{A}$ are the same, so as the unordered nets of $\mathbf{A} \succ \mathbf{B}$ and $\mathbf{B} \succ \mathbf{A}$.

▶ **Proposition 54.** *Given* $\mathbf{A}, \mathbf{B}$ *and* $\mathbf{C}$ *three types;* $(\mathbf{A} \succ \mathbf{B}) \succ \mathbf{C} = \mathbf{A} \succ (\mathbf{B} \succ \mathbf{C})$ *and* $(\mathbf{A} \parallel \mathbf{B}) \parallel \mathbf{C} = \mathbf{A} \parallel (\mathbf{B} \parallel \mathbf{C})$.

▶ **Definition 55.** *Given* $\mathbf{A}$ *and* $\mathbf{B}$ *two types with one conclusion, we define their* tensor product *(denoted* $\otimes$*) and their* compositional product *(denoted* $\parr$*):*

$$\mathbf{A} \otimes \mathbf{B} \triangleq \{S + \langle S(1), S(2) \rhd_\otimes p\rangle \mid S \in \mathbf{A} \parallel \mathbf{B}\}^{\perp\perp} \qquad \mathbf{A} \parr \mathbf{B} \triangleq \{S + \langle S(1), S(2) \rhd_\parr p\rangle \mid S \in \mathbf{A} \succ \mathbf{B}\}^{\perp\perp}$$

where p *denotes a fresh position.*

▶ **Proposition 56** (Duality)**.** *Given* $\mathbf{A}$ *and* $\mathbf{B}$ *two types with one conclusion,* $(\mathbf{A} \otimes \mathbf{B})^\perp = \mathbf{A}^\perp \parr \mathbf{B}^\perp$ *and* $(\mathbf{A} \parr \mathbf{B})^\perp = \mathbf{A}^\perp \otimes \mathbf{B}^\perp$.

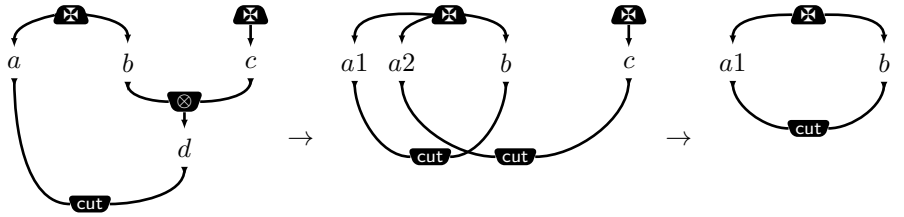
4 Realisability Model: Adequacy

We introduce our realisability model on untyped nets and prove it is adequate. We identify a sufficient property of interpretation bases to prove adequacy (Theorem 64): for any basis $\mathcal{B}$ satisfying the property, a net S representing an $\mathsf{MLL}^{\text{✠}}$ proof of a sequent Γ is a realiser of Γ i.e. it belongs to $[\![\Gamma]\!]_\mathcal{B}$. This adequacy result immediately applies to MLL, since a net representing a proof of MLL represents, in particular, a proof of $\mathsf{MLL}^{\text{✠}}$.

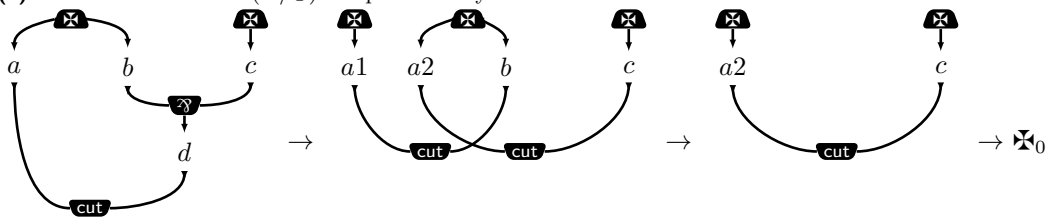
We start by giving an interpretation of formulas and hypersequents of multiplicative linear logic. We provide an interpretation of hypersequents instead of sequents as it turns out that handling hypersequents is more convenient and proving a result on hypersequents proves it on sequents too. However, do keep in mind that the proof trees we defined using Figure 6c are constructed with sequents.

▶ **Definition 57.** *An* interpretation basis $\mathcal{B}$ *is a function that associates with each atomic proposition* X *a type* $[\![X]\!]_\mathcal{B}$*, the* interpretation *of* X*, such that:*

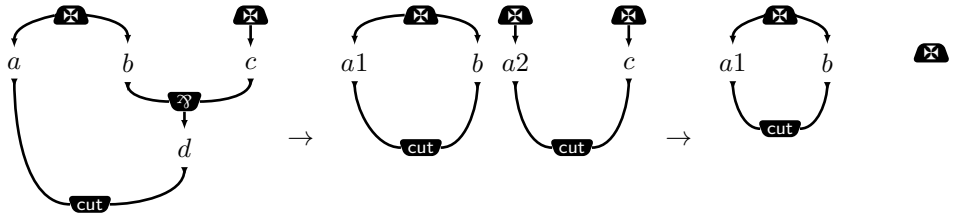
- *Each net in* $[\![X]\!]_\mathcal{B}$ *has one conclusion.*
- *For any atomic proposition* X*, we have* $[\![X^\perp]\!]_\mathcal{B} \subseteq [\![X]\!]_\mathcal{B}^\perp$.



(a) The elimination of a (✠/⊗) cut preserves cycles in a net.



(b) The elimination of a (✠/⅋) cut *may* break a cycle in a net.



(c) The elimination of a (✠/⅋) cut can preserve cycles.

Figure 10 The evolution of (switching) cycles and (switching) disconnections during non homogeneous cut elimination.

▶ **Definition 58.** *Given an interpretation basis $\mathcal{B}$, the* interpretation *of* MLL *formulas and of hypersequents of* MLL *is defined by induction:*

$$[\![A \otimes B]\!]_{\mathcal{B}} \triangleq [\![A]\!]_{\mathcal{B}} \otimes [\![B]\!]_{\mathcal{B}}. \qquad [\![\mathcal{H}_1, \mathcal{H}_2]\!]_{\mathcal{B}} \triangleq [\![\mathcal{H}_1]\!]_{\mathcal{B}} \succ [\![\mathcal{H}_2]\!]_{\mathcal{B}}.$$
$$[\![A \parr B]\!]_{\mathcal{B}} \triangleq [\![A]\!]_{\mathcal{B}} \parr [\![B]\!]_{\mathcal{B}}. \qquad [\![\mathcal{H}_1 \parallel \mathcal{H}_2]\!]_{\mathcal{B}} \triangleq [\![\mathcal{H}_1]\!]_{\mathcal{B}} \parallel [\![\mathcal{H}_2]\!]_{\mathcal{B}}.$$

▶ Remark 59. Using duality of types (Proposition 56) and the properties of orthogonality one proves that for an interpretation basis $\mathcal{B}$ and an MLL formula A we have $[\![A^\perp]\!]_{\mathcal{B}} \subseteq [\![A]\!]_{\mathcal{B}}^{\perp}$.

▶ **Definition 60.** *A multiplicative net* realises *– with respect to an interpretation basis $\mathcal{B}$ – an hypersequent $\mathcal{H}$ of* MLL *formulas whenever it belongs to $[\![\mathcal{H}]\!]_{\mathcal{B}}$.*

▶ Notation 61. For a hypersequent $\mathcal{H}$, we will often write $S \Vdash_{\mathcal{B}} \mathcal{H}$ instead of $S \in [\![\mathcal{H}]\!]_{\mathcal{B}}$, and sometimes $S \Vdash \mathcal{H}$ or $S \in [\![\mathcal{H}]\!]$ when there is no ambiguity on the basis $\mathcal{B}$.

From the point of view of cut elimination, a daimon link with n outputs may be thought as the approximation of a proof net with n outputs. More precisely, by iterating the process we have seen in Remark 24, every cut–free proof π of a formula C can be obtained by applying the cut elimination procedure to the daimon link ✠$_1$ (of conclusion C) cut against the appropriate identities of $C, C^\perp$ (this generalises to a sequent Γ and ✠$_n$). Furthermore daimon links and proof nets (with the same number of conclusions) are interchangeable with respect to geometrical correctness (Table 1): in a correct (resp. incorrect) net S, substituting a daimon link with n outputs by a proof net with n outputs produces a correct (resp. incorrect) net. However, proof nets and daimons (with the same number of conclusions) differ on realisability: for instance a proof net ending with a tensor link can never realise a formula of the form $A \parr B$ whereas a daimon link can (Theorem 64). We will thus say that a daimon link "approximates" a sequent: this suggests Definition 62.

▶ **Definition 62.** *A type* $\mathbf{A}$ *is approximable if and only if* $\maltese_1 \in \mathbf{A}$. *A basis* $\mathcal{B}$ *is* approximable *if for each* $X \in \mathsf{Var}$*, the type* $[\![X]\!]_{\mathcal{B}}$ *is approximable.*

▶ Remark 63. Because inclusion is preserved by bi–orthogonal closure, a type $\mathbf{A}$ is approximable if and only if $\{\maltese_1\}^{\perp\perp} \subseteq \mathbf{A}$ which is equivalent to the inclusion $\mathbf{A}^{\perp} \subseteq \{\maltese_1\}^{\perp}$.

▶ **Theorem 64** (Adequacy)**.** *Let* $\mathcal{B}$ *be an approximable basis. For any net* S *and sequent* Γ $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma \Rightarrow S \Vdash_{\mathcal{B}} \Gamma$.

Proof. The technique is standard in the works on realisability (see [12] or [14]): one proceeds by induction on the size of a proof π represented by S. For the base case one must show that $\maltese_n$ realises any sequent Γ with n formulas. To do so one first checks that, for any formula A, $[\![A]\!]_{\mathcal{B}}$ is approximable ($\maltese_1 \in [\![A]\!]_{\mathcal{B}}$). ◀

▶ Remark 65. An approximable basis yields adequacy, in particular, for MLL. Notice, however, that there exist bases yielding an interpretation that is adequate for MLL but not for $\mathsf{MLL}^{\maltese}$.

5 Testability and tests

The partitions involved in the Danos Regnier criterion (Theorem 76) and their orthogonality with the daimons of a net can be translated as *tests*; so that for a formula A, a net S testable by A (definition 66 below) and orthogonal to $\mathsf{tests}(A)$ is a correct net (Theorem 76). We will show that these tests are proofs of $\mathsf{MLL}^{\maltese}$ (Theorem 77). This means that for realisers in an approximable basis, testability (Definition 66) and correct typeability (Definition 31) coincide: this is Proposition 82.

▶ **Definition 66** ((Atomic) testable cut–free nets)**.** *A* formula labelling *of a cut–free net* S *is a function* $\tau : V_S \to \mathsf{Form}$ *such that:*

- ***(Par)*** *When* $\langle p_1, p_2 \rhd_{⅋} p\rangle$ *occurs in* S*: if* $\tau(p_1) = A$ *and* $\tau(p_2) = B$ *then* $\tau(p) = A ⅋ B$.
- ***(Tens)*** *When* $\langle p_1, p_2 \rhd_{\otimes} p\rangle$ *occurs in* S*: if* $\tau(p_1) = A$ *and* $\tau(p_2) = B$ *then* $\tau(p) = A \otimes B$.

A formula labelling of a cut–free net S *is* atomic *when for each daimon link* $\langle \rhd_{\maltese} p_1, \dots, p_n\rangle$ *in* S *the formula* $\tau(p_i)$ *is a propositional variable.*

A cut–free net S *with* n *conclusions is* testable *(resp.* atomic testable*) by a sequent* $\Gamma = A_1, \dots, A_n$*, which we denote* $S \eqsim \Gamma$ *(resp.* $S \overset{at}{\eqsim} \Gamma$*), if there exists a formula (resp. an atomic formula) labelling* τ *of* S *such that* $\tau(S(i)) = A_i$ *for each* $1 \leq i \leq n$.

▶ Remark 67. $S \eqsim \Gamma$ iff $S \overset{at}{\eqsim} \Delta$ and $\Gamma = \theta\Delta$ for some substitution θ and sequent Δ.

▶ Remark 68. $S \overset{at}{\eqsim} \Gamma$ iff S without its $\maltese$–links is the syntactic forest of (the formulas of) Γ.

▶ Remark 69. A cut–free proof net $S \vdash^{\maltese}_{\mathsf{MLL}} \Gamma$ is in particular testable by that sequent i.e. $S \eqsim \Gamma$. However, a net $S \eqsim \Gamma$ which is testable by Γ may not be a proof net because it could contain cycles or disconnections: the testability condition only provides information on the multiplicative links constituting the net S. When is S atomic testable by A, orthogonality with the tests of A coincides with correctness (Proposition 75).

▶ Remark 70. Let $S \overset{at}{\eqsim} A_1, \dots, A_n$ be a cut–free net. For any nets $T_1, \dots, T_n$ cut–free and atomically testable respectively by ${A_1}^{\perp}, \dots, {A_n}^{\perp}$ denoting S_0 the normal form of $S :: T_1 \parallel \cdots \parallel T_n$, S_0 is obtained by homogeneous cut–elimination, and we have (1) S_0 equals $\maltese_0$ (2) S_0 is equal to the sum of $k \geq 2$ daimon without conclusions ($S_0 = \sum_{1 \leq i \leq k} \maltese_0$) or (3) S_0 contains a cyclic cut ($S_0 = R + \langle \rhd_{\maltese} \vec{q}, a, \vec{r}, b, \vec{p}\rangle + \langle a, b \rhd_{\mathsf{cut}}\rangle$).

▶ Remark 71. Given a net $S = (|S|, \mathbf{a}(S))$ we denote $S^{\maltese} = (\left|S^{\maltese}\right|, \mathbf{a}(S^{\maltese}))$ the net such that $\left|S^{\maltese}\right|$ is the hypergraph consisting of the daimon links occurring in S. The arrangement $\mathbf{a}(S^{\maltese})$ is induced by $\mathbf{a}(S)$ because above every conclusion of S there is binary tree: each initial position p can be associated with a sequence $\xi = \mathsf{adr}(p)$ of $\{\ell, r\}^*$ and an integer $i = \mathsf{root}(p)$ so that going up from $S(i)$ following the left/right instruction of ξ one reaches the initial position p. The initial positions of S are then ordered by the lexicographical order of $(\mathsf{root}(p), \mathsf{adr}(p))$ fixing $\ell \leq r$.

▶ Notation 72. Given a net S with n initial positions, and $P = \{C_1, \ldots, C_k\}$ a partition of the initial positions of S we denote by $\mathbf{Nat}_S(P)$ the partition $\{\mathbf{a}(S^{\maltese})^{-1}(C_1), \ldots, \mathbf{a}(S^{\maltese})^{-1}(C_k)\}$ of $\{1, \ldots, n\}$. We might abusively write $\mathbf{Nat}(P)$ for $\mathbf{Nat}_S(P)$.

▶ **Proposition 73.** *Let A be a formula, given two cut free nets $S \overset{at}{\vDash} A$ and $T \overset{at}{\vDash} A^\perp$ the assertions are equivalent:*
1. *The nets S and T are orthogonal.*
2. *The nets $S^{\maltese}$ and $T^{\maltese}$ are orthogonal.*
3. *The partition $\mathbf{Nat}_S(\mathsf{P}_{\maltese}(S))$ and $\mathbf{Nat}_T(\mathsf{P}_{\maltese}(T))$ are orthogonal.*

▶ **Definition 74.** *A cut-free net T is a* test *of a formula A if $T \overset{at}{\vDash} A^\perp$ and there exists a net $S \overset{at}{\vDash} A$ and a switching σS such that $\mathbf{Nat}_T(\mathsf{P}_{\maltese}(T)) = \mathbf{Nat}_S(\uparrow^i \sigma S)$. We denote by* $\mathsf{tests}(A)$ *the set $\{S \mid S$ is a test of $A\}$.*

▶ **Proposition 75.** *For S cut–free, $S \overset{at}{\vDash} A$, we have: $S \vdash_{\mathsf{MLL}^{\maltese}} A \Leftrightarrow S \perp \mathsf{tests}(A)$.*

A net S with n conclusion can always be transformed in a net with 1 conclusion by putting a bunch of par–links below its conclusions; this allows to generalise the previous proposition.

▶ **Theorem 76** (Danos–Regnier Tests). *Given a cut–free net $S \overset{at}{\vDash} A_1, \ldots, A_n$; $S \vdash_{\mathsf{MLL}^{\maltese}} A_1, \ldots, A_n$ if and only if S is orthogonal to $\mathsf{tests}(A_1) \parallel \cdots \parallel \mathsf{tests}(A_n)$.*

▶ **Theorem 77.** *Any test T of a formula A is correctly typeable by $A^\perp$, $T \vdash_{\mathsf{MLL}^{\maltese}} A^\perp$.*

Proof. Consider a test T of A then by Theorem 76 any net $S \vdash_{\mathsf{MLL}^{\maltese}} A$ is orthogonal to T. By the counter–proof criterion [4] a net $N \overset{at}{\vDash} A^\perp$ orthogonal to each proof of A is a proof; therefore it follows that T is a proof of $A^\perp$. ◀

▶ Remark 78. Theorem 76 is a refinement of the counter–proof criterion of P.L. Curien [4]: if $S \overset{at}{\vDash} A$ and $S \perp \mathsf{tests}(A)$ then $S \vdash_{\mathsf{MLL}^{\maltese}} A$ – and every element of $\mathsf{tests}(A)$ are proofs of $A^\perp$ (Theorem 77), but the converse does not hold.

From Theorem 76 and Theorem 77 one obtains an “interactive” criterion for the nets of multiplicative linear logic (MLL). One takes a net of S of MLL (i.e. a net with binary daimons) and confronts it with the tests of the according formulas (Definition 74). A straightforward consequence of the Theorem 76 is the reformulation of Béchet’s theorem in our framework.

▶ **Corollary 79.** *Let $S \overset{at}{\vDash} A_1, \ldots, A_n$ be a cut–free net. If S is not correct then there exists nets $T_1 \in \mathsf{tests}(A_1), \ldots, T_n \in \mathsf{tests}(A_n)$ such that the normal form of $S :: T_1 \parallel \cdots \parallel T_n$ is not correct: we are in case (2) or (3) of Remark 70.*

▶ Remark 80. The Corollary 79 obviously applies to MLL nets, the main difference with Béchet’s original result is that his opponents are MLL proof nets (in our framework they are $\mathsf{MLL}^{\maltese}$ proof nets). However it is not difficult to adapt our techniques to obtain Béchet’s result.

▶ Remark 81. Consider an approximable basis $\mathcal{B}$ and a sequent $\Gamma = A_1, \dots, A_n$ we have $[\![\Gamma]\!]_{\mathcal{B}} = ([\![A_1]\!]^{\perp}_{\mathcal{B}} \parallel \cdots \parallel [\![A_n]\!]^{\perp}_{\mathcal{B}})^{\perp}$. By Theorem 64, for any $A_i^{\perp}$ we have $\{\!|A_i^{\perp} : \mathsf{MLL}^{\maltese}|\!\} \subseteq [\![A_i^{\perp}]\!]_{\mathcal{B}}$ while $\mathsf{tests}(A_i) \subseteq \{\!|A_i^{\perp} : \mathsf{MLL}^{\maltese}|\!\}$ (Theorem 77) thus $\mathsf{tests}(A_i) \subseteq [\![A_i^{\perp}]\!]_{\mathcal{B}} \subseteq [\![A_i]\!]^{\perp}_{\mathcal{B}}$ (Remark 59). Because the $\parallel$–construction preserves inclusions and orthogonality inverts inclusions we derive that $[\![\Gamma]\!]_{\mathcal{B}} \subseteq (\mathsf{tests}(A_1) \parallel \cdots \parallel \mathsf{tests}(A_n))^{\perp}$.

Remark 81 combined with the previous theorem (Theorem 76) means that for realisers in an approximable basis, testability and (correct) typeability collapse.

▶ **Proposition 82.** *Given $\mathcal{B}$ an approximable basis*[10] *and a sequent Γ for any cut–free net $S \in [\![\Gamma]\!]_{\mathcal{B}}$ the assertions are equivalent:*

1. $S \vDash \Gamma$ *i.e.* $S \overset{at}{\bowtie} \Delta$ *for some sequent* $\Delta \leq \Gamma$.
2. $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$.

6 Completeness

Using Proposition 82 we provide a completeness result; we exhibit an approximable basis for which a net S realising a sequent Γ is testable, and so equivalently $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$. This basis, denoted $\mathbf{1}$, maps each atomic formula to $\{\maltese_1\}^{\perp\perp}$.

▶ **Proposition 83.** *For any sequent Γ and any cut–free net S; if $S \in [\![\Gamma]\!]_{\mathbf{1}}$ then $S \vDash \Gamma$.*

▶ Remark 84. By the Proposition 83 and the Theorem 64 we have that $S \in [\![\Gamma]\!]_{\mathbf{1}}$ iff $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$.

Since the base $\mathbf{1}$ is approximable, Proposition 82 allows to prove:

▶ **Theorem 85** ($\mathsf{MLL}^{\maltese}$ completeness)**.** *Given a cut–free net S and a sequent Γ;*

- *If for all basis $\mathcal{B}$ we have $S \in [\![\Gamma]\!]_{\mathcal{B}}$, then $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$.*
- *$S \in [\![\Gamma]\!]_{\mathcal{B}}$ for any approximable basis $\mathcal{B}$ iff $S \vdash_{\mathsf{MLL}^{\maltese}} \Gamma$.*

▶ Remark 86. The non homogeneous cut elimination allows to distinguish the types $[\![X, X^{\perp}]\!]_{\mathcal{B}}$ and $[\![X, Y]\!]_{\mathcal{B}}$ for a well chosen basis: for instance for the basis, that we will denote $\mathcal{B}\langle ⅋ \rangle$, which maps positive propositional variables to $\{\maltese_{⅋}\}^{\perp}$ and negative propositional variables to $\{\maltese_{⅋}\}^{\perp\perp}$, where $\maltese_{⅋}$ denotes the geometrically incorrect net $\langle \rhd_{\maltese} a \rangle + \langle \rhd_{\maltese} b \rangle + \langle a, b \rhd_{⅋} c \rangle$.

In that case, (1) because $\maltese_2$ is not orthogonal to $\maltese_{⅋} \parallel \maltese_{⅋}$ (Figure 11) it follows that $\maltese_2 \notin [\![X, X]\!]_{\mathcal{B}\langle ⅋ \rangle}$ and more generally $\maltese_2 \notin [\![X, Y]\!]_{\mathcal{B}\langle ⅋ \rangle}$; (2) by the property expressed in Remark 90 (and illustrated in Figure 12), $\maltese_2 \in [\![X, X^{\perp}]\!]_{\mathcal{B}\langle ⅋ \rangle}$; (3) point (1) above is not in contradiction with the theorem of adequacy (Theorem 64) because, even though $\maltese_2 \vdash_{\mathsf{MLL}^{\maltese}} X, Y$, the basis $\mathcal{B}\langle ⅋ \rangle$ is not approximable.

▶ Remark 87. The ability to distinguish realisers of the sequents $X, X^{\perp}$ and X, Y (Remark 86) allows us to derive the completeness result for MLL (Theorem 88) from the completeness result for $\mathsf{MLL}^{\maltese}$ (Theorem 85). In Remark 86, to show that $\maltese_2 \notin [\![X, Y]\!]_{\mathcal{B}\langle ⅋ \rangle}$ we have used incorrect nets (specifically $\maltese_{⅋}$), which explains that the completeness theorem for MLL (Theorem 88) refers to *any* basis $\mathcal{B}$ (and not only to approximable basis). In the terms of Table 1, we retrieve provability correctness by using interactions with geometrically incorrect nets.

▶ **Theorem 88** (MLL completeness)**.** *Let S be a cut–free net such that each of its daimon link has exactly two outputs, Γ be a sequent such that $S \overset{at}{\bowtie} \Gamma$; if $S \in [\![\Gamma]\!]_{\mathcal{B}}$ for any basis $\mathcal{B}$ then, $S \vdash_{\mathsf{MLL}} \Gamma$.*

[10] The Proposition 82 actually holds for any "adequate" basis $\mathcal{B}$.

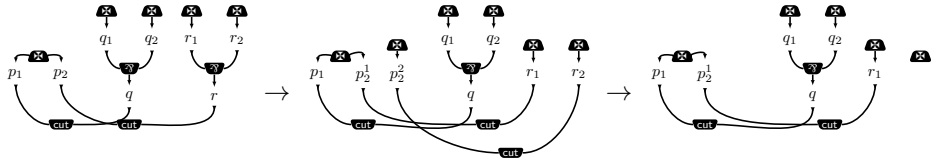


Figure 11 The daimon link ✠_2 is not orthogonal to $\text{✠}_{⅋} \parallel \text{✠}_{⅋}$: a disconnected net never reduces to a connected one (and ✠_0 is connected).

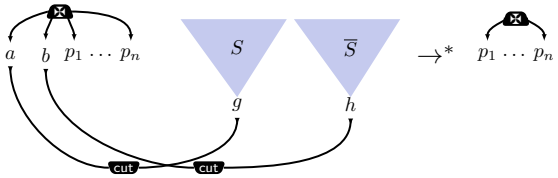


Figure 12 The interaction of two orthogonal nets S and $\overline{S}$ with a daimon reduces to a daimon (with two less outputs).

▶ Remark 89. A result of adequacy for MLL can also be stated: given an interpretation basis $\mathcal{B}$ (not necessarily approximable) such that for each propositional variable X we have $[\![X^\perp]\!]_\mathcal{B} = [\![X]\!]_\mathcal{B}^\perp$, for any net S, if $S \vdash_{\mathsf{MLL}} \Gamma$ then $S \in [\![\Gamma]\!]_\mathcal{B}$.

▶ Remark 90. The completeness result for MLL (Theorem 88) only identifies cut–free and *atomic* proofs (i.e. where axioms introduce sequents of the form $X, X^\perp$). This is because for any atomic formulas X and Y, and for any basis $\mathcal{B}$ such that $[\![X^\perp]\!]_\mathcal{B} = [\![X]\!]_\mathcal{B}^\perp$, $\text{✠}_2 \in [\![X ⅋ X^\perp, Y ⅋ Y^\perp]\!]_\mathcal{B}$ while $X ⅋ X^\perp$ and $Y ⅋ Y^\perp$ are not dual formulas: contrary to the atomic case we cannot use ✠_2 to distinguish $[\![X ⅋ X^\perp, Y ⅋ Y^\perp]\!]_{\mathcal{B}\langle⅋\rangle}$ from $[\![X ⅋ X^\perp, X^\perp \otimes X]\!]_{\mathcal{B}\langle⅋\rangle}$.

The fact that $\text{✠}_2 \in [\![X ⅋ X^\perp, Y ⅋ Y^\perp]\!]_{\mathcal{B}\langle⅋\rangle}$ (and more generally for any basis $\mathcal{B}$ such that $[\![X^\perp]\!]_\mathcal{B} = [\![X]\!]_\mathcal{B}^\perp$) is derived from the fact that, for any integer k and for any two orthogonal nets S_1 and S_2 with one conclusion, the interaction $\text{✠}_{k+2} :: (S_1 \parallel S_2)$ has *at least one* reduction to ✠_k by cut elimination (Figure 12). We use this property for $k = 2$ and $k = 4$ to show that $\text{✠}_2 \in [\![X ⅋ X^\perp, Y ⅋ Y^\perp]\!]_\mathcal{B}$. More precisely, we prove that, $\text{✠}_2 \perp [\![X ⅋ X^\perp]\!]_\mathcal{B}^\perp \parallel [\![Y ⅋ Y^\perp]\!]_\mathcal{B}^\perp$: given $S, \overline{S}$ and $R, \overline{R}$ two pairs of orthogonal nets (with one conclusion), when all nets $S, \overline{S}, R, \overline{R}$ have disjoint sets of vertices, we can derive the following:

$$\begin{aligned}
& \langle \rhd_{\text{✠}} a, b\rangle :: S + \overline{S} + \langle S(1), \overline{S}(1) \rhd_\otimes q\rangle + R + \overline{R} + \langle R(1), \overline{R}(1) \rhd_\otimes r\rangle \\
\to \cdot \to \quad & \langle \rhd_{\text{✠}} a_1, a_2, b_1, b_2\rangle :: S + \overline{S} + R + \overline{R} \\
\to^* \quad & \langle \rhd_{\text{✠}} b_1, b_2\rangle :: R + \overline{R} \\
\to^* \quad & \text{✠}_0
\end{aligned}$$

References

1 Denis Bechet. Minimality of the correctness criterion for multiplicative proof nets. *Mathematical Structures in Computer Science*, 8(6):543–558, 1998. URL: `http://journals.cambridge.org/action/displayAbstract?aid=44779`, `doi:10.1017/S096012959800262X`.

2 Emmanuel Beffara. A concurrent model for linear logic. *Electronic Notes in Theoretical Computer Science*, 155:147–168, 2006. Proceedings of the 21st Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXI). `doi:10.1016/j.entcs.2005.11.055`.

3 Emmanuel Beffara, Félix Castro, Mauricio Guillermo, and Étienne Miquey. Concurrent realizability on conjunctive structures. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction, FSCD 2023, July 3-6, 2023, Rome, Italy*, volume 260 of *LIPIcs*, pages 28:1–28:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.FSCD.2023.28.

4 Pierre-Louis Curien. Introduction to linear logic and ludics, part II. *CoRR*, abs/cs/0501039, 2005. arXiv:cs/0501039.

5 Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28(3):181–203, October 1989. doi:10.1007/BF01622878.

6 Valeria C. V. de Paiva. A dialectica-like model of linear logic. In David H. Pitt, David E. Rydeheard, Peter Dybjer, Andrew M. Pitts, and Axel Poigné, editors, *Category Theory and Computer Science*, pages 341–356, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg. doi:10.1007/BFB0018360.

7 Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–101, 1987. doi:10.1016/0304-3975(87)90045-4.

8 Jean-Yves Girard. Multiplicatives. In G. Lolli, editor, *Logic and Computer Science: New Trends and Applications*, pages 11–34. Rosenberg & Sellier, 1987.

9 Jean-Yves Girard. Proof-nets: The parallel syntax for proof-theory. In *Logic and Algebra*, pages 97–124. Marcel Dekker, 1996.

10 Jean-Yves Girard. Locus solum: From the rules of logic to the logic of rules. In Laurent Fribourg, editor, *Computer Science Logic*, pages 38–38, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

11 Jean-Baptiste Joinet and Thomas Seiller. From abstraction and indiscernibility to classification and types: revisiting hermann weyl's theory of ideal elements. *Kagaku tetsugaku*, 53(2):65–93, 2021. doi:10.4216/jpssj.53.2_65.

12 Jean-Louis Krivine. Realizability in classical logic. *Panoramas et synthèses*, 27:197–229, 2005. URL: https://hal.science/hal-00154500.

13 International Research Network (IRN) Linear Logic. *Handbook of Linear Logic*. International Research Network (IRN) Linear Logic, 2023. URL: https://ll-handbook.frama.io/ll-handbook/ll-handbook-public.pdf.

14 Paulo Oliva. Modified realizability interpretation of classical linear logic. In *22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*, pages 431–442, 2007. doi:10.1109/LICS.2007.32.

15 Thomas Seiller. Interaction graphs: Multiplicatives. *Annals of Pure and Applied Logic*, 163(12):1808–1837, 2012. doi:10.1016/j.apal.2012.04.005.

16 Thomas Seiller. Interaction graphs: Exponentials. *Log. Methods Comput. Sci.*, 15, 2013.

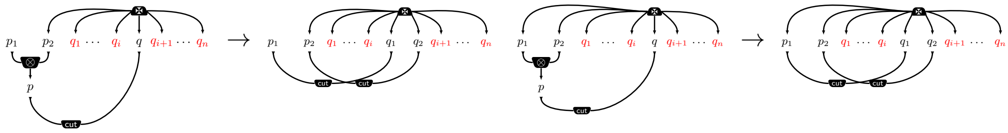
17 Thomas Seiller. Interaction graphs: Full linear logic. *CoRR*, abs/1504.04152, 2015. arXiv:1504.04152.

18 Thomas Seiller. Interaction graphs: Additives. *Annals of Pure and Applied Logic*, 167(2):95–154, 2016. doi:10.1016/j.apal.2015.10.001.

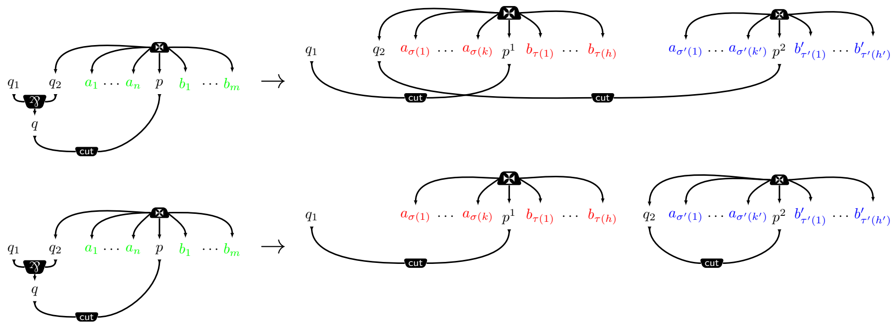
19 Thomas Seiller. Interaction graphs: Graphings. *Annals of Pure and Applied Logic*, 168(2):278–320, 2017. doi:10.1016/j.apal.2016.10.007.

20 Thomas Seiller. Mathematical informatics, 2024. Habilitation thesis. URL: https://theses.hal.science/tel-04616661.

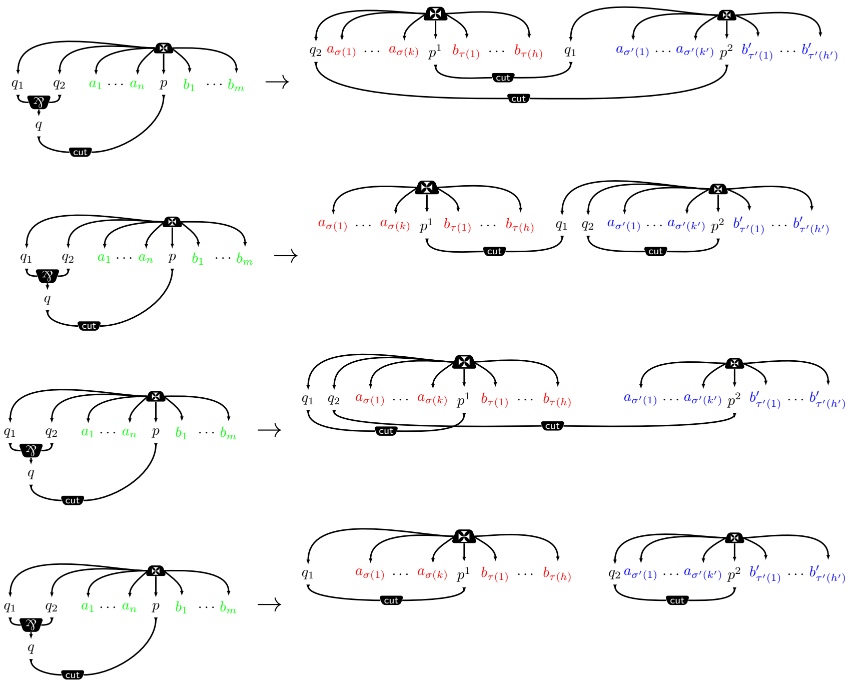
A Additional Figures



(a) Extra cases for the elimination of (✠/⊗) cuts, on the left the elimination step when one of the inputs belongs to the daimon above the cut, on the right the elimination step when both inputs belong to the daimon above the cut.



(b) Extra cases for the elimination of (✠/⅋) cuts: when one of the inputs belongs to the daimon above the cut.



(c) Extra cases for the elimination of (✠/⅋) cuts: when both inputs belong to the daimon above the cut.

Figure 13 Complements to Figure 4 for defining non homogeneous cut elimination (Definition 22).

Classical Linear Logic in Perfect Banach Lattices

Pedro H. Azevedo de Amorim
Oxford University, UK

Leon Witzman
Nanyang Technological University, Singapore

Dexter Kozen
Cornell University, Ithaca, NY, USA

Abstract

In recent years, researchers have proposed various models of linear logic with strong connections to measure theory, with *probabilistic coherence spaces* (**PCoh**) being one of the most prominent. One of the main limitations of the **PCoh** model is that it cannot interpret continuous measures. To overcome this obstacle, Ehrhard has extended **PCoh** to a category of positive cones and linear Scott-continuous functions and shown that it is a model of intuitionistic linear logic. In this work we show that the category $\mathbf{PBanLat_1}$ of perfect Banach lattices and positive linear functions of norm at most 1 can serve the same purpose, with some added benefits. We show that $\mathbf{PBanLat_1}$ is a model of classical linear logic (without exponential) and that **PCoh** embeds fully and faithfully in $\mathbf{PBanLat_1}$ while preserving the monoidal and $*$-autonomous structures. Finally, we show how $\mathbf{PBanLat_1}$ can be used to give semantics to a higher-order probabilistic programming language.

2012 ACM Subject Classification Theory of computation → Categorical semantics; Theory of computation → Linear logic; Theory of computation → Denotational semantics; Theory of computation → Probabilistic computation

Keywords and phrases Probabilistic Semantics, Linear Logic, Categorical Semantics

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.44

Funding *Pedro H. Azevedo de Amorim*: Pedro H. Azevedo de Amorim was funded by the National Science Foundation under grant CCF-2008083. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.
Dexter Kozen: Dexter Kozen was funded by the National Science Foundation under grants AitF-1637532, SaTC-1717581, and CCF-2008083. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Acknowledgements The authors would like to thank Raphaëlle Crubillé, Christine Tasson, Thomas Ehrhard and Fredrik Dahlqvist for lively discussions on the subject. We would also like to thank Arthur Azevedo de Amorim and Michael Roberts for reading an earlier draft of this work.

1 Introduction

Recent work has shown that linear logic has deep connections to the semantics of probabilistic programming languages [8, 13, 10, 12, 11, 25, 9, 19, 7]. By using monoidal closed categories instead of cartesian closed categories, linear logic provides an alternative categorical framework for higher-order functions. This was foreshadowed in early work on probabilistic semantics [20] in which bounded linear operators on Banach lattices were used to interpret a first-order imperative probabilistic programming language. This can be seen as evidence that a linear approach might be a natural alternative to cartesian closed categories.

Since then, many probabilistically-flavored models of linear logic have appeared. For instance, the connection between the early work of Kozen [20] and linear logic has been recently made precise by Dahlqvist and Kozen [7], where the category of regular ordered Banach spaces and regular maps (**RoBan**) was used to extend the semantics of Kozen [20] with higher-order functions. They also showed that **RoBan** is a model of intuitionistic linear logic.

An appealing aspect of the **RoBan** model is that ordered Banach spaces are mathematically well-understood objects with a well-developed classical theory, thus providing a plethora of useful theorems to reason about programs. This is illustrated by Dahlqvist and Kozen [7] by using results from ergodic theory to prove the correctness of a Gibbs sampling algorithm implemented in a higher-order language. However, the programming model supported by the semantics is somewhat brittle, in that the soundness of the system depends on a tricky interaction between three different type grammars with several syntactic restrictions.

A different approach was taken by Ehrhard and Danos [8], in which a category **PCoh** was defined and shown to be a model of classical linear logic. The model was used to interpret a version of PCF extended with discrete probabilities [13]. Although this category handles discrete probabilities very nicely, it cannot interpret continuous distributions such as the normal distribution over $\mathbb{R}$, a severe limitation for real-world applications. To remedy this, a category of positive cones with measurability paths and linear Scott-continuous functions $\mathbf{CLin}_m$ has recently been introduced and shown to be a conservative extension of the intuitionistic fragment of **PCoh** [6].

From a programming point of view, the language of Ehrhard et al. [12] is an extension of the simply typed λ-calculus with recursion, making it a simple and expressive programming model. However, the definition of positive cone with measurability paths deviates from standard objects from the probability literature and thus would require a large amount of mathematical effort to rephrase useful theorems that could be used to reason about programs.

Although these previous approaches are valuable contributions to our understanding of higher-order probabilistic programming through linear logic, missing up to now is a comprehensive model that embodies the following desirable aspects:

- extends **PCoh** to admit continuous measures;
- is a model of classical (not just intuitionistic) linear logic, thus allowing it to handle other computational interpretations of linear logic such as session types;
- has a simple and expressive programming model that can handle higher-order computation;
- is based on well-understood classical structures from measure theory and functional analysis.

In this paper we propose such a model. Our model extends **PCoh** with continuous probabilities and satisfies all of the properties above. Our model is based on complete normed vector lattices, called *Banach lattices*. To accommodate the second point, we work with spaces with an involutive linear negation, the so-called *perfect spaces*.

Compared to previous models, our model has simpler tensor product, which we believe lead to a more perspicuous and theoretically satisfying generalization of **PCoh**. For example, we invite a comparison with $\mathbf{CLin}_m$, where the construction rely on categorical machinery which, though elegant, are indirect.

Most importantly, Banach lattices can be seen as an abstraction of ordinary measure spaces and are well-studied in functional analysis, with many results from measure theory holding for certain classes of Banach lattices. There is a vast literature on the subject; see Fremlin [14] for a thorough introduction.

In order to justify the viability of our model, we show that it can be used to interpret a recently introduced higher-order probabilistic calculus [2], and we extend the core calculus with recursion.

Summary of contributions

- In §3, we define the category $\mathbf{PBanLat_1}$ of perfect Banach lattices and order-continuous positive linear operators with norm at most 1 and show that it is a model of classical linear logic.
- In §4, we show that there is a full and faithful monoidal closed functor $\mathbf{PCoh} \to \mathbf{PBanLat_1}$. This is a more adequate extension than the model $\mathbf{CLin}_m$ proposed by Ehrhard [11], since it also accommodates the classical aspects of the linear structure of $\mathbf{PCoh}$.
- In §5, we show that $\mathbf{PBanLat_1}$ is isomorphic to a category of lattices of positive complete cones.
- In §6, we show that $\mathbf{PBanLat_1}$ is a model to the recently defined calculus by Azevedo de Amorim [2].

Our work contributes both to the study of quantitative models of linear logic as well as to a deeper understanding of higher-order probability theory, shedding light on the importance of linear logic as a vehicle to interpret higher-order programs without cartesian closure.

2 Riesz spaces

Our model depends on technical definitions and constructions from the vector lattice literature. This section contains a brief self-contained presentation of the subject. We point the interested reader to introductory texts [1, 26] for good presentations of much of the material presented in this section.

Although we are primarily interested in Banach lattices – normed vector lattices with a completeness property – we start by defining the objects in the general unnormed case.

▶ **Definition 1.** *Let* $\mathbb{R}_+ = \{a \in \mathbb{R} \mid a \geq 0\}$. *A* Riesz space *is a partially-ordered vector space* $(V, \leq)$ *over* $\mathbb{R}$ *such that*

- *if* $x \leq y$, *then* $x + w \leq y + w$;
- *if* $x \leq y$, *then* $\alpha x \leq \alpha y$ *for* $\alpha \in \mathbb{R}_+$; *and*
- *it is an upper semilattice with respect to* $\leq$ *with join operation* $\vee$.

It follows that the space is also a lattice with meet operation $x \wedge y = -(-x \vee -y)$.

Many standard vector spaces are Riesz spaces.

▶ **Example 2.** The following are Riesz spaces:

- $\mathbb{R}^n$ with the pointwise ordering;
- the set of bounded sequences of real numbers with pointwise ordering;
- the set of signed measures on a measurable space;
- the set of bounded measurable functions on a measurable space.

Unlike the real numbers, there are elements that are neither negative nor positive, but a notable characteristic of Riesz spaces is that every element decomposes uniquely into its positive and negative parts.

▶ **Definition 3.** *For* v *an element of a Riesz space, define* $v^+ = v \vee 0$, $v^- = (-v) \vee 0$ *and* $|v| = v \vee -v = v^+ + v^-$.

Then v^+ and v^- are the unique positive elements such that $v = v^+ - v^-$ and $v^+ \wedge v^- = 0$. Thus Riesz spaces are completely characterized by their positive elements. This often simplifies constructions, as one can often prove a property for the positive elements, then extend to the entire space using this decomposition.

Given a Riesz space V, let V^+ denote the set of positive elements of V. Using the decomposition property mentioned above, it follows that $V = V^+ - V^+$, where $-$ applied to sets denotes elementwise subtraction.

2.1 Order convergence

Every topology gives rise to a notion of convergence. For normed spaces, one usually studies convergence in the norm topology. However, ordered spaces also carry an *order topology.*

▶ **Definition 4.** *Let D be a directed set and V a Riesz space. A* net $\{v_\alpha\}_{\alpha \in D}$ *is a function $D \to V$. We say that the net is increasing (respectively, decreasing) and write $\{v_\alpha\}\uparrow$ (respectively, $\{v_\alpha\}\downarrow$) if $\alpha \leq_D \beta$ implies $v_\alpha \leq_V v_\beta$ (respectively, $v_\alpha \geq_V v_\beta$).*

▶ **Definition 5.** *Given a decreasing net $\{x_\alpha\}$, we write $\{x_\alpha\} \downarrow 0$ if $\inf\{x_\alpha\} = 0$.*

▶ **Definition 6** (Order convergence)**.** *We say that a net $\{x_\alpha\}$* converges in order *to x and write $x_\alpha \xrightarrow{o} x$ if there is a decreasing net $\{y_\alpha\} \downarrow 0$ such that for all α, $|x_\alpha - x| \leq y_\alpha$.*

In general, this notion of convergence is neither weaker nor stronger than convergence in the norm topology. However, when a net converges in both order and norm, it converges to the same value in both. When it is clear from the context, we will denote order convergence as $\to$.

2.2 Riesz subspaces, solids, ideals and bands

In the theory of Riesz spaces, there are classes of subspaces that have many interesting properties that will be used in our constructions.

▶ **Definition 7.** *A subset S of a Riesz space is*

- solid *if $x \in S$ and $|y| \leq |x|$ implies $y \in S$,*
- *an* ideal *if it is a solid linear subspace,*
- *a* band *if it is an ideal and closed under existing suprema.*

▶ **Definition 8.** *We say that a Riesz space V is* Archimedean *if for every $v \in V^+$, $\{v/n\}_{n \in \mathbb{N}} \downarrow 0$. Furthermore, if every bounded subset of V admits a supremum, then we say that V is* Dedekind complete*.*

▶ **Proposition 9.** *Every band in a Dedekind complete Riesz space is Dedekind complete.*

▶ **Definition 10.** *A Riesz subspace $A \subseteq V$ is said to be* order dense *if for every element $0 < v \in V$ there is an element $a \in A$ such that $0 < a \leq v$.*

▶ **Theorem 11** ([1, Theorem 1.34])**.** *A Riesz subspace A is order dense in an Archimedean Riesz space V iff for every $v \in V^+$,*

$$\{a \in A \,|\, 0 \leq a \leq v\} \uparrow v.$$

2.3 Order-continuous functions

As usual when studying vector spaces with extra structure, we care only about linear maps that interact nicely with the extra structure. In our case, the linear functions will have to respect the partial order.

We call a linear function $f : V \to W$ *positive* if it maps positive elements of V to positive elements of W; that is, it restricts to a function $V^+ \to W^+$. A linear function is *regular* if it can be written as the difference of two positive functions.

▶ **Definition 12.** *A linear function $T : V \to W$ between Riesz spaces V and W if $Tv_\alpha \xrightarrow{o} Tv$ whenever $\{v_\alpha\}$ is an increasing net with supremum v.*

We can also characterize the positive order-continuous functions as those that preserve existing suprema and infima.

Order continuity interacts well with order density. Indeed, it is possible to show using Theorem 11 the following lemma

▶ **Lemma 13.** *If V is an Archimedian Riesz space and $f, g : V \to W$ are two linear order-continuous functions that agree on an order-dense subset of V, then $f = g$.*

This lemma will come in handy when constructing our model. Furthermore, the space of order-continuous linear functions on certain Riesz spaces are well-behaved subsets of the regular linear functions.

▶ **Theorem 14** ([1, Theorem 1.57])**.** *If W is Dedekind complete, then the set of order-continuous linear functions $V \to W$ is a band in the space of regular functions, thus forms a Dedekind-complete Riesz space.*

Proof. The Riesz space structure is given by Theorem 1.18 of Aliprantis and Burkinshaw [1]. ◀

▶ **Definition 15.** *A Riesz space is* separated *if for every distinct pair $v_1, v_2 \in V$, there exists an order-continuous linear functional $f : V \to \mathbb{R}$ such that $f(v_1) \neq f(v_2)$.*

2.4 Normed Riesz spaces

Now we will introduce normed Riesz spaces. In the context of probabilistic semantics, the norm plays an important role, as it can be used to distinguish between arbitrary measures and (sub)-probability distributions, the measures with norm at most 1.

▶ **Definition 16.** *Let V be a real vector space. A* norm *is a function $\|\cdot\| : V \to \mathbb{R}^+$ such that:*

- $\|v\| = 0$ *iff* $v = 0$
- $\|\alpha v\| = |\alpha| \, \|v\|$
- $\|v + u\| \leq \|v\| + \|u\|$.

For Riesz spaces, we require the norm to satisfy the additional property

$$|v| \leq |u| \text{ implies } \|v\| \leq \|u\|.$$

If the Riesz space is also complete with respect to the norm, we call it a *Banach lattice*. In vector space models of linear logic, the norm is typically used to distinguish between the product & and the coproduct $\oplus$, as they both have the same underlying set, but distinct norms. However, in the context of program semantics, the norm also has the extra role of allowing the interpretation of recursive programs.

▶ **Example 17.** The set $\mathcal{M}(\mathbb{R})$ of signed measures over the Borel σ-algebra on $\mathbb{R}$ is a Riesz space (cf. Section 2.6). We can equip it with the *total variation* norm $\|\mu\| = \mu^+(\mathbb{R}) + \mu^-(\mathbb{R})$.

Theorem 14 shows that by assuming the right amount of structure on the Riesz space, the set of order-continuous linear functions between Riesz spaces also has a lattice structure. It is not immediately clear whether this result generalizes to the normed case. Luckily, Dedekind completeness is once again enough.

▶ **Example 18.** Let V and W be normed Riesz spaces with W Dedekind complete. The set of order-continuous linear functions $V \to W$ can be equipped with the *regular norm*

$$\|T\|_r = \sup_{\|x\|_V \leq 1} \||T|(x)\|_W$$

where $|T|$ is given by Theorem 14 and Definition 3.

▶ **Definition 19.** *Let V be a normed Riesz space. The* closed unit ball *of V is the set $\mathcal{B}(V) = \{v \in V \mid \|v\| \leq 1\}$.*

Banach lattices

Banach lattices are normed Riesz spaces that are also Banach spaces. In the usual categorical study of Banach spaces, the relevant morphisms are the norm-continuous linear functions.

▶ **Definition 20.** *A linear function f between normed Riesz spaces V and W is said to be* norm-continuous *(or* norm-bounded*) if $\sup_{v \in \mathcal{B}(V)} \|f(v)\|$ is finite.*

Since we are interested in spaces with two distinct structures, a partial order and a norm, it is not immediately clear which class of morphisms one should care about. In general, the space of all norm-continuous linear functions between Banach lattices is not a Banach lattice, making them unable to give semantics to linear implication.

Normed Riesz spaces are also problematic, as not every order-continuous function is norm-continuous, making it unclear how one would equip the space of order-continuous functions with a norm. However, if the codomain is a Banach lattice, then every order-continuous linear function is also norm-continuous [1]. This suggests that one should work with Banach lattices but only use order-continuous linear functions.

▶ **Definition 21.** *The category* $\mathbf{BanLat_1}$ *has separated Banach lattices as objects and order-continuous positive linear functions of norm at most one as morphisms.*

These objects have been widely studied in functional analysis, being influential in the linear operator approach to measure theory [14]. A subtlety when working with a norm and a partial order is that there are two distinct notions of convergence in play that on the surface appear only tenuously related. However, a useful property has been identified in the literature that brings some harmony between the two.

▶ **Definition 22.** *A normed Riesz space is said to satisfy the* (sequential) weak Fatou property *if every norm-bounded monotone (sequence) net has a supremum.*

In the context of program semantics, the sequential version of this property has been used before to interpret recursive programs [8, 12].

▶ **Lemma 23.** *Let $f : V \to V$ be a positive order-continuous function (not necessarily linear) such that $f(\mathcal{B}(V)) \subseteq \mathcal{B}(V)$. If V satisfies the weak Fatou property, then f admits a fixpoint.*

Proof. It can be directly shown that the limit of the ω-chain $\{f^n(0)\}_{n\in\mathbb{N}}$ is a fixpoint of f. Note that when f is linear, the theorem is trivially true, since $f(0) = 0$. ◀

▶ **Lemma 24** ([14, Lemma 354B(d)]). *Every band in a Banach lattice is a Banach lattice.*

▶ **Theorem 25.** *If V and W are Banach lattices, then the set of order-continuous linear functions between V and W is a Banach lattice.*

Proof. The proof is a direct consequence of Banach lattices being Dedekind complete – e.g. Fremlin [14, Proposition 354E(e)] – and the space of order-continuous being a band in the space of regular linear functions. ◀

2.5 Dualities

The category $\mathbf{BanLat_1}$ seems to be a good candidate in which to interpret intuitionistic linear logic. However, since the linear negation connective $(-)^\perp$ is usually interpreted as the linear dual $V \multimap \mathbb{R}$ in models of linear logic based on vector spaces over $\mathbb{R}$, $\mathbf{BanLat_1}$ would not be able to model *classical* linear logic, since there are examples of Banach lattices that are not isomorphic to their bidual, e.g. summable real sequences.

A recurring challenge in models of linear logic is to make an involutive linear negation – typical of finite-dimensional spaces – coexist with $!V$, which requires infinite-dimensional spaces. Since we are interested in defining a model of classical linear logic, we should only work with Riesz spaces that are isomorphic to their bidual.

▶ **Definition 26.** *Let V^σ denote the space of order-continuous functionals $V \multimap \mathbb{R}$. A Riesz space V is said to be* perfect *if the map $\sigma_V = \lambda x f\,.\,f(x) : V \multimap V^{\sigma\sigma}$ is an isomorphism.*

We will write σ for σ_V when V is clear from context.

▶ **Definition 27.** *The category $\mathbf{PBanLat_1}$ has perfect Banach lattices as objects and positive order-continuous linear functions of norm at most one as morphisms.*

Although the definition of perfect spaces is simple, it is difficult to manipulate in practice. The following theorems provide some alternative characterisations, both in the normed and unnormed cases:

▶ **Theorem 28** ([21, Theorem 41.4, Volume XIII]). *Let V be a separated normed Riesz space. Then V is perfect and Banach iff V has the weak Fatou property.*

▶ **Theorem 29** ([1, Theorem 1.71]). *A Riesz space V is perfect iff*

- *it is separated;*
- *whenever $0 \leq \{x_\alpha\}_{\alpha:D}\uparrow$ and $\sup_{\alpha:D}\{f(x_\alpha)\}_{\alpha:D} < \infty$ for all positive $f \in V^\sigma$ and directed set D, there exists $x \in V$ such that $0 \leq \{x_\alpha\}_{\alpha:D} \uparrow x$.*

▶ **Corollary 30.** *Bands of perfect Riesz spaces are also perfect.*

▶ **Lemma 31.** *Every perfect Riesz space is Dedekind complete.*

Proof. The proof follows from the second condition of Theorem 29. ◀

▶ **Lemma 32.** *Every Riesz space of the form V^σ is perfect.*

Proof. To show the first point of Theorem 29, assume that $f_1 \neq f_2 \in V^\sigma$. Then there is $v \in V$ such that $f_1(v) \neq f_2(v)$. Using the fact that $\lambda f.f(v)$ is an element of $V^{\sigma\sigma}$, we can conclude that V^σ is separated. For the second point, let us assume that $0 \leq \{f_\alpha\}\uparrow$ and that for all $F \in V^{\sigma\sigma}$, if $F \geq 0$, then $\sup_\alpha F(f_\alpha) < \infty$. From this hypothesis, it follows that for all $v \in V$, if $v \geq 0$, then $\sup_\alpha f_\alpha(v) = \sup_\alpha \sigma(x)(f_\alpha) < \infty$. This means that the function $f(x) = \sup_\alpha f_\alpha(x)$ is well-defined, linear, and order-continuous. By Lemma 1.18 in Aliprantis and Burkinshaw [1], V^σ is Dedekind complete and f bounds f_α. ◀

An interesting fact that is not obvious from the definitions is that the bidual of Riesz spaces can be seen as a sort of completion procedure. We formalize this claim using adjunctions, but first we need a lemma.

▶ **Lemma 33** ([1, Theorem 1.70])**.** *Let V be an Archimedean Riesz space. The set $\sigma(V)$ is an order-dense Riesz subspace of $V^{\sigma\sigma}$.*

▶ **Theorem 34.** *The functor $(-)^{\sigma\sigma} : \mathbf{BanLat_1} \to \mathbf{PBanLat_1}$ is left adjoint to the forgetful functor U.*

Proof. We observe that if $f : V \multimap W$, then $\sigma^{-1} \circ f^{\sigma\sigma} : V^{\sigma\sigma} \multimap W$. In the other direction, if we have a function $f : V^{\sigma\sigma} \multimap W$, we can consider its restriction $f \upharpoonright V : V \multimap W$. To show that these operations are inverses, we use Theorem 11 and Lemma 33, which allow us to show that if two order-continuous functions agree on $\sigma(V)$, then they agree everywhere. ◀

Note that this implies that $\mathbf{PBanLat_1}$ is a reflective subcategory of $\mathbf{BanLat_1}$, which means that it is closed under the same (co)limits that exists in $\mathbf{BanLat_1}$, c.f. Borceux [3, Section 3.5].

2.6 Signed measures as Riesz spaces

Measures are usually defined as countably additive, nonnegative real-valued functions on a σ-algebra. *Signed measures* provide a slight generalization by dropping the requirement of nonnegativity.

▶ **Definition 35.** *Let (X, Σ) be a measurable space. A* signed measure *is a function $\mu : \Sigma \to \mathbb{R}$ such that $\mu(\emptyset) = 0$ and $\mu(\bigcup_{i\in\mathbb{N}} A_i) = \sum_{i\in\mathbb{N}} \mu(A_i)$ for disjoint sets $(A_i)_{i\in\mathbb{N}}$. The infinite series on the right hand side must converge absolutely.*

An important difference between ordinary measures and signed measures is that signed measures come equipped with a natural vector space structure. Indeed, it can be shown that signed measures are perfect Riesz spaces.

▶ **Lemma 36.** *Let (X, Σ) be a measurable space. The space $\mathcal{M}(X, \Sigma)$ of signed measures is a normed Riesz space.*

Proof. The vector space structure is defined pointwise with lattice structure defined by $\mu \vee \nu = (\mu - \nu)^+ + \nu$ using the Hahn-Jordan decomposition and the norm is the total-variation norm. ◀

When a measure μ is positive, its total variation norm is its total mass $\mu(X)$.

▶ **Theorem 37.** *Let (X, Σ) be a measurable space. The space $\mathcal{M}(X, \Sigma)$ of signed measures with the total variation norm is a perfect Banach lattice.*

Proof. The proof follows by applying Theorem 28, the lemma above and observing that since the order of measures is given pointwise, you can define their suprema pointwise as well. ◀

3 Models of linear logic

The categorical semantics of linear logic is very well understood; see Mellies [22] for an overview. In this section, we show that $\mathbf{PBanLat_1}$ is a model of classical linear logic.

3.1 Symmetric Monoidal Closed Structure

In order for $\mathbf{PBanLat_1}$ to interpret the multiplicative fragment of linear logic, i.e. give semantics to a linear λ-calculus with tensors, it must be a symmetric monoidal closed category. Concretely, it needs a *monoidal product* $\otimes$ such that for every object A, the functor $A \otimes -$ has a right adjoint $A \multimap -$, known as *linear implication*.

For models based on vector spaces, the monoidal product is typically given by the *tensor product*. For such models, linear implication has a natural interpretation in terms of linear functions. Furthermore, since our spaces are perfect, we have an involutive *linear negation* $A^\perp$ defined as the space $A \multimap \mathbb{R}$, and, in models of classical linear logic, the equation $A \otimes B = (A \multimap B^\perp)^\perp$ holds. Thus the tensor product $\otimes$ can be defined in terms of linear implication $\multimap$ and negation $^\perp$ in such models.

Note that this circumvents one of the main complications with the model of Ehrhard [11], where the existence of a suitable monoidal product is established non-constructively using a categorical density argument.

3.1.1 Internal Homs

Since the category $\mathbf{PBanLat_1}$ has order-continuous linear functions with norm at most 1 as morphisms, it makes sense to define the internal hom object $V \multimap W$ as the space of order-continuous linear functions between perfect Banach lattices V and W. This definition is justified by the following theorem.

▶ **Lemma 38** (c.f. Section B). *If V and W are perfect Riesz spaces, then the set of order continuous linear functions $V \multimap W$ is a perfect Riesz space.*

From Theorem 25 and the theorem above, it follows that if V and W are perfect Banach lattices, then so is $V \multimap W$. By using standard techniques from the literature on vector models of linear logic, we have

▶ **Theorem 39.** *The operation $\multimap : \mathbf{PBanLat_1}^{op} \times \mathbf{PBanLat_1} \to \mathbf{PBanLat_1}$ is functorial.*

3.1.2 Monoidal structure

As mentioned above, the monoidal structure on vector space models of linear logic is usually defined as a tensor product, and monoidal closure is obtained from the universal property of tensor products. The usual recipe for defining tensor products is to use a free construction modulo the tensor product equations. When working with infinite-dimensional spaces, a completion procedure may be required as well.

Indeed, this is the approach taken by Fremlin [15], in which a tensor product is defined for perfect Riesz spaces via a more traditional construction using the completion of the algebraic tensor product. It is also shown by Fremlin [15] that $V \otimes W \cong (V \multimap W^\perp)^\perp$, meaning that their construction is isomorphic to ours.

In contrast, our construction starts with the definition $V \otimes W \triangleq (V \multimap W^\sigma)^\sigma$, as required by the laws of linear logic. We then show that it satisfies the expected universal property of tensor products: for every biliear function $f : V \times W \to Y$, there is a unique linear function $\widehat{f} : V \otimes W \to Y$ such that $\widehat{f} \circ \iota = f$, where $\iota : V \times W \to Y$ is the bilinear inclusion function.

We show this using the fact that the internal hom can be used to classify bilinear functions using $V \multimap (W \multimap Y)$, then showing that this space is isomorphic to $V \otimes W \multimap Y$.

▶ **Lemma 40.** $V \otimes W \multimap Y \cong V \multimap W \multimap Y$.

Proof. If V and W are perfect Riesz spaces, then $V \multimap W \cong W^\sigma \multimap V^\sigma$. Then

$$\begin{aligned} V \otimes W \multimap Y &= (V \multimap W^\sigma)^\sigma \multimap Y \\ &\cong Y^\sigma \multimap (V \multimap W^\sigma) \cong V \multimap Y^\sigma \multimap W^\sigma \\ &\cong V \multimap W \multimap Y. \end{aligned}$$

◀

▶ **Theorem 41.** *$V \otimes W$, defined as $(V \multimap W^\sigma)^\sigma$, satisfies the universal property of tensor products.*

Proof. Observe that the set of (norm bounded) bilinear order-continuous functions $V \times W \to Y$ is (isometrically, in the normed case) isomorphic to $V \multimap W \multimap Y$. We must now show $V \otimes W \multimap Y \cong V \multimap W \multimap Y$. This is exactly Lemma 40. ◀

Using the universal property of tensor products and the (easy to prove) facts that $V \otimes (W \otimes Y) \cong (V \otimes W) \otimes Y$ and $V \otimes W \cong W \otimes V$, we can conclude:

▶ **Theorem 42.** $\mathbf{PBanLat_1}$ *is a symmetric monoidal closed category.*

It is difficult in general to give an intuitive characterization of the elements of a tensor product. This is also the case with our construction. Nevertheless, in the context of measures, we can give some intuition for the elements of $\mathcal{M}(A) \otimes \mathcal{M}(B)$. Let μ_A and μ_B be probability distributions on measurable spaces A and B, respectively. The product distribution $\mu_A \otimes \mu_B$ is the joint probability distribution on $A \times B$ with marginals μ_A and μ_B obtained by sampling μ_A and μ_B independently. This is an element of $\mathcal{M}(A) \otimes \mathcal{M}(B)$, but there are also other joint distributions in $\mathcal{M}(A) \otimes \mathcal{M}(B)$ that do not represent independent samples. For example, let $A = B = \{0, 1\}$ and consider the joint distribution $\frac{1}{2}(\delta_0 \otimes \delta_0 + \delta_1 \otimes \delta_1)$. Sampling this distribution returns $(0, 0)$ or $(1, 1)$, each with probability $1/2$, so the two components are clearly not independent.

In general, not every joint distribution is an element of the tensor product, as explained by Dahlqvist and Kozen [7]. From a programming point of view, the universal property of tensor products says that the behavior of a program taking inputs of type $\mathcal{M}(A) \otimes \mathcal{M}(B)$ is fully characterized by independent distributions over A and B.

3.2 $*$-autonomous categories

Classical linear logic differs from its intuitionistic variant by requiring that linear negation be involutive, that is, $A^{\perp\perp} = A$ for every formula A. Categorically, this is modeled by *$*$-autonomous categories*, symmetric monoidal closed categories $\mathbf{C}$ with a functor $(-)^* : \mathbf{C}^{\mathrm{op}} \to \mathbf{C}$ such that every object A is naturally isomorphic to A^{**} and for every three objects A, B, C, there is a natural bijection $\mathrm{Hom}(A \otimes B, C^*) \cong \mathrm{Hom}(A, (B \otimes C)^*)$. Equivalently, a $*$-autonomous category is a symmetric monoidal closed category $\mathbf{C}$ equipped with a *dualizing object* $\perp$ such that for every object A, the unit $\partial_A : A \to (A \multimap \perp) \multimap \perp$ is an isomorphism.

In our case, the dualizing object is $\mathbb{R}$, the unit is the linear function $\sigma_V : V \to V^{\sigma\sigma}$, and the isomorphism holds by assumption.

▶ **Theorem 43.** $\mathbf{PBanLat_1}$ *is a $*$-autonomous category.*

3.3 Cartesian and co-Cartesian structure

Cartesian and co-Cartesian structure are useful in the formation of product and sum types. In models of linear logic, these are represented by linear conjunction & and disjunction $\oplus$, respectively. In $\mathbf{PBanLat_1}$, both operations have $V \times W$ as their underlying set with lattice operations defined componentwise. In the normed case, we can distinguish them by choosing different norms.

▶ **Definition 44.** *Let V and W be normed Riesz spaces. We define*

- *the product $V \,\&\, W = (V \times W, \|-\|_{\text{sum}})$, where $\|(v,w)\|_{\text{sum}} = \|v\| + \|w\|$.*
- *the coproduct $V \oplus W = (V \times W, \|-\|_{\max})$, where $\|(v,w)\|_{\max} = \max(\|v\|, \|w\|)$.*

Since convergence for both is defined componentwise, by using Theorem 28 we can show that if V and W are perfect and Banach, then $V \,\&\, W$ and $V \oplus W$ are as well. The unit $\top$ for the product and 0 for the coproduct are both the trivial Riesz space $\{0\}$.

▶ **Theorem 45.** $\mathbf{PBanLat_1}$ *is (co-)Cartesian.*

4 Probabilistic coherence spaces and Banach lattices

Probabilistic coherence spaces (PCS) [8] are a model of linear logic with a vector space flavor. It has been shown by Ehrhard [11] that its intuitionistic fragment can be fully and faithfully embedded in a category of positive cones. In this section, we show that Banach lattices, contrary to previous work [11], extends the $*$-autonomous structure of the category of probabilistic coherence spaces as well as its symmetric monoidal closed structure. We make use of the vector space construction presented in the original paper [8].

▶ **Definition 46.** *A Probabilistic Coherence Space (PCS) is a pair $(|X|, \mathcal{P}(X))$, where $|X|$ is a countable set and $\mathcal{P}(X) \subseteq |X| \to \mathbb{R}^+$ called the* web *such that:*

- $\forall a \in |X|\ \exists \varepsilon_a > 0\ \varepsilon_a \cdot \delta_a \in \mathcal{P}(X)$, *where* $\delta_a(a') = 1$ *iff* $a = a'$ *and* 0 *otherwise;*
- $\forall a \in |X|\ \exists \lambda_a\ \forall x \in \mathcal{P}(X)\ x_a \leq \lambda_a$*;*
- $\mathcal{P}(X)^{\perp\perp} = \mathcal{P}(X)$, *where* $\mathcal{P}(X)^{\perp} = \{x \in |X| \to \mathbb{R}^+ \mid \forall v \in \mathcal{P}(X)\ \sum_{a \in X} x_a v_a \leq 1\}$.

▶ **Definition 47.** *Let $(|X|, \mathcal{P}(X))$ be a PCS. Its linear negation is the PCS $(|X|, \mathcal{P}(X)^{\perp})$.*

▶ **Definition 48.** *Let $(|X|, \mathcal{P}(X))$ and $(|Y|, \mathcal{P}(Y))$ be PCSs. The PCS $X \multimap Y$ is the pair $(|X| \times |Y|, \mathcal{P}(X \multimap Y))$, where $P(X \multimap Y)) = \{M : |X| \times |Y| \to \mathbb{R}^+ \mid \forall v \in \mathcal{P}(X)\ M \cdot v \in \mathcal{P}(Y)\}$, where $(M \cdot v)(y) = \sum_{x:X} M(x,y)v(x)$.*

The intuition behind Definition 46 is that the web of every PCS corresponds to the positive unit ball of a partially-ordered vector space. This idea is used by Ehrhard and Danos [8] to define a functor that maps every PCS to a Banach space. It is possible to show that this vector space can be equipped with a Riesz space structure, where the order is defined pointwise.

▶ **Definition 49.** *Given a PCS $(|X|, \mathcal{P}X)$, we define $BX = \{u \in \mathbb{R}^{|X|} \mid |u| \in \mathcal{P}X\}$ and $eX = \bigcup_{\lambda > 0} \lambda BX$. The pair $(eX, u \mapsto \sup_{u' \in \mathcal{P}X^{\perp}} \langle |u|, u' \rangle)$ is the normed Riesz space associated with the PCS $(|X|, \mathcal{P}X)$.*

It is shown by Ehrhard and Danos [8] that eX is a Banach space. Furthermore, the lattice structure can be defined pointwise, making eX a Banach lattice. Later in this section we will show that e can be made into a functor.

PCoh and duality

In this section we show that the functor e preserves the $*$-autonomous structure of **PCoh**.

▶ **Theorem 50** (c.f. Section C). *For every probabilistic coherence space X, there is a natural isomorphism $e(X^{\perp}) \cong e(X)^{\sigma}$.*

▶ **Corollary 51.** *For every PCS $(|X|, \mathcal{P}(X))$ the vector space eX is a perfect Banach lattice.*

Since convergence for PCS is defined componentwise, it is possible to use a similar proof technique to show

▶ **Theorem 52.** *The operation e is monoidal closed and functorial.*

Proof. The functoriality of e has been proven in Section 5.1 of Ehrhard and Danos [8]. The proof of preservation of monoidal closure is similar to the proof of Theorem 50. ◀

Another important theorem which is direct to show is.

▶ **Theorem 53.** *The functor $e : \mathbf{PCoh} \to \mathbf{PBanLat_1}$ is full and faithful.*

5 Categories of Cones and $\mathbf{PBanLat_1}$

Even though $\mathbf{PBanLat_1}$ is a mathematically natural model of linear logic, it relies on tools from functional analysis not usually familiar to computer scientists. On the other hand, in recent years, cones have found numerous applications in semantics of programming languages and logics [12, 6, 23, 18]. In this section we show that $\mathbf{PBanLat_1}$ is isomorphic to a category cones, meaning that computer scientists can translate their intuitions about cones to this novel setting without having to learn functional analysis.

As it was frequently mentioned throughout this paper, every Banach lattice gives rise to a positive cone. Furthermore, since every $\mathbf{PBanLat_1}$ morphism $f : V \to W$ is positive and has norm at most 1, it restricts to a linear function $\mathcal{B}(V)^+ \to \mathcal{B}(W)^+$. With this observations we state a few definitions from previous work [6, 11], which assume that the cones are separated.

▶ **Definition 54** (cf. [12, Definition 4.1]). *A cone C is a $\mathbb{R}^+$-semimodule with a norm $\|\cdot\| : C \to \mathbb{R}^+$ such that it satisfies the cancellation property $x + y_1 = x + y_2$ implies $y_1 = y_2$, for every points x, y_1 and y_2.*

Every cone can be equipped with the partial order $x \leq y$ if and only if there is a z such that $x + z = y$, meaning that it is possible to define a partial subtraction operation whenever $x \leq y$, calling $y - x$ the element such that $x + (y - x) = y$.

A function $f : C_1 \to C_2$ between cones is linear if it commutes with addition and scalar multiplication, it is monotonic if it preserves the order relation, and it is Scott-continuous if for every directed set x_α with supremum x, $\sup_\alpha f(x_\alpha) = f(x)$. As is the case with partially-ordered vector spaces, there are different classes of cones where the order and the norm have particular properties:

▶ **Definition 55.** *A cone C is said to be:*

- *Sequentially complete if every norm-bounded sequence has a least upper bound.*
- *Directed complete if every norm-bounded directed set has a least upper bound.*
- *A lattice cone if the poset structure is a lattice.*

Using this notation, it seems appropriate to imagine that there should be a functor $\mathbf{PBanLat_1} \to \mathbf{CLat}$, where $\mathbf{CLat}$ is the category of directed complete cone lattices. It is unclear, however, if there is a mapping on morphisms. Luckily, the lemma below guarantees that the mapping is well-defined. Its proof follows from the weak Fatou property.

▶ **Lemma 56.** *Let V and W be two perfect Banach lattices and $f : V \to W$ a linear, positive function of norm at most 1. The function f is order-continuous if and only if $\sup_{x \in A} f(x) = f(v)$ whenever $A \subseteq V^+$ is a non-empty upwards-directed set with supremum v.*

Since the mapping on morphisms is basically the identity, the functorial laws hold, which allows us to conclude that there is a functor $\mathbf{PBanLat_1} \to \mathbf{CLat}$.

Next, we would like to map every positive cone to a vector space. Let C be a positive cone and define $C - C = \{(c_1, c_2) \,|\, c_1, c_2 \in C\}/\sim$, where $\sim$ is the binary relation $(c_1, c_2) \sim (c_3, c_4)$ iff $c_1 + c_4 = c_2 + c_3$. Intuitively, $C - C$ corresponds to the vector space of formal differences $c_1 - c_2$ of elements in C. The equivalence relation is used to capture the fact that, for instance, $(3, 2)$ and $(4, 3)$ should represent the same real number, since $3 - 2 = 1 = 4 - 3$.

▶ **Theorem 57** (c.f. Section D). *Let C be a directed complete cone lattice. Then $C - C$ is a perfect Banach lattice.*

By linearity, Scott-continuous functions $f : C \to D$ with norm at most 1 extend to order-continuous functions $f : (C - C) \to (D - D)$ with norm at most 1 and we can prove that there is a functor $\mathbf{CLat} \to \mathbf{PBanLat_1}$. With this functor and the positive cone restriction functor defined, it is a direct calculation to show:

▶ **Theorem 58.** *The categories $\mathbf{PBanLat_1}$ and $\mathbf{CLat}$ are isomorphic.*

Variables	x, y, z		
Reals	r	$\in$	$\mathbb{R}$
MK Expressions	M	$::=$	$x \mid r \mid \mathsf{uniform} \mid (M_1, M_2) \mid \pi_1\, M \mid \pi_2\, M$
			$\mid \mathsf{let}\ x = M\ \mathsf{in}\ N$
LL Expressions	t, u	$::=$	$x \mid \lambda x.t \mid t\ u \mid t \otimes u \mid \mathsf{let}\ x \otimes y = t\ \mathsf{in}\ u$
			$\mid \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M$
Types MK	τ	$::=$	$\mathbb{R} \mid \tau \times \tau$
Types LL	$\underline{\tau}$	$::=$	$1 \mid \mathcal{M}\tau \mid \underline{\tau} \multimap \underline{\tau} \mid \underline{\tau} \otimes \underline{\tau}$
Linear Contexts	Γ	$::=$	$x_1 : \underline{\tau}_1, \ldots, x_n : \underline{\tau}_n$
MK Contexts	Γ	$::=$	$x_1 : \tau_1, \ldots, x_n : \tau_n$

Figure 1 Terms and Types of λ_{MK}^{LL}.

6 A Probabilistic Calculus

Though it is theoretically interesting understanding how $\mathbf{PBanLat_1}$ relates to existing models of linear logic, we are also interested in using it as a semantic basis for a language with probabilistic primitives. Being symmetric monoidal closed, it can give semantics to the linear λ-calculus. This, however, is insufficient from a programming point of view. The linearity restrictions are severely limiting in terms of which programs one can define in this language. A frequently used solution to this lack of expressivity is to use the exponential modality, where the coKleisli category is Cartesian closed, meaning that it can interpret the λ-calculus.

However, even though we have not defined a linear logic exponential in $\mathbf{PBanLat_1}$, we can still get non-linear programming by using recent work [2] that proposes a new syntax for programming with linear operators and Markov kernels. The proposed two-level calculus allows for non-linear programs to be defined by using a lax-monoidal modality.

The λ_{MK}^{LL} metalanguage

The semantic structure used to interpret the calculus of Azevedo de Amorim [2] is given by a triple $(\mathbf{C}, \mathbf{L}, \mathcal{M})$, where $\mathbf{C}$ is roughly a category of Markov kernels[1], $\mathbf{L}$ is a symmetric monoidal closed category and $\mathcal{M} : \mathbf{C} \to \mathbf{L}$ is a lax monoidal functor.

This two-level structure manisfests itself at the syntactic level by having a two-level syntax: the first level is used to program kernels while the second one serves as a kind of metalanguage that has access to higher-order functions, both of which are depicted in Figure 1, The linear language has linear function types, which allows for higher-order programming and, unlike most languages based on linear logic, it has a modality $\mathcal{M}$, which corresponds to the types that may be sampled from. The variables bound by the linear context are, roughly speaking, computations. In the language for kernels there are no linearity restrictions and, therefore, variables, i.e. samples from distributions, can be freely duplicated and discarded. Under this perspective, the variables in MK programs should be thought of as values. The intuition behind this language is that linearity forbids distributions to be sampled more than once, but once you have the sample in hands, it can be used as many times as you want.

Each layer has its own typing judgement relations $\vdash_{LL}$ and $\vdash_{MK}$, which we go over in more detail in Section A. We highlight one of the most interesting rules; it is the rule that allows programs to be transported between layers:

$$\textsc{Sample}\quad \frac{x_1 : \tau_1 \cdots x_n : \tau_n \vdash_{MK} M : \tau \quad \Delta; \Gamma_i \vdash_{LL} t_i : \mathcal{M}\tau_i \qquad 0 < i \leq n}{\Delta; \Gamma_1, \cdots, \Gamma_n \vdash_{LL} \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M : \mathcal{M}\tau}$$

Operationally, it samples from n LL programs $\{t_i\}_i$, each sample is bound to the corresponding variable in $\{x_i\}_i$ and finally the continuation M is executed.

We want to model λ_{MK}^{LL} with $\mathbf{PBanLat_1}$. For that we still need a CD category and a lax monoidal functor. For the CD category we will use the category of measurable spaces and sub-Markov kernels.

▶ **Definition 59.** *The category* **sStoch** *has measurable spaces as objects and sub-Markov kernels as morphisms, i.e. measurable functions between a measurable space and the space of subprobability distributions over a measurable space.*

sStoch is a CD category, which means that it is symmetric monoidal, with the monoidal product being the product measurable space.

▶ **Theorem 60** (c.f. Section E)**.** *There is a lax monoidal functor* $\mathcal{M} : \mathbf{sStoch} \to \mathbf{PBanLat_1}$.

This means that the triple $(\mathbf{sStoch}, \mathbf{PBanLat_1}, \mathcal{M})$ is a λ_{MK}^{LL} model.

7 Related work

There have been a number of semantics of linear logic based on vector space-like objects. Two important families of such semantics are the ones based on probabilistic coherence spaces and the ones based on Banach spaces. As we will explain below, we see our model as a nice synthesis of these two approaches.

[1] a CD category, to be more precise

Positive Cone Semantics of Linear Logic

To overcome the limitation that **PCoh** cannot represent continuous distributions, Ehrhard et al. define a cartesian closed category $\mathbf{CStab_m}$ [12], which uses normed $\mathbb{R}^+$-semimodule – which are in correspondence with positive cones of partially ordered vector spaces – to interpret a probabilistic variant of PCF with continuous distributions. In a follow-up paper, Ehrhard [11] has defined a category $\mathbf{CLin_m}$ of sequentially complete positive cones with measurability paths and linear Scott continuous maps in which **PCoh** embeds fully and faithfully.

A similar approach was taken by Slavnov [24], who defined a category **CCones** of so-called coherent cones and linear contractive functions and showed that it is a model of classical linear logic. These cones come equipped with a different notion of completeness that is stronger than sequential completeness but weaker than ours.

From a mathematical point of view, the objects of both **CCones** and $\mathbf{CStab_m}$ are not as well understood as Banach lattices, making them not ideal semantic frameworks to reason about probabilistic programs, since many useful lemmas for reasoning about programs would have to be reproved. Besides, our model provides a clear mathematical justification for having Fatou-like properties in the semantics: it is forced upon it by Theorem 28 instead of being there for denotational reasons, as is the case of $\mathbf{CStab_m}$, or in enabling the exponential construction, as is the case of **CCones**, showing a kind of canonicity of our model.

Vector Space Semantics of Linear Logic

Dahlqvist and Kozen [7] have defined a category of partially ordered Banach spaces **RoBan**, shown that it is a model of intuitionistic linear logic, and used it to interpret a higher-order imperative probabilistic language with while loops and soft-conditioning.

Their model also uses a mathematically well-understood class of vector spaces. That being said, by using a more general class of vector spaces than we do, their model has less structure than ours. A practical consequence of this lack of structure is that in order to guarantee the soundness of their semantics, they define 6 type grammars that are used for different program constructs. As an example, in order to interpret conditionals and while loops the context may only have Dedekind complete types.

Another relevant vector space model is the one based on complex coherent Banach spaces [17]. However, since they are complex vector spaces, it is unclear if it would be possible to embed **PCoh** into them.

Neither **RoBan** nor $\mathbf{CStab_m}$ are models of classical linear logic.

8 Conclusion

In this paper we have shown that $\mathbf{PBanLat_1}$ is a model of classical linear logic that conservatively extends **PCoh** and can be used to give semantics to a recursive probabilistic calculus. Our model differs from existing extensions of **PCoh** that only extends **PCoh**'s intuitionistic fragment, meaning that they do not have an involutive negation. We believe that our model is a good fit for formal verification purposes because Riesz spaces have decades of research and have been extensively used in the formalization of stochastic processes.

For future work, we are interested in showing that $\mathbf{PBanLat_1}$ can accommodate exponentials and use this category for reasoning about correctness properties of probabilistic programs such as inference algorithms.

References

1 Charalambos D Aliprantis and Owen Burkinshaw. *Positive operators*. Springer, 2006. doi:10.1007/978-1-4020-5008-4.

2 Pedro H. Azevedo de Amorim. A higher-order language for markov kernels and linear operators. In *Foundations of Software Science and Computation Structures (FoSSaCS)*, 2023. doi:10.1007/978-3-031-30829-1_5.

3 Francis Borceux. *Handbook of categorical algebra: volume 1, Basic category theory*, volume 1. Cambridge University Press, 1994.

4 Francis Borceux. *Handbook of categorical algebra: volume 2, Categories and Structures*, volume 2. Cambridge University Press, 1994.

5 Kenta Cho and Bart Jacobs. Disintegration and bayesian inversion via string diagrams. *Mathematical Structures in Computer Science*, 2019.

6 Raphaëlle Crubillé. Probabilistic stable functions on discrete cones are power series. In *Logic in Computer Science (LICS)*, 2018.

7 Fredrik Dahlqvist and Dexter Kozen. Semantics of higher-order probabilistic programs with conditioning. In *Principles of Programming Languages (POPL)*, 2019.

8 Vincent Danos and Thomas Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Information and Computation*, 209(6):966–991, 2011. doi:10.1016/J.IC.2011.02.001.

9 Thomas Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12(5):579–623, 2002. doi:10.1017/S0960129502003729.

10 Thomas Ehrhard. Differentials and distances in probabilistic coherence spaces. *arXiv preprint*, 2019. arXiv:1902.04836.

11 Thomas Ehrhard. On the linear structure of cones. In *Logic in Computer Science (LICS)*, 2020.

12 Thomas Ehrhard, Michele Pagani, and Christine Tasson. Measurable cones and stable, measurable functions: a model for probabilistic higher-order programming. In *Principles of Programming Languages (POPL)*, 2017.

13 Thomas Ehrhard, Christine Tasson, and Michele Pagani. Probabilistic coherence spaces are fully abstract for probabilistic PCF. In *Principles of Programming Languages (POPL)*, 2014.

14 David H Fremlin. *Measure theory*. Torres Fremlin, 2000.

15 DH Fremlin. Abstract Köthe spaces IV. In *Mathematical Proceedings of the Cambridge Philosophical Society*, pages 45–52. Cambridge University Press, 1968.

16 Tobias Fritz. A synthetic approach to markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370:107239, 2020.

17 Jean-Yves Girard. Coherent banach spaces: a continuous denotational semantics. *Theoretical Computer Science*, 227(1-2):275–297, 1999. doi:10.1016/S0304-3975(99)00056-0.

18 Klaus Keimel and Gordon D Plotkin. Mixed powerdomains for probability and nondeterminism. *Logical Methods in Computer Science*, 2017.

19 Marie Kerjean and Christine Tasson. Mackey-complete spaces and power series–a topological model of differential linear logic. *Mathematical Structures in Computer Science*, 28(4):472–507, 2018. doi:10.1017/S0960129516000281.

20 Dexter Kozen. Semantics of probabilistic programs. In *Symposium on Foundations of Computer Science (SFCS)*, 1979.

21 WAJ Luxemberg and AC Zaanen. Notes on Banach function spaces VI-XIII. *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen, Series A*, 66:251–263, 1963.

22 Paul-André Mellies. Categorical semantics of linear logic. *Panoramas et syntheses*, 27:15–215, 2009.

23 Peter Selinger. Towards a semantics for higher-order quantum computation. In *Quantum Programming Languages (QPL)*, 2004.

24 Sergey Slavnov. Linear logic in normed cones: probabilistic coherence spaces and beyond. *Mathematical Structures in Computer Science*, 31(5):495–534, 2021. doi:10.1017/S0960129521000177.
25 Christine Tasson and Thomas Ehrhard. Probabilistic call by push value. *Logical Methods in Computer Science*, 15, 2019. doi:10.23638/LMCS-15(1:3)2019.
26 Adriaan C Zaanen. *Introduction to operator theory in Riesz spaces*. Springer, 2012.

A A Metalanguage for Linear Operators and Markov Kernels

In this section we further explain the two-level language λ_{MK}^{LL} and its semantics. The language MK corresponds to an effectful language with probabilistic primitives and where free variables are assumed to be values, as opposed to computations. For instance, the program $x : \mathbb{N}, y : \mathbb{N} \vdash_{MK} x + y : \mathbb{N}$ is interpreted as a deterministic program. This language is interpreted in a CD category, which can be seen as an abstraction for programming with commutative effects [16].

▶ **Definition 61** ([5, Definition 2.2]). *CD categories are symmetric monoidal categories such that every object A has a commutative comonoid structure* $\mathsf{copy}_A : A \to A \otimes A$ *and* $\mathsf{delete}_A : A \to 1$ *satisfying certain structural properties.*

In the context of probabilistic programming, there are many CD categories to choose from. In particular, for any subprobability monad, its Kleisli category is a CD category. This is the case for the **sStoch** category, since it can be characterized as the category of measurable sets and measurable functions $A \to \mathcal{G}(B)$, where $\mathcal{G}$ is the subprobability monad over **Meas**.

The language LL is basically a linear λ-calculus. By itself, linearity limits the expressivity of the language quite a bit. In the original paper, the author argues that for probabilistic programming, the linear usage of variables is, semantically, too restrictive, since many linear probabilistic calculi, in the algebraic sense, may use variables more than once [2]. This observation led to the introduction of the $\mathcal{M}$ modality in the LL language which allows MK programs to be called from an LL program. Semantically, this is interpreted as a lax monoidal functor.

▶ **Definition 62** ([4, Definition 6.4.1]). *Let* $\mathbf{C}$ *and* $\mathbf{D}$ *be monoidal categories. A (lax) monoidal functor is a functor* $F : \mathbf{C} \to \mathbf{D}$ *equipped with a natural transformation* $\varepsilon_{A,B} : FA \otimes_{\mathbf{D}} FB \to F(A \otimes_{\mathbf{C}} B)$ *and a morphism* $I_{\mathbf{D}} \to F(I_{\mathbf{C}})$ *making certain coherence diagrams commute.*

From a programming point of view, types $\mathcal{M}\tau$ should be thought of as types that can be sampled from. Supposing that the language has a primitive $\mathsf{uniform}$ for the uniform distribution over the unit interval the Sample construct can be used to write the program

$$\mathsf{sample}\ \mathsf{uniform}\ \mathsf{as}\ x\ \mathsf{in}\ (x + x)$$

The program above samples from a uniform distribution and adds the result to itself. This program illustrates why this syntax increases the expressivity of the linear λ-calculus. By allowing the continuation $x + x$ to be an MK program, variables may be freely reused or discarded without worrying about syntactic restriction imposed by linearity.

However, once inside the MK language, there is no way of going back to the higher-order language, meaning that the program $\mathsf{sample}\ \mathsf{uniform}\ \mathsf{as}\ x\ \mathsf{in}\ (\mathsf{sample}\ \mathsf{uniform}\ \mathsf{as}\ y\ \mathsf{in}\ (x+y))$ is not well-typed. This is mitigated by lax monoidality, which makes it possible to simultaneously sample from distributions: $\mathsf{sample}\ (\mathsf{uniform}, \mathsf{uniform})\ \mathsf{as}\ (x, y)\ \mathsf{in}\ (x + y)$.

$$\textsc{Var}\ \frac{}{\Gamma, x : \tau \vdash_{MK} x : \tau} \qquad \textsc{Const}\ \frac{r \in \mathbb{R}}{\Gamma \vdash_{MK} r : \mathbb{R}} \qquad \textsc{Uniform}\ \frac{}{\Gamma \vdash_{MK} \mathsf{uniform} : \mathbb{R}}$$

$$\textsc{Let}\ \frac{\Gamma \vdash_{MK} t : \tau_1 \qquad \Gamma, x : \tau_1 \vdash_{MK} u : \tau}{\Gamma \vdash_{MK} \mathsf{let}\ x = t\ \mathsf{in}\ u : \tau} \qquad \textsc{Pair}\ \frac{\Gamma \vdash_{MK} t : \tau_1 \qquad \Gamma \vdash_{MK} u : \tau_2}{\Gamma \vdash_{MK} (t, u) : \tau_1 \times \tau_2}$$

$$\textsc{Proj1}\ \frac{\Gamma \vdash_{MK} t : \tau_1 \times \tau_2}{\Gamma \vdash_{MK} \pi_1 t : \tau_1} \qquad \textsc{Proj2}\ \frac{\Gamma \vdash_{MK} t : \tau_1 \times \tau_2}{\Gamma \vdash_{MK} \pi_2 t : \tau_2} \qquad \textsc{Axiom}\ \frac{}{x : \tau \vdash_{LL} x : \tau}$$

$$\textsc{Unit}\ \frac{}{\cdot \vdash_{LL} \mathsf{unit} : 1} \qquad \textsc{Abstraction}\ \frac{\Gamma, x : \tau_1 \vdash_{LL} t : \tau_2}{\Gamma \vdash_{LL} \lambda x.t : \tau_1 \multimap \tau_2}$$

$$\textsc{Application}\ \frac{\Gamma_1 \vdash_{LL} t : \tau_1 \multimap \tau_2 \qquad \Gamma_2 \vdash_{LL} u : \tau_1}{\Gamma_1, \Gamma_2 \vdash_{LL} t\ u : \tau_2} \qquad \textsc{Tensor}\ \frac{\Gamma_1 \vdash_{LL} t : \tau_1 \qquad \Gamma_2 \vdash_{LL} u : \tau_2}{\Gamma_1, \Gamma_2 \vdash_{LL} t \otimes u : \tau_1 \otimes \tau_2}$$

$$\textsc{LetTensor}\ \frac{\Gamma_1 \vdash_{LL} t : \tau_1 \otimes \tau_2 \qquad \Gamma_2, x : \tau_1, y : \tau_2 \vdash_{LL} u : \tau}{\Gamma_1, \Gamma_2 \vdash_{LL} \mathsf{let}\ x \otimes y = t\ \mathsf{in}\ u : \tau}$$

$$\textsc{Sample}\ \frac{x_1 : \tau_1 \cdots x_n : \tau_n \vdash_{MK} M : \tau \quad \Delta; \Gamma_i \vdash_{LL} t_i : \mathcal{M}\tau_i \qquad 0 \leq i < n}{\Delta; \Gamma_1, \cdots, \Gamma_n \vdash_{LL} \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M : \mathcal{M}\tau}$$

Figure 2 Typing rules for λ^{LL}_{MK}.

▶ **Definition 63.** *A model of λ^{LL}_{MK} is a triple $(\mathbf{C}, \mathbf{L}, \mathcal{M})$, where $\mathbf{C}$, a symmetric monoidal closed category $\mathbf{L}$ and $\mathcal{M} : \mathbf{M} \to \mathbf{C}$ is a lax monoidal functor.*

The typing rules are depicted in Figure 2. They are basically the amalgamation of the rules for programming with CD categories, i.e. a first-order expression language with pairs, with symmetric monoidal closed categories, i.e. the linear λ-calculus with tensor types. The main novelty is the introduction of the lax monoidal modality $\mathcal{M}$ and its accompanying typing rule Sample which connects the MK and LL languages.

Much like the typing rules, the categorical semantics of λ^{LL}_{MK} is the combination of the categorical semantics of the internal languages of CD categories and the linear λ-calculus with the exception of the Sample rule that makes use of the functor $\mathcal{M}$. The full semantics is depicted in Figure 3.

B Proof of Lemma 38

By Theorem 14, $V \multimap W$ is a Riesz space. Applying Theorem 29, we can also show that it is perfect. To show separability, let $f_1, f_2 : V \multimap W$ be distinct functions. Then there is a point $v \in V$ such that $f_1(v) \neq f_2(v)$. Since W is perfect, it is separated, therefore there exists

$$\textsc{Var}\ \frac{}{\tau \times \Gamma \xrightarrow{id_\tau \times del_\Gamma} \tau} \qquad \textsc{Let}\ \frac{\Gamma \xrightarrow{M} \tau_1 \qquad \Gamma \times \tau_1 \xrightarrow{N} \tau_2}{\Gamma \xrightarrow{copy;(id\times M);N} \tau_2} \qquad \times\ \textsc{Intro}\ \frac{\Gamma \xrightarrow{M} \tau_1 \qquad \Gamma \xrightarrow{N} \tau_2}{\Gamma \xrightarrow{copy;M\times N} \tau_1 \times \tau_2}$$

$$\times\ \textsc{Elim}_i\ \frac{\Gamma \xrightarrow{M} \tau_1 \times \tau_2}{\Gamma \xrightarrow{M;(id_{\tau_i}\times del)} \tau_i}$$

$$\textsc{Var}\ \frac{}{\underline{\tau} \xrightarrow{id_\tau} \underline{\tau}} \qquad \textsc{Abstraction}\ \frac{\Gamma \otimes \underline{\tau_1} \xrightarrow{t} \underline{\tau_2}}{\Gamma \xrightarrow{\mathsf{cur}(t)} \underline{\tau_1} \multimap \underline{\tau_2}} \qquad \textsc{Application}\ \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \multimap \underline{\tau_2} \qquad \Gamma_2 \xrightarrow{u} \underline{\tau_1}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{(t\otimes u);\mathsf{ev}} \underline{\tau_2}}$$

$$\otimes\ \textsc{Intro}\ \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \qquad \Gamma_2 \xrightarrow{u} \underline{\tau_2}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{t\otimes u} \underline{\tau_1} \otimes \underline{\tau_2}} \qquad \otimes\ \textsc{Elim}\ \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \otimes \underline{\tau_2} \qquad \Gamma_2 \otimes \underline{\tau_1} \otimes \underline{\tau_2} \xrightarrow{u} \underline{\tau}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{(id\otimes t);u} \underline{\tau}}$$

$$\textsc{Sample}\ \frac{\tau_1 \times \cdots \times \tau_n \xrightarrow{M} \tau \qquad \Gamma_i \xrightarrow{t_i} \mathcal{M}\tau_i}{\Gamma_1 \otimes \cdots \otimes \Gamma_n \xrightarrow{t_1 \otimes \cdots \otimes t_n} \mathcal{M}\tau_1 \otimes \cdots \otimes \mathcal{M}\tau_n \xrightarrow{\mu} \mathcal{M}(\tau_1 \times \cdots \times \tau_n) \xrightarrow{\mathcal{M}M} \mathcal{M}\tau}$$

Figure 3 Categorical Semantics of λ_{MK}^{LL}.

$g : W \multimap \mathbb{R}$ such that $g(f_1(v)) \neq g(f_2(v))$. Then the order-continuous function $\lambda f\,.\,g(f(v))$ separates the points f_1 and f_2, therefore $V \multimap W$ is separated.

Now let $0 \leq \{f_\alpha\}\uparrow$ be an increasing net such that $\sup_\alpha F(f_\alpha) < \infty$ for all positive $F : (V \multimap W) \multimap \mathbb{R}$. We can define an f such that $f_\alpha \uparrow f$ pointwise. Let $v \in V^+$ and let $F : W \multimap \mathbb{R}$ be a positive functional. Consider the functional $\lambda f\,.\,F(f(v)) : (V \multimap W) \multimap \mathbb{R}$. By hypothesis, $\sup_\alpha(F(f_\alpha(v))) < \infty$, and since W is perfect and $\{f_\alpha(v)\}$ is a positive net in W, there exists $f(v) \in W$ such that $f_\alpha(v) \uparrow f(v)$. This defines f on elements of V^+, and for arbitrary $v \in V$ we take $f(v) = f(v^+) - f(v^-)$. Then $\sup_\alpha f_\alpha = f$.

C Proof of Theorem 50

If $u \in e(X^\perp)$, consider the element $f_u = \lambda x\,.\,\langle u^+, x\rangle - \langle u^-, x\rangle$. It is possible to show that the function $\lambda x\,.\,\langle u, x\rangle$ is positive and Scott-continuous, therefore order-continuous for every $u \in \mathcal{P}(X)$. Using this result, it is not hard to show that $f_u \in e(X)^\sigma$.

Conversely, consider an element $f \in e(X)^\sigma$. Without loss of generality, we can assume that f is positive. We want to associate to f an element in $e(X^\perp)$. As is shown by Ehrhard and Danos [8], we can alternatively characterize the space $e(X)$ as

$$\{u \in \mathbb{R}^{|X|} \mid \exists \lambda > 0\ \forall u' \in \mathcal{P}(X^\perp)\ \langle |u|, u'\rangle \leq \lambda\}.$$

Consider the function $f_\delta = \lambda x\,.\,f(\delta_x)$. Let us show that $f_\delta \in e(X^\perp)$. To do this, we show that for every $u \in \mathcal{P}(X)$, $\langle |f'|, u\rangle$ is uniformly bounded. Let $(u_\alpha)_{\alpha\in\mathbb{P}_{\text{fin}}(X)}$ be the ascending net $u_{\alpha,a} = u_a$ if $a \in \alpha$ and 0 otherwise. By expanding the definition, we get the equality

$$\langle |f_\delta|, u_\alpha\rangle = \sum_{a\in|X|} |f(\delta_a)|u_{\alpha,a} =$$
$$\sum_{a\in|X|} |f(\delta_a u_{\alpha,a})| = \sum_{a\in|X|} f(\delta_a u_{\alpha,a}).$$

We get the last equality from f being a positive function. Since every u_α has finite support, the expression above is well defined.

$$\sum_{a\in|X|} f(\delta_a u_{\alpha,a}) = f\left(\sum_{a\in|X|} \delta_a u_{\alpha,a}\right) = f(u_\alpha)$$

Since f is order-continuous and monotone and $\{u_\alpha\}$ is an increasing net, we can conclude that $\langle |f_\delta|, u\rangle \leq f(u)$, therefore for every $u \in \mathcal{P}(X)$, $\langle |f_\delta|, u\rangle \leq \|f\|$ and $f_\delta \in e(X^\perp)$. If f is not positive, we decompose it as the difference of two positive maps $f = f^+ - f^-$ and define $f_\delta = f^+_\delta - f^-_\delta$.

A direct calculation shows that this is indeed an isomorphism.

D Proof of Theorem 57

Let C be a directed complete lattice cone. In order to define functions over it we use the universal property of quotients: it suffices to define it over every pair (c_1, c_2) while guaranteeing that the function acts the same over every equivalence class.

For instance, the vector space structure can be simply defined componentwise. Let $(c_1, c_2), (c_3, c_4) \in C - C$ then we define

$$(c_1, c_2) + (c_3, c_4) = (c_1 + c_3, c_2 + c_4)$$
$$\alpha(c_1, c_2) = (\alpha c_1, \alpha c_2) \text{ for } \alpha \geq 0$$
$$\alpha(c_1, c_2) = (-\alpha c_2, -\alpha c_1) \text{ otherwise}$$

The lattice operations require a bit more ingenuity, and we first observe the equation $u \vee v = u + (v - u)^+$ which holds in every Riesz space, reducing the lowest upper bound operation to addition and the positive part. By doing some algebraic manipulations we get that if $(c_1, c_2), (c_3, c_4) \in C - C$ then we define $(c_1, c_2) \vee (c_3, c_4) = (c_1, c_2) - ((c_3, c_4) - (c_1, c_2))^+ = (c_1, c_2) + (c_3 + c_2 - (c_1 + c_4) \wedge (c_2 + c_3), 0) = (c_1 + c_3 + c_2 - (c_1 + c_4) \wedge (c_2 + c_3), c_2)$. The lattice equations such as commutativity and idempotency follow by unfolding the definitions and from C being a lattice.

Before defining a norm over $C - C$ we first need the following lemma

▶ **Lemma 64.** $(C - C)^+ \cong \{(c, 0) \mid c \in C\} \cong C$.

Proof. The mapping $\{(c, 0) \mid c \in C\} \to (C - C)^+$ is the injection through the equivalence class function and the mapping in the other direction can be constructed by observing that whenever $(c_1, c_2) \geq (0, 0)$ it can be shown that $c_1 \geq c_2$ and, therefore, $(c_1 - c_2, 0) = (c_1, c_2)$ and this decomposition is unique, since $(c, 0) = (d, 0)$ implies, by definition of $\sim$ that $c = d$. The second isomorphism is trivial. ◀

Given a norm over C it is possible to extend it to a norm over $C - C$. This follows from the property of normed Riesz spaces, where $\|\|v\|\| = \|v\|$ which forces us to define $\|(c_1, c_2)\| = \|\|(c_1, c_2)\|\|_C$. Note that since $|(c_1, c_2)|$ is a positive element of $C - C$, by the lemma above it can be mapped back to an element of C which, in turn, has a norm.

Therefore, we have shown that $C - C$ is a normed Riesz space. Since C has the directed completeness property it follows that $C - C$ has the weak Fatou property and, therefore, it is Banach and perfect.

E Proof of Theorem 60

There is a standard functor $\mathcal{M}$ that maps measurable sets to the vector space of signed measures and sub-Markov kernels $f : A \to MB$ to the linear function $\mathcal{M}f(\mu) = \int f d\mu$. The proof of linearity is standard, but order-continuity requires a few words. Let $\{\mu_\alpha\} \downarrow 0$ be a descending arrow, $\mathcal{M}f(\mu_\alpha) = \int f d\mu_\alpha \leq \int 1 d\mu_\alpha = \mu_\alpha(A)$ which, as μ_α goes to zero, so does $\mu_\alpha(A)$, making $\tilde{f}$ order-continuous. The functorial laws also follows from standard proofs from the literature.

To show that $\mathcal{M}$ is lax monoidal, we need to define a natural transformation $\mu_{X,Y} : \mathcal{M}(X) \otimes \mathcal{M}(Y) \to \mathcal{M}(X \times Y)$ which is easily defined by the universal property of the tensor product and a morphism $\varepsilon : \mathbb{R} \multimap \mathcal{M}(1)$ which maps a real number r to the measure $r\delta_{\{*\}}$, where $*$ is the only member of the singleton set 1. Showing that the necessary diagrams commute follows from the universal property of the tensor product.

Insights from Univalent Foundations: A Case Study Using Double Categories

Nima Rasekh
Institute of Mathematics and Computer Science, University of Greifswald, Germany

Niels van der Weide
Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands

Benedikt Ahrens
Delft University of Technology, The Netherlands
University of Birmingham, UK

Paige Randall North
Department of Information and Computing Sciences and Mathematical Institute, Utrecht University, The Netherlands

Abstract

Category theory unifies mathematical concepts, aiding comparisons across structures by incorporating not just objects, but also morphisms capturing interactions between objects. Of particular importance in some applications are double categories, which are categories with two classes of morphisms, axiomatizing two different kinds of interactions between objects. These have found applications in many areas of mathematics and theoretical computer science, for instance, the study of lenses, open systems, and rewriting.

However, double categories come with a wide variety of equivalences, which makes it challenging to transport structure along equivalences. To deal with this challenge, we propose the *univalence maxim*: each notion of equivalence of categorical structures has a corresponding notion of univalent categorical structure which induces that notion of equivalence. We also prove corresponding univalence principles, which allow us to transport structure and properties along equivalences. In this way, the usually informal practice of reasoning modulo equivalence becomes grounded in an entirely formal logical principle.

We apply this perspective to various double categorical structures, such as (pseudo) double categories and double bicategories. Concretely, we characterize and formalize their definitions in Coq UniMath up to chosen equivalences, which we achieve by establishing their univalence principles.

2012 ACM Subject Classification Theory of computation ⟶ Logic and verification; Theory of computation ⟶ Type theory

Keywords and phrases formalization of mathematics, category theory, double categories, univalent foundations

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.45

Related Version *Previous Version*: `https://arxiv.org/abs/2402.05265`

Supplementary Material *Software*: `https://doi.org/10.5281/zenodo.13828995` [36]

Funding *Niels van der Weide*: This research was supported by the NWO project "The Power of Equality" OCENW.M20.380, which is financed by the Dutch Research Council (NWO).

Acknowledgements We gratefully acknowledge the work by the Coq development team in providing the Coq proof assistant and surrounding infrastructure, as well as their support in keeping UniMath compatible with Coq. We are very grateful to Mike Shulman for answering our questions about profunctors. We would also like to thank Lyne Moser for many valuable discussions and explanations regarding double categories, their equivalences, and important references. The first author is grateful to the Max Planck Institute for Mathematics in Bonn for its hospitality and financial support.

1 Introduction

The advancement of mathematics has resulted in ever more intricate structures, which come with various commonly used identifications, weakening set-theoretic equalities. For instance, groups have one common type of identification: isomorphisms. Two groups have the same group-theoretic properties if and only if they are isomorphic. Hence, structures on, and properties of, groups can be transported along isomorphisms. Already in this first example we can witness the challenge of dealing with equivalences in set-theoretic foundations. Indeed, one needs to prove every time that a suitable structure or property can be transported along isomorphisms of groups, as only transport along equalities would come for free.

If we generalize groups, we encounter one of the first examples of a structure with more than one important type of sameness: categories. Indeed, it comes with two major identifications, isomorphisms and categorical equivalences. Here, our choice depends on whether we study objects of a category up to equality, as is often done in the study of syntax [5, 12], in which case we use isomorphisms; or whether we study objects up to isomorphism, the more common choice, in which case we use categorical equivalences. Hence, we now need suitable results regarding transporting structures and properties both for isomorphisms and categorical equivalences [10, 17, 24], which is significantly more challenging given the amount of data an equivalence entails. At least we can still benefit from the fact that categorical equivalences generalize isomorphism, meaning it often suffices to restrict to the case of categorical equivalences.

The situation becomes untenable when we further generalize to higher categories, and particularly *double categories*, a structure that is a key ingredient in the study of lenses [11, 13], rewriting [9, 8], open systems [7, 6], and programming language theory [28, 14]. Double categories come with objects, two classes of morphisms, known as horizontal and vertical morphisms, and suitable data detailing how they interact, known as squares. This additional structure enables a wide range of configurations and identifications. In particular we can employ the squares to relax the associativity and unitality of the composition of horizontal or vertical morphisms. Also, we can use the intricate structure to define a wide range of equivalences, with the fascinating observation that none of them is more general than all the others.

As a result, a significant part of the double categorical literature has exclusively focused on one type of equivalences, called *horizontal equivalence* (Section 5), which prioritizes horizontal morphisms and hence ignores the natural symmetry between horizontal and vertical morphisms inherent to the definition of a double category. Examples include the work on limits [20], adjunctions [21], formal category theory [31], lenses [13] and open systems [6]. At the same time, symmetric notions of equivalences, such as *gregarious equivalences* (Section 9), have received far less attention, even though they are the natural context for other important double categorical constructions, such as the *square functor* [16]. The square functor is already conjecturally applied to modern aspects of algebraic geometry [18], with other anticipated applications currently hindered by insufficient theoretical advances in this direction. Addressing this situation requires the ability to adjust our definition of double categories based on the equivalence of interest, which can then be employed in the aforementioned examples.

Beyond mathematics, the fact that double categories do not come with a distinguished notion of equivalence also complicates any effort formalizing double category theory in intensional type theories via proof assistants. Indeed, we would like to have suitable principles that permit transporting results regarding double categories along equivalences. However, as

there are several incomparable equivalences in the literature, it is not possible to only have one principle. We would rather need one such transport principle, and corresponding notion of double category, for each equivalence.

The Univalence Maxim. In this paper, we propose the *univalence maxim* to resolve the aforementioned challenges and to provide suitable transport principles, formally. Our univalence maxim takes place in univalent type theory, a variant of Martin-Löf type theory with the univalence axiom.

In type theory, Martin-Löf's identity type is a fundamental concept, capturing formally when two objects are considered "the same". The univalence axiom adds extensionality principles, postulating that the identity type of types coincides with the type of equivalences between these types. That is, two types are identified if we have an equivalence between them, and thus equivalent types have the same properties. This perspective also extends to structures. Indeed, in univalent foundations, each notion of structure comes with a notion of equivalence such that identity of structures corresponds to equivalence. For instance, one can prove that identity of algebraic structures, such as groups, corresponds to isomorphism. The correspondence between identity and equivalence for structures is known as the *structure identity principle*. Such principles allow us to prove directly that structure and properties can be transported along equivalences.

The *univalence maxim* we propose says the following: for every chosen equivalence of a given categorical structure, there exists a tailored definition for which identity precisely coincides with the chosen notion of equivalence. We can already witness manifestations of the maxim for categories, which, as we discussed, can be considered up to isomorphism or up to categorical equivalence. Indeed, in the univalent setting we have two notions of categories, namely *setcategories*, whose properties are automatically invariant under isomorphisms, and *univalent categories*, whose properties are automatically invariant under categorical equivalences. See Section 2 for further details.

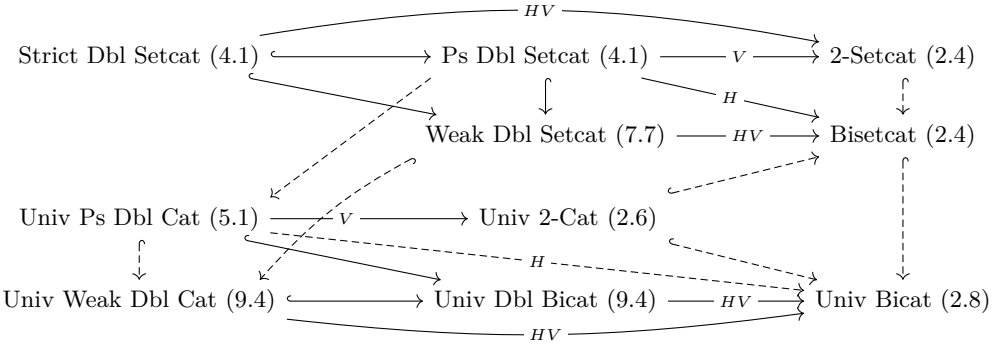
While we can already apply the univalence maxim to categories, its true power manifests in the more complicated framework of double categories. Indeed, beyond obtaining precise transport principles which assist formalization, by establishing a correspondence between various double categorical equivalences and suitably defined double categories, we are for the first time able to structure the existing zoo of double categorical notions that can be found in the literature, the results of which we summarized in the *Diagrammatic Summary* below. Having such principles available also helps with the formalization of double categories, because it allows us to view equivalences via the usual Martin-Löf identity type.

Contributions. In this paper we apply the univalence maxim to categories, 2-categories and double categories. More specifically,

1. in Section 4 we define *(pseudo) double setcategories* and prove that they are invariant under isomorphisms;
2. we provide a further generalization to *double bicategories* (Section 7) and *univalent double bicategories* invariant under gregarious equivalences (Section 9).
3. we also develop the theory of double bicategories, such as companions (Section 8) and computationally feasible methods to establish univalence (Section 10).

We review univalent categories and 2-categories following [3, 2] in Section 2 and *univalent pseudo double categories* in Section 5 following our previous work in [34].

Diagrammatic Summary. We can summarize our major double categorical notions and their relations diagrammatically. Here a dashed arrow represents an inclusion that only respects the categorical structure (i.e. only holds in classical setting), whereas a solid arrow indicates an inclusion that respects categorical structure and univalence conditions. Moreover, an arrow labeled V denotes the underlying vertical 2-category or bicategory (Definitions 3.7 and 7.1), whereas an arrow labeled H is the underlying horizontal 2-category or bicategory (Definitions 3.7 and 7.1).



Formalization. The main results have been formalized using the Coq proof assistant and the UniMath library. Links to the corresponding identifier in the code are in blue.

There are two differences between the formalization and the definitions presented in the paper. While in the paper, we present strict double categories as a special instance of pseudo double categories (Definition 3.1), we use an unfolded approach in the formalization. Second, here we define Verity double bicategories (Definition 7.1) using two bicategories whose types of objects are equal. This is not so in the formalization, where instead a more unfolded approach is used.

Related Work. The study of (higher) categories in univalent foundations has a rich history. Indeed, a study of univalent categories was commenced by Ahrens, Kapulkin, and Shulman in [3], and later extended by Ahrens, Frumin, Maggesi, Veltri, and Van der Weide to a study of univalence for 2-categories and bicategories in [2]. Both these works are reviewed more carefully in Section 2. This existing work on (2-)categories motivated us to pursue univalence principles for double categorical structures. This first commenced with [34], where we focused on univalence principles for horizontal equivalences, given their centrality in the current double categorical literature. In that paper we introduced univalent pseudo double categories and proved that their identities correspond to horizontal equivalences. Further details regarding our past work can be found in Section 5. The current paper is a natural continuation of that effort, generalizing from horizontal equivalences to gregarious equivalences and from double categories to double bicategories. This effort was also motivated by work in [4], where Ahrens, North, Shulman, and Tsementzis established a very broad univalence principle, which in particular applies to univalent double (bi)categories and hence establishes the basis for our work in Section 9.

Double categories acquired some attention from the formalization community, and several libraries on formalized mathematics contain double categories. Murray, Pronk, and Szyld [27] worked towards defining double categories in the Lean proof assistant[1]. In 1lab [32], internal

[1] `https://github.com/leanprover-community/mathlib/pull/18204`

categories are defined, and thus double categories are also defined as category objects in the category of setcategories. Finally, in the library by Hu and Carette [22] a definition of double category has been implemented[2]. Each of these formalization only considers strict double categories, whereas we also consider weaker notions. In addition, our formalization contains a study of various univalence principles.

2 (2-)Categories in Univalent Foundations

In this section we realize the vision of the univalence maxim for categories and bicategories, based on work done in [2, 3]. More precisely, we analyze two notions of equivalences for categories (isomorphisms, equivalences) and three notions of equivalences for 2-categories and bicategories (isomorphisms, equivalences, biequivalences). For each of these notions, we define a categorical structure whose identities correspond to that notion of equivalence.

We start with categories. In classical mathematics a category is defined as a class of objects and a set of morphisms, depending on two objects, with a unital and associative composition of morphisms. In univalent foundations, categories are defined to have a type of objects and a set of morphisms [3, Definition 3.1]. Here a type is called a set if any two identities between two terms are equal.

Note that categories have two important notions of equivalences: isomorphisms and equivalences. Categories that are invariant under isomorphism, are also known has *setcategories*. The other notion of interest, *univalent categories*, are categories that are invariant under equivalence. These notions are defined as follows.

▶ **Definition 2.1.** A category is said to be a **setcategory** if its type of objects is a set. A category C is said to be **univalent** if for all objects $x, y : \mathsf{C}$ the map $\mathsf{idtoiso}_{x,y}$, which sends identities $p : x = y$ to isomorphisms $\mathsf{idtoiso}_{x,y}(p) : x \cong y$, is an equivalence of types.

The univalence condition implies that equalities of categories are precisely categorical equivalences [3, Theorem 6.17], giving us the desired invariance property. Univalent categories provide us with an alternative way to characterize the fact that setcategories are invariant under isomorphisms:

▶ **Proposition 2.2.** *The category of setcategories and functors is univalent.*

Unfortunately, we cannot repeat the argument and incorporate the equivalence invariance of univalent categories into the construction of a univalent category, as the type of equivalences between two univalent categories is generally too complex and does not form a set. However, we can in fact construct a univalent bicategory of univalent categories [2, Proposition 3.19].

▶ **Proposition 2.3.** *The bicategory of univalent categories, functors, and natural transformations is univalent.*

Next we look at bicategories, and 2-categories, which are bicategories with identity associators and unitors. As bicategories not only have objects and 1-morphisms, but also 2-morphisms between 1-morphisms, the number of relevant equivalences increases significantly. Here we focus in particular on three equivalences: isomorphisms of bicategories, equivalences of bicategories (equivalences of the underlying 1-categories that are isomorphisms of 2-morphisms), and biequivalences. Our goal is to construct for each type of equivalence a bicategory (2-category) such that their identities correspond to the chosen equivalence.

[2] `https://github.com/agda/agda-categories/blob/36abe6bff98be027bd4fcc3306d6dac8b2140079/src/Categories/Double/Core.agda`

For the first kind of equivalence, taking Definition 2.1 as motivation we analogously impose appropriate set level restrictions to obtain isomorphism invariance.

▶ **Definition 2.4.** A bicategory (2-category) is said to be a **bisetcategory** (**2-setcategory**) if its type of objects and of 1-morphisms are sets.

▶ **Proposition 2.5.** *The two categories given by bisetcategories and functors, and by 2-setcategories and functors are univalent.*

For the second kind of equivalence, we combine Proposition 2.2 and Proposition 2.3.

▶ **Definition 2.6.** A 2-category is said to be **univalent** if the underlying 1-category is univalent and the 2-morphisms form a set.

▶ **Proposition 2.7.** *Identities of univalent 2-categories correspond to equivalences.*

Finally, we want a 2-categorical structure invariant under biequivalence. In light of Proposition 2.3, all hom categories need to be univalent, which we call the *local univalence* condition. Moreover, we also need a *global univalence* condition, stating that identities of objects correspond to equivalences in the 2-category. In general this means that objects form a 2-type and compositions of 1-morphisms is generally not strictly associative or unital. Hence, this univalence condition only applies to bicategories.

▶ **Definition 2.8.** A bicategory is **univalent** if it is globally and locally univalent.

See [2, Definition 3.1] for a more explicit description of its definition. Finally, univalent bicategories do exhibit the anticipated invariance property; see [4, Example 9.1].

▶ **Proposition 2.9.** *Identities of univalent bicategories correspond to biequivalences.*

3 Definition of Pseudo Double Categories

In the previous section, we reviewed (bi)categories and showed how imposing additional conditions in univalent foundations lead to (bi)categories up to a desired notion of sameness. For the rest of this paper, we conduct a similar analysis for double categorical structures. As mentioned in the introduction, defining a double category is more complex and involves more data, providing a wider range of examples and equivalences. Consequently, our analysis is more challenging and valuable, and more relevant to the broader literature.

In this transitional section we commence with a review of a general definition of pseudo double categories and explicate our examples. Then, in the next two sections we show how imposing additional conditions result in the desired equivalences.

▶ **Definition 3.1.** A **pseudo double category** consists of

1. a category C called the **vertical category**;
2. for all objects $x : \mathsf{C}$ and $y : \mathsf{C}$, a type $x \nrightarrow y$ of **horizontal morphisms**;
3. for every object $x : \mathsf{C}$ a **horizontal identity** $\mathsf{id}_x : x \nrightarrow x$;
4. for all horizontal morphisms $h : x \nrightarrow y$ and $k : y \nrightarrow z$, a **horizontal composition** $h \odot k : x \nrightarrow z$;
5. for all horizontal morphisms $h : x_1 \nrightarrow y_1$ and $k : x_2 \nrightarrow y_2$ and vertical morphisms $v_x : x_1 \nrightarrow x_2$ and $v_y : y_1 \nrightarrow y_2$, a set $\left(v_x \begin{smallmatrix} h \\ k \end{smallmatrix} v_y\right)$ of **squares**;
6. for all horizontal morphisms $h : x \nrightarrow y$, we have a **vertical identity** $\mathsf{id}^v_{\mathsf{sq}}(h) : \left(\mathsf{id}_x \begin{smallmatrix} h \\ h \end{smallmatrix} \mathsf{id}_y\right)$;
7. for all $\tau_1 : \left(v_1 \begin{smallmatrix} h \\ k \end{smallmatrix} w_1\right)$ and $\tau_2 : \left(v_2 \begin{smallmatrix} k \\ l \end{smallmatrix} w_2\right)$, a square $\tau_1 \cdot_{\mathsf{sq}} \tau_2 : \left(v_1 \cdot v_2 \begin{smallmatrix} h \\ l \end{smallmatrix} w_1 \cdot w_2\right)$;

8. for all $v : x \longrightarrow y$, we have a **horizontal identity** $\mathsf{id}^h_{\mathsf{sq}}(v) : \begin{pmatrix} v & \mathsf{id}_x \\ & \mathsf{id}_y \end{pmatrix}$;
9. for all $\tau_1 : \begin{pmatrix} v_1 & h_1 \\ & k_1 \end{pmatrix}$ and $\tau_2 : \begin{pmatrix} v_2 & h_2 \\ & k_2 \end{pmatrix}$, a square $\tau_1 \odot_{\mathsf{sq}} \tau_2 : \begin{pmatrix} v_1 & h_1 \odot h_2 \\ & k_1 \odot k_2 \end{pmatrix}$;
10. for all $h : x \nrightarrow y$, we have a **left unitor** $\lambda_h : \begin{pmatrix} \mathsf{id}_x & \mathsf{id}_x \odot h \\ & h \end{pmatrix}$;
11. for all $h : x \nrightarrow y$, we have a **right unitor** $\rho_h : \begin{pmatrix} \mathsf{id}_x & h \odot \mathsf{id}_y \\ & h \end{pmatrix}$;
12. for all $h_1 : w \nrightarrow x$, $h_2 : x \nrightarrow y$, and $h_3 : y \nrightarrow z$, we have an **associator**

$$\alpha_{(h_1,h_2,h_3)} : \begin{pmatrix} \mathsf{id}_w & h_1 \odot (h_2 \odot h_3) \\ & (h_1 \odot h_2) \odot h_3 \end{pmatrix} \mathsf{id}_z \Big).$$

This data is required to satisfy several laws, which are found in the literature [34].

A **strict double category** is a pseudo double category in which horizontal morphisms form a set and in which the unitors and associators are identities.

We now give precise definitions of the four major classes of examples that play a prominent role throughout this paper: squares, spans, structured cospans and profunctors.

▶ **Example 3.2.** Let C be a category. Then we define a pseudo double category $\mathsf{Sq}(\mathsf{C})$ **of squares** . The objects are objects in C and the horizontal and vertical morphisms are morphisms in C. The type of squares $\begin{pmatrix} v_1 & h_1 \\ & h_2 \end{pmatrix} v_2 \Big)$ is defined to be $h_1 \cdot v_2 = v_1 \cdot h_2$.

▶ **Example 3.3.** Let C be a category with pullbacks. Then we define a pseudo double category $\mathsf{Span}(\mathsf{C})$ **of spans** . The objects are objects in C and the vertical morphisms are morphisms in C. The horizontal morphisms are spans, which are diagrams of the form $x \xleftarrow{\varphi} z \xrightarrow{\psi} y$; and Given morphisms $f : x_1 \longrightarrow x_2$ and $g : y_1 \longrightarrow y_2$, then a square with f and g as vertical sides and spans $x_1 \xleftarrow{\varphi_1} z_1 \xrightarrow{\psi_1} y_1$ and $x_2 \xleftarrow{\varphi_2} z_2 \xrightarrow{\psi_2} y_2$ as horizontal sides is a morphism $h : z_1 \longrightarrow z_2$ such that the following diagram commutes.

$$\begin{array}{ccccc} x_1 & \xleftarrow{\varphi_1} & z_1 & \xrightarrow{\psi_1} & y_1 \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle g} \\ x_2 & \xleftarrow[\varphi_2]{} & z_2 & \xrightarrow[\psi_2]{} & y_1 \end{array}$$

Even if C is a setcategory, $\mathsf{Span}(\mathsf{C})$ is generally only a pseudo double category. This is because composition of spans is given by pullbacks, which is only weakly unital and associative. Note that spans have been used in the study of rewriting systems [9, 8].

▶ **Example 3.4.** Suppose that we have a functor $L : \mathsf{C}_1 \longrightarrow \mathsf{C}_2$. We define the double category $\mathsf{StructCospan}(L)$ **of structured cospans** . The objects are objects in C_1 and the vertical morphisms are morphisms in C_1. The horizontal morphisms are **structured cospans**, which are diagrams of the form $L(x) \xrightarrow{\varphi} z \xleftarrow{\psi} L(y)$; Given two structured cospans $L(x_1) \xrightarrow{\varphi_1} z_1 \xleftarrow{\psi_1} L(y_1)$ and $L(x_2) \xrightarrow{\varphi_2} z_2 \xleftarrow{\psi_2} L(y_2)$, and two morphisms $f : x_1 \longrightarrow x_2$ and $g : y_1 \longrightarrow y_2$, a square consists of a morphism $h : z_1 \longrightarrow z_2$ such that the following diagram commutes

$$\begin{array}{ccccc} L(x_1) & \xrightarrow{\varphi_1} & z_1 & \xleftarrow{\psi_1} & L(y_1) \\ {\scriptstyle L(f)}\downarrow & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle L(g)} \\ L(x_2) & \xrightarrow[\varphi_2]{} & z_2 & \xleftarrow[\psi_2]{} & L(y_1) \end{array}$$

Note that structured cospans are used to study open systems [7, 6].

▶ **Example 3.5.** We define the pseudo double category Prof **of profunctors** . The objects are small setcategories and the vertical morphisms are functors. The horizontal morphisms are profunctors from a category C to D, meaning functors of the form $\mathsf{D}^{\mathsf{op}} \times \mathsf{C} \longrightarrow \mathsf{Set}$, and given profunctors $P : \mathsf{C}_1 \longrightarrow \mathsf{D}_1$ and $Q : \mathsf{C}_2 \longrightarrow \mathsf{D}_2$ and functors $F : \mathsf{C}_1 \longrightarrow \mathsf{C}_2$ and $G : \mathsf{D}_1 \longrightarrow \mathsf{D}_2$, we define squares $\left(F \begin{smallmatrix} P \\ Q \end{smallmatrix} G\right)$ to be natural transformations $P \Rightarrow (F \times G) \cdot Q$. Again, this pseudo double category will not be a strict double category, as profunctors do not compose strictly. Note that profunctors are important in the study of lenses [11, 13].

Note that the composition of profunctors is defined as a colimit in the category of sets. To guarantee that the desired colimit exists, we require that the involved categories are small.

▶ **Example 3.6** (Gradual type theory, [28, Definition 5.2])**.** Let C be a 2-category. We define a **pseudo double category** Coreflect(C) **of coreflections** . The objects are objects in C, the vertical morphisms are morphisms in C, and the horizontal morphisms are adjunctions in C whose unit is an equality (i.e., coreflections). The squares are given by 2-cells in C.

Double categories include the data of several (2-)categories. These are known as the *underlying 2-categories*, and they are defined as follows.

▶ **Definition 3.7.** Given a pseudo double category C, we define a strict 2-category $\mathsf{Ver}(\mathsf{C})$, which we call the **underlying vertical 2-category** as the 2-category whose objects are objects in C, whose 1-cells are vertical morphisms in C, and whose 2-cells are squares with horizontal identity sides.

In addition, we define a bicategory $\mathsf{Hor}(\mathsf{C})$, which we call the **underlying horizontal bicategory** , as the bicategory whose objects are objects in C, whose 1-cells are horizontal morphisms in C, and whose 2-cells are squares with vertical identity sides.

4 (Pseudo) Double Set-categories

Having defined strict and pseudo double categories, we now impose conditions to obtain isomorphism invariant definitions, following the vision of the univalence maxim. Concretely, we first construct isomorphism invariant notions of (pseudo) double categories and then study several classes of examples. Following Section 2 and Definition 2.1, we need to impose a set level condition to obtain isomorphism invariance, motivating the following definitions.

▶ **Definition 4.1.** A **strict double setcategory** is a strict double category whose objects form a set. In addition, a **pseudo double setcategory** is a double category whose objects and horizontal morphisms form a set.

Using similar ideas to the one used in Proposition 2.2, we confirm their desired invariance.

▶ **Theorem 4.2.** *The category of strict double setcategories, with objects being strict double setcategories and morphisms being strict double functors, is univalent.*

▶ **Theorem 4.3.** *The category of pseudo double setcategories, with objects being pseudo double setcategories and morphisms being strict double functors, is univalent.*

We now review our motivating examples in light of this invariance property.

▶ **Proposition 4.4.** *Let* C *be a setcategory. Then* $\mathsf{Sq}(\mathsf{C})$ *is a strict double setcategory.*

▶ **Proposition 4.5.** *For a setcategory* C *with pullbacks,* $\mathsf{Span}(\mathsf{C})$ *is a pseudo double setcategory.*

▶ **Proposition 4.6.** *Suppose that* C_1 *and* C_2 *are setcategories and that* C_2 *has pushouts. Then* $\mathsf{StructCospan}(L)$ *is a pseudo double setcategory.*

▶ **Proposition 4.7.** *Let* C *be a 2-setcategory. Then* $\mathsf{Coreflect}(\mathsf{C})$ *as defined in Example 3.6 is a pseudo double setcategory.*

The previous examples are very consistent with our intuition that an isomorphism invariant category should indeed result in isomorphism invariant double categories of squares, (co)spans, coreflections, or polynomials. Observe the class of examples of profunctors, Example 3.5, is in fact not a pseudo double setcategory. See Proposition 5.7 for a more detailed discussion. Let us instead present one further example coming from 2-category theory.

▶ **Example 4.8.** For a bisetcategory B we define the pseudo double setcategory $\widehat{\mathsf{B}}$, whose objects are objects in B. Vertical morphisms are morphisms in B, and horizontal morphisms $x \nrightarrow y$ are identities $x = y$. Squares $\left(\begin{smallmatrix} f & p \\ & q & g \end{smallmatrix}\right)$ are 2-cells $f \cdot \mathsf{idtoiso}(q) \Rightarrow \mathsf{idtoiso}(p) \cdot g$.

We end this section by observing that the set condition is preserved by taking underlying bicategories. This results supports the intuitive fact that taking underlying 2-categories preserves isomorphism invariance.

▶ **Proposition 4.9.** *If* C *is a strict double setcategory, then* $\mathsf{Ver}(\mathsf{C})$ *is a strict* 2*-setcategory.*

▶ **Proposition 4.10.** *If* C *is a pseudo double setcategory, then* $\mathsf{Ver}(\mathsf{C})$ *is a strict* 2*-setcategory.*

▶ **Proposition 4.11.** *If* C *is a pseudo double setcategory, then* $\mathsf{Hor}(\mathsf{C})$ *is a bisetcategory.*

5 Univalent Pseudo Double Categories

In the previous section we focused on isomorphism invariance. In this section we continue realizing our univalence maxim, this time studying pseudo double categories invariant under *vertical equivalences*, which are characterized by inducing equivalences on the underlying vertical category and for any two objects x, y inducing equivalences on the category given by horizontal morphisms $x \nrightarrow y$ and squares with trivial vertical sides. Our analysis relies on our previous work done in [34]. We then end this section analyzing several examples.

Following Definition 2.1, we would expect a univalence condition for these two underlying categories. This, however, implies that the underlying horizontal category is neither univalent nor has a set of objects, meaning horizontal composition cannot be strict. Thus we have to work with pseudo double categories, resulting in the following definition.

▶ **Definition 5.1.** A pseudo double category C is said to be **univalent** if its underlying vertical category is univalent and if for all $x, y : \mathsf{C}$ the category whose objects are horizontal morphisms $x \nrightarrow y$ and whose morphisms are squares with vertical identity sides, is univalent.

Building on our insights in Proposition 2.3, we similarly construct a univalent bicategory of univalent pseudo double categories [34].

▶ **Theorem 5.2.** *The bicategory of univalent pseudo double categories with lax double functors is univalent.*

Note that we use lax double functors in Theorem 5.2 whereas we use strict double functors in Theorems 4.2 and 4.3. As the univalence condition is motivated by vertical equivalences, it is not symmetric. For examples identities of objects only correspond to vertical isomorphisms, and identities of horizontal morphisms correspond to isomorphisms of squares (composed vertically). However, some double categories satisfy a stronger univalence condition that is in fact symmetric.

▶ **Definition 5.3.** A pseudo double category C is said to be **symmetrically univalent** if the horizontal morphisms form a set, C is univalent, the category of objects and horizontal morphisms is univalent, and for all $x, y : \mathsf{C}$ the category of vertical morphisms $x \longrightarrow y$ and squares with horizontal identity sides, is univalent.

Let us present a variety of examples of univalent and symmetrically univalent pseudo double categories. Again we focus on our three classes of interest, namely squares, spans and profunctors [34], but we also provide additional examples.

▶ **Proposition 5.4.** *If* C *is a univalent category, then* $\mathsf{Sq}(\mathsf{C})$ *is a symmetrically univalent pseudo double category.*

▶ **Proposition 5.5.** *Let* C *be a univalent category with pullbacks. Then the pseudo double category* $\mathsf{Span}(\mathsf{C})$*, is univalent, but not symmetrically univalent.*

▶ **Proposition 5.6.** *If* C_1 *and* C_2 *are univalent categories such that* C_2 *has pushouts, then* $\mathsf{StructCospan}(L)$ *is a univalent pseudo double category.*

▶ **Proposition 5.7.** *The pseudo double category of profunctors is univalent.*

▶ **Proposition 5.8.** *If* C *is a univalent 2-category. Then* $\mathsf{Coreflect}(\mathsf{C})$ *is a univalent pseudo double category.*

In the later sections, we analyze enriched versions of double categories of profunctors benefiting from appropriately defined weak double categories (Example 7.5). However, if we focus on categories enriched over a poset, which includes quantales and has found applications in automata theory [1, 30] and fuzzy logic [15], we do get stricter double categories.

▶ **Proposition 5.9.** *Suppose that* V *is a complete and cocomplete symmetric monoidal category and suppose that* V *is a poset. Then we have a univalent pseudo double category whose objects are univalent categories enriched over* V*, vertical morphisms are enriched functors, horizontal morphisms are enriched profunctors, and whose squares are given by enriched natural transformations.*

▶ Remark 5.10. Note we cannot construct a univalent pseudo double category given by univalent categories, functors and profunctors. Indeed, the type of univalent categories is a 2-type as it includes all 1-types, by [33, Example 9.9.6] and [2, Example 2.18], and hence cannot be the objects of a univalent category, which is at most a 1-type ([3, Lemma 3.8]).

In the last section we observed that double setcategories (both pseudo and strict) preserve isomorphism invariance when taking underlying bicategories. We might hence anticipate similar results for univalent pseudo categories. Unfortunately we only have a partial result. Indeed, vertical equivalences are preserved by taking the underlying vertical 2-category, confirmed by the following result.

▶ **Proposition 5.11.** *If* D *is univalent, then so is* $\mathsf{Ver}(\mathsf{D})$*.*

However, it is generally untrue that univalent pseudo double categories will induce globally univalent underlying horizontal bicategories. Indeed, the underlying horizontal bicategory of the pseudo double category Prof is given by small setcategories, profunctors and 2-morphisms, and we already observed in Proposition 5.7 that profunctors can be an isomorphism without the underlying categories being isomorphic. However, not all hope is lost and we do recover a local univalence condition.

▶ **Proposition 5.12.** *If* D *is a univalent, then* $\mathsf{Hor}(\mathsf{D})$ *is locally univalent.*

Note that $\mathsf{Hor}(\mathsf{D})$ is not necessarily univalent. This necessitates a double categorical notion that accommodates biequivalences of bicategories.

6 Motivating Verity Double Bicategories

Pseudo-double categories only exhibit non-strict compositions in the horizontal direction and are hence unable to incorporate all relevant examples (Remark 5.10) or biequivalences as an invariant (Proposition 5.12). We hence require a notion of a doubly weak double category with non-strict compositions in both directions. However, providing a direct definition similar to Definition 3.1 results in a fully faithful embedding from pseudo double-categories that does not preserve vertical equivalences. In univalent foundations, in addition to the non-preservation of equivalences, this embedding also does not preserve univalence. Concretely, due to the strictness of vertical compositions, identities in the type of objects correspond to vertical **isomorphisms**, whereas the weak vertical composition and symmetric nature of weak double categories demands that identities correspond to a pair of horizontal and vertical **equivalences** which interact well with each other, i.e. form a companion pair (Section 8). Hence, a univalent pseudo double category results in a non-univalent weak double category.

Univalent foundations hence can propose a solution to non-preservation of equivalences, namely by introducing a *pre-double categorical structure*, whose notion of univalence can incorporate both univalent pseudo double categories and univalent weak double categories, which will in particular imply preservation of equivalences. Taking the previous paragraph as motivation what such a structure should entail is a notion of 2-morphisms divorced from the 2-morphisms induced by squares (as defined in Definition 3.7). By appropriately choosing the 2-cells we can then restrict to both cases of interest:

- If we choose the vertical 2-cells to be identities then equivalences in the vertical 2-category would be isomorphism, recovering the identities of univalent pseudo-categories;
- if we choose the vertical (and horizontal) 2-cells to coincide with 2-cells induced by squares, then identities of objects correspond to equivalences we expect to see in weak double categories.

Thus, in order to pursue our study of equivalences of double categories, we first develop pre-double category theory, whose structure involves objects, horizontal (vertical) morphisms, horizontal (vertical) 2-morphisms, and squares, along with appropriately coherent compositions. Fortunately, a suitable candidate for such a notion has already been proposed by Verity , where it is called a *double bicategory* [35], along with a univalence principle [4]. Hence, for the remainder of the paper the aim is to study and formalize double bicategories, study its univalence principle, and establish an embedding from univalent pseudo categories to univalent double bicategories.

7 Verity Double Bicategories

Following the discussion of Section 6 we commence with the definition and formalization of double bicategories due to Verity [35], and also describe various examples. In Section 9 we will then pursue its univalence principles.

▶ **Definition 7.1.** A **Verity double bicategory** B consists of

1. a bicategory $\mathsf{H_B}$ whose objects, 1-cells, and 2-cells are called **horizontal**;
2. a bicategory $\mathsf{V_B}$ with the same type of objects as $\mathsf{H_B}$, and whose objects, 1-cells, and 2-cells are called **vertical**;
3. for all objects $x_1, x_2, y_1, y_2 : \mathsf{H_B}$, horizontal 1-cells $h_1 : x_1 \longrightarrow x_2$ and $h_2 : y_1 \longrightarrow y_2$, and vertical 1-cells $v_1 : x_1 \longrightarrow y_1$ and $v_2 : x_2 \longrightarrow y_2$ a set $\begin{pmatrix} v_1 & {\scriptstyle h_1} \\ & {\scriptstyle h_2} \end{pmatrix} v_2$ of **squares**;
4. for all horizontal 1-cells $h : x \longrightarrow y$ a square $\mathsf{id}^h_{\mathsf{sq}}(h) : \begin{pmatrix} \mathsf{id}_x & {\scriptstyle h} \\ {\scriptstyle h} & \mathsf{id}_y \end{pmatrix}$;

5. for all vertical 1-cells $v : x \longrightarrow y$ a square $\mathsf{id}^{v}_{\mathsf{sq}}(v) : \begin{pmatrix} v & \mathsf{id}_x \\ & \mathsf{id}_y & v \end{pmatrix}$;
6. for all $s_1 : \begin{pmatrix} v_1 & h \\ & k & w_1 \end{pmatrix}$ and $s_2 : \begin{pmatrix} v_2 & k \\ & l & w_2 \end{pmatrix}$, a square $s_1 \cdot_{\mathsf{sq}} s_2 : \begin{pmatrix} v_1 \cdot v_2 & h \\ & l & w_1 \cdot w_2 \end{pmatrix}$;
7. for all $s_1 : \begin{pmatrix} v_1 & h_1 \\ & k_1 & v_2 \end{pmatrix}$ and $s_2 : \begin{pmatrix} v_2 & h_2 \\ & k_2 & v_3 \end{pmatrix}$, a square $s_1 \odot_{\mathsf{sq}} s_2 : \begin{pmatrix} v_1 & h_1 \odot h_2 \\ & k_1 \odot k_2 & v_3 \end{pmatrix}$;
8. for all vertical 2-cells $\tau : v_1 \Rightarrow v_2$ and squares $s : \begin{pmatrix} v_2 & h \\ & k & w \end{pmatrix}$, a square $\tau \lhd s : \begin{pmatrix} v_1 & h \\ & k & w \end{pmatrix}$;
9. for all vertical 2-cells $\tau : w_1 \Rightarrow w_2$ and squares $s : \begin{pmatrix} v & h \\ & k & w_1 \end{pmatrix}$, a square $\tau \rhd s : \begin{pmatrix} v & h \\ & k & w_2 \end{pmatrix}$;
10. for all horizontal 2-cells $\tau : h_1 \Rightarrow h_2$ and squares $s : \begin{pmatrix} v & h_2 \\ & k & w \end{pmatrix}$, a square $\tau \vartriangle s : \begin{pmatrix} v & h_1 \\ & k & w \end{pmatrix}$;
11. for all horizontal 2-cells $\tau : k_1 \Rightarrow k_2$ and squares $s : \begin{pmatrix} v & h \\ & k_1 & w \end{pmatrix}$, a square $\tau \triangledown s : \begin{pmatrix} v & h \\ & k_2 & w \end{pmatrix}$.

We call $\mathsf{H_B}$ the **underlying horizontal bicategory** and $\mathsf{V_B}$ the **underlying vertical bicategory**. In addition to the data explicated here, we have various laws governing their behavior. See the formalization or [35, Definition 1.4.1] for further details

Let us note that this definition does in fact satisfy all the desired conditions outlined in Section 6. Indeed, we have independently defined horizontal and vertical 2-cells and compositions of all 1-morphisms are defined weakly, giving us a symmetric definition. Following our vision, we now define *weak double categories* by adding an appropriate saturation condition identifying 2-cells with certain squares.

▶ **Definition 7.2.** Suppose we have a Verity double bicategory B. For all horizontal 1-cells $h_1, h_2 : x \longrightarrow y$ we have a map $\mathsf{CellToSq_H}$ sending 2-cells $\tau : h_1 \Rightarrow h_2$ to the square $\tau \vartriangle \mathsf{id}^{h}_{\mathsf{sq}}(h_2) : \begin{pmatrix} \mathsf{id}_x & h_1 \\ & h_2 & \mathsf{id}_y \end{pmatrix}$. We say that B is **horizontally saturated** if $\mathsf{CellToSq_H}$ is an equivalence. Similarly, the map $\mathsf{CellToSq_V}$ sends vertical 2-cells $\tau : v_1 \Rightarrow v_2$ to the square $\tau \lhd \mathsf{id}^{v}_{\mathsf{sq}}(v_2) : \begin{pmatrix} v_1 & \mathsf{id}_x \\ & \mathsf{id}_y & v_2 \end{pmatrix}$, and B is **vertically saturated** if $\mathsf{CellToSq_V}$ is an equivalence. A **weak double category** is a horizontally and vertically saturated Verity double bicategory.

Our definition of weak double categories by definition comes with an inclusion in Verity double bicategories. We now establish the second inclusion suggested in Section 6 and show that every pseudo double category gives rise to a Verity double bicategory.

▶ **Example 7.3.** Suppose that we have a pseudo double category C. We define a Verity double bicategory $\overline{\mathsf{C}}$. Its underlying horizontal bicategory is the discrete bicategory on the underlying vertical category of C, and the underlying vertical bicategory is the underlying horizontal bicategory of C. The squares are defined to be squares in C. Note that $\overline{\mathsf{C}}$ is vertically saturated, but not necessarily horizontally saturated.

Notice that the assignment of the horizontal 2-cells in Example 7.3 is not unique, and our choice is motivated by the desire to realize our programme, meaning to guarantee that univalent pseudo-double categories give us univalent Verity double bicategories. See Remark 10.10 for more details

We now proceed to look at several examples of Verity double bicategories. By Example 7.3, every pseudo double category results in a Verity double bicategory. So here we focus on examples giving us weak double categories, again motivated by our three classes of examples introduced in Section 3, squares, profunctors and spans.

▶ **Example 7.4.** Let B be a bicategory. We define a Verity double bicategory $\mathsf{Sq(B)}$ **of squares** . The horizontal bicategory $\mathsf{H_{Sq(B)}}$ is B, and the vertical bicategory $\mathsf{V_{Sq(B)}}$ is $\mathsf{B^{co}}$. The squares $\begin{pmatrix} v & h \\ & k & w \end{pmatrix}$ are defined to be 2-cells $h \cdot w \Rightarrow v \cdot k$. Note that $\mathsf{Sq(B)}$ is both horizontally and vertically saturated, meaning it is a weak double category.

In Example 7.4, the 2-cells in the vertical bicategory $\mathsf{V_{Sq(B)}}$ are reversed. This is necessary to get the right whiskering operations.

▶ **Example 7.5.** We define a Verity double bicategory Prof **of profunctors** . The underlying horizontal bicategory $\mathsf{H}_{\mathsf{Prof}}$ is the bicategory $\mathsf{UnivCat}^{\mathsf{co}}$ of small univalent categories, and the objects of $\mathsf{V}_{\mathsf{Prof}}$ are small univalent categories, the 1-cells are profunctors, and the 2-cells are natural transformations. The squares are defined in the same way as in Example 3.5.

The identity and composition operations for profunctors are defined in the same way as in Example 3.5. For the details of the whiskering constructions, we refer the reader to the formalization. Note that Prof is both vertically and horizontally saturated, giving us a weak double category.

Similarly, we define the Verity double bicategory $\mathsf{Prof}_{\mathsf{V}}$ **of enriched profunctors** . Let V be a complete and cocomplete symmetric monoidal closed category. The underlying horizontal bicategory $\mathsf{H}_{\mathsf{Prof}_{\mathsf{V}}}$ is the bicategory $\mathsf{UnivCat}_{\mathsf{V}}^{\mathsf{co}}$ of small univalent enriched categories, and the objects of the underlying vertical $\mathsf{V}_{\mathsf{Prof}_{\mathsf{V}}}$ are small univalent categories, 1-cells are enriched profunctors, and 2-cells are enriched natural transformations. The squares are defined in a similar way as in Example 3.5.

Identity, composition, and whiskering operations are defined similarly to Example 7.5. Since $\mathsf{Prof}_{\mathsf{V}}$ is both vertically and horizontally saturated, it is a weak double category.

Note here we use small categories in Example 7.5 to guarantee that the desired coends exist, analogous to Example 3.5.

Unlike the previous sections we do not construct a weak double category of spans in a bicategory, as such a construction would require additional categorical layers; see also [25, Section 4]. However, we do have one additional example motivated by *mate calculus*.

▶ **Example 7.6** (Mate calculus, [23, Proposition 2.2])**.** Let B be a bicategory. We define a Verity double bicategory LAdj(B) . The horizontal bicategory $\mathsf{H}_{\mathsf{LAdj(B)}}$ is the bicategory whose objects are objects in B, 1-cells are left adjoints, and 2-cells are mate-pairs, and the vertical bicategory $\mathsf{V}_{\mathsf{LAdj(B)}}$ is B^{co}. Given adjunctions $h : x \dashv y$ and $k : x' \dashv y'$, and 1-cells $v : x \longrightarrow x'$ and $w; y \longrightarrow y$', squares $\left(v \begin{smallmatrix} h \\ k \end{smallmatrix} w\right)$ are defined to be 2-cells $h \cdot w \Rightarrow v \cdot k$.

▶ Remark 7.7. Similar to Section 4, we can impose a set condition on the types of objects and morphisms to define *Verity double bisetcategories* and *weak double setcategories*, whose identities, analogous to Theorems 4.2 and 4.3, correspond to isomorphisms. However, similar to the classical setting (Section 6), pseudo double setcategories fully faithfully embeds in weak double setcategories, hence obviating the need to generalize to any pre-double categorical notion, such as Verity double bicategories.

Given these similarities, we proceed to the study of univalent Verity double bicategories and their relation to univalent pseudo categories and univalent weak double categories.

8 Companion Pairs

In the previous section we defined Verity double bicategories and showed that this general notion includes both pseudo double categories and weak double categories. Our major aim is to show that these assignments preserves univalence. Due to the symmetric nature of Verity double bicategories, equivalences of objects are symmetric as well, so they need to be given by a pair of horizontal and vertical equivalences that interact well with each other. In this section we provide a precise characterization of the interaction between horizontal and vertical morphisms, via *companion pairs*.

▶ **Definition 8.1.** Suppose that we have a Verity double bicategory B and a horizontal morphism $h : x \longrightarrow y$ and a vertical morphism $v : x \longrightarrow y$. Then we say that h and v form a **companion pair** if we have squares $\eta : \begin{pmatrix} v & {\scriptstyle h} \\ {\scriptstyle \mathsf{id}_y} & \mathsf{id}_y \end{pmatrix}$ and $\varepsilon : \begin{pmatrix} \mathsf{id}_x & {\scriptstyle \mathsf{id}_x} \\ {\scriptstyle h} & v \end{pmatrix}$ such that the squares $\rho \triangleright \ell^{-1} \triangleleft (\varepsilon \odot_{\mathsf{sq}} \eta)$ and $\rho \triangledown \ell^{-1} \triangle (\varepsilon \cdot_{\mathsf{sq}} \eta)$ are identity squares. Here, ℓ is the left unitor. We call η the **unit** and ε the **counit**.

An alternative approach to the interaction between horizontal and vertical morphisms is given by *conjoints*, which are companion pairs in the horizontal dual of B. This notion is prevalent in *formal category theory* [29, 37, 38].

Beyond the definition we also need several key properties of companion pairs that we confirm here. Concretely, we want to know that companion pairs include identities and are closed under composition. Moreover, companions, if they exist, are unique up to isomorphism. Finally, companions of equivalences are also equivalences.

▶ **Proposition 8.2.** *Given an object x in a Verity double bicategory B, then the horizontal identity id_x and vertical identity id_x form a companion pair.*

▶ **Proposition 8.3.** *Given companion pairs $h_1 : x \longrightarrow y$ and $v_1 : x \longrightarrow y$, and $h_2 : y \longrightarrow z$ and $v_2 : y \longrightarrow z$, then $h_1 \cdot h_2$ and $v_1 \cdot v_2$ also form a companion pair.*

▶ **Proposition 8.4.** *Let B be a Verity double bicategory such that H_{B} and V_{B} are locally univalent and such that B is vertically saturated. For every horizontal 1-cell $h : x \longrightarrow y$, we have that all vertical 1-cells $v, v' : x \longrightarrow y$ that form a companion pair with h are equal.*

▶ **Proposition 8.5.** *Suppose that we have a Verity double bicategory B such that B is vertically saturated. Given a horizontal adjoint equivalence $l \dashv r$ such that l and r have companion pairs l' and r' respectively, then we have a vertical adjoint equivalence given by $l' \dashv r'$.*

In many important examples of Verity double bicategories, every horizontal 1-morphism has a companion, which is a key ingredient towards establishing its univalence condition. This holds for $\mathsf{Sq}(\mathsf{B})$, Prof, and $\mathsf{Prof}_{\mathsf{V}}$, whereas if we have a pseudo double category C, then $\overline{\mathsf{C}}$ has companion pairs if C has.

▶ **Proposition 8.6.** *Let B be a bicategory. Given a 1-cell $f : x \longrightarrow y$ in B, then f and f form a companion pair in $\mathsf{Sq}(\mathsf{B})$.*

▶ **Proposition 8.7.** *Suppose that we have a functor $F : \mathsf{C}_1 \longrightarrow \mathsf{C}_2$. Note that F gives rise to a profunctor $\mathsf{rep}_\ell(F) : \mathsf{C}_1 \longrightarrow \mathsf{C}_2$ that sends objects $x : \mathsf{C}_1$ and $y : \mathsf{C}_2$ to the set of morphisms $F(y) \longrightarrow x$. Then F and $\mathsf{rep}_\ell(F)$ form a companion pair.*

9 Univalent Double Bicategories

In this section we use companion pairs introduced in Section 8 to present a univalence principle for double bicategories (Section 7), further advancing our general maxim. Given the amount data a Verity double bicategory involves, we split up the univalence condition into two parts. The first one is a local conditions imposed on the hom-categories in the underlying horizontal and vertical bicategories.

▶ **Definition 9.1.** A Verity double bicategory B is said to be **locally univalent** if both H_{B} and V_{B} are locally univalent.

The second univalence condition is global and focuses on the type of objects in Verity double bicategories. Here we use companion pairs.

▶ **Definition 9.2.** A **gregarious equivalence** from x to y in a Verity double bicategory B consists of a horizontal adjoint equivalence $h : x \longrightarrow y$, a vertical adjoint equivalence $v : x \longrightarrow y$ such that h and v form a companion pair.

▶ **Proposition 9.3.** *Given an object x in a Verity double bicategory, then the horizontal identity id_x and the vertical identity id_x form a gregarious equivalence.*

Following [4], we define *gregarious univalence* using gregarious equivalences.

▶ **Definition 9.4.** Given a Verity double bicategory and objects x and y, we define the map $\mathsf{IdToGregEq}_{x,y}$ sending identities $x = y$ to gregarious equivalences using path induction and the fact that the identity is a gregarious equivalence (Proposition 9.3). A Verity double bicategory is said to be **gregarious univalent** if the map $\mathsf{IdToGregEq}_{x,y}$ is an equivalence of types for all x and y.

Following the univalence principle in [4, Example 9.3], identities of gregarious univalent Verity double bicategories are equivalent to the type of gregarious equivalences between them. Finally, by Definition 7.2, a weak double category is univalent if it is univalent as a double bicategory, confirming our intuition presented in Section 6.

10 Univalence and Weak Horizontal Invariance

In this final section we tie up our discussion of univalent double categorical structures, by establishing the following two facts. First, there is a large class of univalent double bicategories, such as $\mathsf{Sq}(\mathsf{B})$, Prof, and $\mathsf{Prof}_{\mathsf{V}}$. Second, every univalent pseudo double category gives rise to a univalent Verity double bicategory.

To deal with the complicated nature of gregarious univalence, we use a more conceptual approach. We define a notion of *weakly horizontally invariant double bicategory* and show that in this case (with some minor conditions) gregarious univalence reduces to horizontal univalence. We end this section with checking these two properties for our cases of interest.

Let us start with the definition of weak horizontal invariance.

▶ **Definition 10.1.** A Verity double bicategory is **weakly horizontally invariant** if every horizontal adjoint equivalence has a companion pair.

▶ **Proposition 10.2.** *Let B be a Verity double bicategory such that H_{B} is globally univalent. Then B is weakly horizontally invariant.*

This is because the horizontal identity has a companion, and to construct companions for arbitrary adjoint equivalence, we use induction on adjoint equivalences. Proving the main theorem requires the following proposition, which characterizes gregarious equivalences.

▶ **Proposition 10.3.** *Let B be a weakly horizontally invariant Verity double bicategory such that B is vertically saturated and such that H_{B} and V_{B} are locally univalent. Then a horizontal morphism $h : x \longrightarrow y$ is an adjoint equivalence if and only if we have a vertical 1-cell $v : x \longrightarrow y$ such that h and v are a gregarious equivalence.*

▶ **Theorem 10.4.** *Let B be a locally univalent, vertically saturated, and weakly horizontally invariant Verity double bicategory. Then B is gregarious univalent if and only if the bicategory H_{B} is globally univalent.*

We now apply Theorem 10.4 to our cases of interest.

▶ **Proposition 10.5.** *Let* B *be a univalent bicategory. Then* $\mathsf{Sq}(\mathsf{B})$ *is weakly horizontally invariant and univalent.*

▶ **Proposition 10.6.** Prof *is weakly horizontally invariant and univalent.*

▶ **Proposition 10.7.** $\mathsf{Prof}_{\mathsf{V}}$ *is weakly horizontally invariant and univalent.*

▶ **Proposition 10.8.** *If* B *be a univalent bicategory, then* $\mathsf{LAdj}(\mathsf{B})$ *is weakly horizontally invariant and univalent.*

▶ **Proposition 10.9.** *Suppose that we have a univalent pseudo double category* C*. Then the double bicategory* $\overline{\mathsf{C}}$ *is weakly horizontally invariant and univalent.*

▶ Remark 10.10. The choice of 2-cells in Example 7.3 was not unique, and our choice was motivated by Proposition 10.9. To obtain gregarious univalence, the horizontal 2-morphisms need to be trivial so that identities are given by isomorphisms. For instance, in the univalent pseudo-double category of setcategories, functors and profunctors, the resulting Verity double bicategory is univalent because the 2-cells are identities.

Let us end by observing that weak horizontal invariance has already been employed to study equivalences of double categories, however, from a categorical perspective [26].

11 Conclusion

In this paper we presented a connection between equivalences of categorical structures and formalizations thereof in `UniMath`. More specifically, we introduced the univalence maxim for categorical structures, which says that every notion of equivalence comes with a corresponding notion of univalent categorical structure, whose identity type corresponds to the given notion of equivalence. We studied the maxim for many different examples, namely categories, 2-categories, and double categories.

As a result of the maxim, we realize that the univalent setting is the appropriate framework to articulate and formalize (higher) categorical definitions and results, as it provides direct access to valuable transport principles along (higher) categorical equivalences of interest, by transforming categorical equivalences into identities (Proposition 2.9 and Theorem 5.2). Moreover, the univalence maxim empowers us to compare and contrast different double categorical notions, tying together disparate results in the double categorical literature and facilitating a holistic approach to all double categorical notions. Concretely, in the univalent setting, we can compare double categories that differ in their strictness properties, for example strict vs. pseudo double categories, and double categories that differ in their chosen notion of sameness, for example pseudo double categories up to isomorphism vs. pseudo double categories up to horizontal equivalence. Here we note that the second type of comparison is not possible in set theoretical foundations.

Finally, let us note that in several cases we obtained a univalence principle by establishing the univalence of an appropriately defined (bi)category. One future aim is to generalize this approach to bicategories and Verity double bicategories, which would require developing *tricategory theory* [19].

References

1 Samson Abramsky and Steven Vickers. Quantales, observational logic and process semantics. *Math. Structures Comput. Sci.*, 3(2):161–227, 1993. doi:10.1017/S0960129500000189.

2 Benedikt Ahrens, Dan Frumin, Marco Maggesi, Niccolò Veltri, and Niels van der Weide. Bicategories in univalent foundations. *Math. Struct. Comput. Sci.*, 31(10):1232–1269, 2021. doi:10.1017/S0960129522000032.

3 Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent categories and the Rezk completion. *Mathematical Structures in Computer Science*, 25:1010–1039, 2015. doi:10.1017/S0960129514000486.

4 Benedikt Ahrens, Paige Randall North, Michael Shulman, and Dimitris Tsementzis. The Univalence Principle, 2022. To be published in Mem. AMS. arXiv:2102.06275v3.

5 Steve Awodey. Natural models of homotopy type theory. *Math. Struct. Comput. Sci.*, 28(2):241–286, 2018. doi:10.1017/S0960129516000268.

6 John C. Baez and Kenny Courser. Structured cospans. *Theory Appl. Categ.*, 35:Paper No. 48, 1771–1822, 2020.

7 John C. Baez, Kenny Courser, and Christina Vasilakopoulou. Structured versus Decorated Cospans. *Compositionality*, 4, September 2022. doi:10.32408/compositionality-4-3.

8 Nicolas Behr, Russ Harmer, and Jean Krivine. Fundamentals of compositional rewriting theory. *J. Log. Algebraic Methods Program.*, 135:100893, 2023. doi:10.1016/J.JLAMP.2023.100893.

9 Nicolas Behr, Paul-André Melliès, and Noam Zeilberger. Convolution products on double categories and categorification of rule algebras. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction, FSCD 2023, July 3-6, 2023, Rome, Italy*, volume 260 of *LIPIcs*, pages 17:1–17:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.FSCD.2023.17.

10 Georges Blanc. Équivalence naturelle et formules logiques en théorie des catégories. *Arch. Math. Logik Grundlag.*, 19(3-4):131–137, 1978. doi:10.1007/BF02011874.

11 Guillaume Boisseau and Jeremy Gibbons. What you needa know about yoneda: profunctor optics and the yoneda lemma (functional pearl). *Proc. ACM Program. Lang.*, 2(ICFP):84:1–84:27, 2018. doi:10.1145/3236779.

12 Simon Castellan, Pierre Clairambault, and Peter Dybjer. Categories with families: Unityped, simply typed, and dependently typed. In *Joachim Lambek: The Interplay of Mathematics, Logic, and Linguistics*, volume 20 of *Outst. Contrib. Log.*, pages 135–180. Springer, Cham, 2021. doi:10.1007/978-3-030-66545-6_5.

13 Bryce Clarke, Derek Elkins, Jeremy Gibbons, Fosco Loregiàn, Bartosz Milewski, Emily Pillmore, and Mario Román. Profunctor optics, a categorical update. *CoRR*, abs/2001.07488, 2020. arXiv:2001.07488.

14 Pierre-Évariste Dagand and Conor McBride. A categorical treatment of ornaments. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*, pages 530–539. IEEE Computer Society, 2013. doi:10.1109/LICS.2013.60.

15 Fredrik Dahlqvist and Renato Neves. An internal language for categories enriched over generalised metric spaces. In *30th EACSL Annual Conference on Computer Science Logic*, volume 216 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 16, 18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CSL.2022.16.

16 Charles Ehresmann. Catégories structurées. III. Quintettes et applications covariantes. In *Topol. et Géom. Diff. (Sém. C. Ehresmann), Vol. V*, volume Vol. V of *Cahiers du Séminaire dirigé par Charles Ehresmann*, page 21. Inst. Henri Poincaré, Paris, 1963.

17 P. Freyd. Properties invariant within equivalence types of categories. In A. Heller and M. Tierney, editors, *Algebra, Topology and Category Theory: A Collection of Papers in Honor of Samuel Eilenberg*, pages 55–61. Academic Press, New York, 1976.

18 Dennis Gaitsgory and Nick Rozenblyum. *A study in derived algebraic geometry. Vol. II. Deformations, Lie theory and formal geometry*, volume 221 of *Mathematical Surveys and Monographs*. American Mathematical Society, Providence, RI, 2017. doi:10.1090/surv/221.2.

19 R. Gordon, A. J. Power, and Ross Street. Coherence for tricategories. *Mem. Amer. Math. Soc.*, 117(558):vi+81, 1995. doi:10.1090/memo/0558.

20 Marco Grandis and Robert Pare. Limits in double categories. *Cahiers Topologie Géom. Différentielle Catég.*, 40(3):162–220, 1999.

21 Marco Grandis and Robert Pare. Adjoint for double categories. Addenda to: "Limits in double categories" [Cah. Topol. Géom. Différ. Catég. **40** (1999), no. 3, 162–220; mr1716779]. *Cah. Topol. Géom. Différ. Catég.*, 45(3):193–240, 2004.

22 Jason Z. S. Hu and Jacques Carette. Formalizing category theory in agda. In Catalin Hritcu and Andrei Popescu, editors, *CPP '21: 10th ACM SIGPLAN International Conference on Certified Programs and Proofs, Virtual Event, Denmark, January 17-19, 2021*, pages 327–342. ACM, 2021. doi:10.1145/3437992.3439922.

23 G. M. Kelly and Ross Street. Review of the elements of 2-categories. In *Category Seminar (Proc. Sem., Sydney, 1972/1973)*, volume Vol. 420 of *Lecture Notes in Math.*, pages 75–103. Springer, Berlin-New York, 1974.

24 Michael Makkai. First order logic with dependent sorts, with applications to category theory, 1995. URL: http://www.math.mcgill.ca/makkai/folds/foldsinpdf/FOLDS.pdf.

25 Jeffrey C. Morton. Double bicategories and double cospans. *J. Homotopy Relat. Struct.*, 4(1):389–428, 2009.

26 Lyne Moser, Maru Sarazola, and Paula Verdugo. A model structure for weakly horizontally invariant double categories. *Algebr. Geom. Topol.*, 23(4):1725–1786, 2023. doi:10.2140/agt.2023.23.1725.

27 Zach Murray, Dorette Pronk, and Martin Szyld. Implementing double categories in the lean proof assistant, 2022. Talk at Science Atlantic Mathematics, Statistics, and Computer Science Conference, October 15, 2022. URL: https://www.mathstat.dal.ca/~mszyld/Zach_slides.pdf.

28 Max S. New and Daniel R. Licata. Call-by-name gradual type theory. *Log. Methods Comput. Sci.*, 16(1), 2020. doi:10.23638/LMCS-16(1:7)2020.

29 Max S. New and Daniel R. Licata. A formal logic for formal category theory. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures - 26th International Conference, FoSSaCS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2023, Paris, France, April 22-27, 2023, Proceedings*, volume 13992 of *Lecture Notes in Computer Science*, pages 113–134. Springer, 2023. doi:10.1007/978-3-031-30829-1_6.

30 Kimmo I. Rosenthal. Quantaloids, enriched categories and automata theory. *Appl. Categ. Structures*, 3(3):279–301, 1995. doi:10.1007/BF00878445.

31 Michael Shulman. Framed bicategories and monoidal fibrations. *Theory Appl. Categ.*, 20:No. 18, 650–738, 2008.

32 The 1Lab Development Team. The 1Lab. URL: https://1lab.dev.

33 The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

34 Niels van der Weide, Nima Rasekh, Benedikt Ahrens, and Paige Randall North. Univalent double categories. In Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, pages 246–259. ACM, 2024. doi:10.1145/3636501.3636955.

35 Dominic Verity. Enriched categories, internal categories and change of base. *Repr. Theory Appl. Categ.*, 20:1–266, 2011.

36 Vladimir Voevodsky, Benedikt Ahrens, Daniel Grayson, et al. Unimath — a computer-checked library of univalent mathematics. available at http://unimath.org. doi:10.5281/zenodo.13828995.

37 R. J. Wood. Abstract proarrows. I. *Cahiers Topologie Géom. Différentielle*, 23(3):279–290, 1982.

38 R. J. Wood. Proarrows. II. *Cahiers Topologie Géom. Différentielle Catég.*, 26(2):135–168, 1985.

Coslice Colimits in Homotopy Type Theory

Perry Hart
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA

Kuen-Bang Hou (Favonia)
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA

Abstract

We contribute to the theory of (homotopy) colimits inside homotopy type theory. The heart of our work characterizes the connection between colimits in coslices of a universe, called *coslice colimits*, and colimits in the universe (i.e., ordinary colimits). To derive this characterization, we find an explicit construction of colimits in coslices that is tailored to reveal the connection. We use the construction to derive properties of colimits. Notably, we prove that the forgetful functor from a coslice creates colimits over trees. We also use the construction to examine how colimits interact with orthogonal factorization systems and with cohomology theories. As a consequence of their interaction with orthogonal factorization systems, all pointed colimits (special kinds of coslice colimits) preserve n-connectedness, which implies that higher groups are closed under colimits on directed graphs. We have formalized our main construction of the coslice colimit functor in Agda.

2012 ACM Subject Classification Theory of computation → Type theory

Keywords and phrases colimits, homotopy type theory, category theory, higher inductive types, synthetic homotopy theory

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.46

Related Version *Technical report*: `https://doi.org/10.48550/arXiv.2411.15103` [6]

Supplementary Material *Software (Agda Code)*: `https://github.com/PHart3/colimits-agda/tree/v0.1.0` [7]

Funding This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-21-1-0009. 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 Air Force.

Acknowledgements We thank the anonymous reviewers for their feedback that improved the writing of this paper. We also thank the anonymous reviewer for HoTT/UF 2023 who pointed out the relationship between adjunctions and factorization systems.

1 Introduction

Homotopy type theory (HoTT) extends Martin-Löf type theory (MLTT) with univalence and higher inductive types [27]. The key feature of HoTT is that all types behave as homotopy types of topological spaces [9]. Thus, with HoTT, we can use purely type-theoretic methods to prove new properties of spaces. Moreover, higher inductive types (HITs) let us bring a vast range of spaces into HoTT. As a result, HoTT is a useful system for developing synthetic homotopy theory and formalizing it in proof assistants like Coq and Agda [5, 8].

We study HITs arising as *(homotopy) colimits* in coslices of a universe, called *coslice colimits*. Coslices of a universe are type-theoretic versions of coslice categories. A colimit in a category is an object formed by gluing together simpler objects in a coherent fashion. The *coherent* requirement ensures that the colimit has a universal property, which reduces proofs about the colimit to proofs about the simpler objects it is built out of. When these objects are spaces, perhaps endowed with extra structure, colimits built out of them find

wide use in homotopy theory. For example, the class of HITs we study includes colimits of *pointed spaces*. Such colimits are key to the *Brown representability theorem*, which is about homotopy functors on the (∞-)category of pointed connected spaces. Indeed, its proof relies on the fact that this category is generated under colimits by compact cogroups.

1.1 Contributions

In this section, we explain the contributions of the paper along with its organization. We start by outlining the heart of the paper, which we call *the main connection*. Afterward, we describe three independent applications of the main connection in synthetic homotopy theory. Details, proofs, and related additional results are found in our associated technical report [6]. Further, we have formalized in Agda our construction of A-colimits as well as the universality of ordinary colimits for Corollary 21.

1.1.1 The main connection (Section 5)

Suppose $\mathcal{U}$ is a universe and A is a type in $\mathcal{U}$. We want to construct colimits in the *coslice* $A/\mathcal{U}$, or *A-colimits*. The (wild) category $A/\mathcal{U}$ has objects $\sum_{T:\mathcal{U}} A \to T$ and morphisms $X \to_A Y \coloneqq \sum_{k:\mathsf{pr}_1(X)\to\mathsf{pr}_1(Y)} k \circ \mathsf{pr}_2(X) \sim \mathsf{pr}_2(Y)$. Here, $\sim$ is defined by $f_1 \sim f_2 \coloneqq \prod_{x:X} f_1(x) = f_2(x)$ for any dependent functions $f_1, f_2 : \prod_{x:X} Y(x)$, called the type of *homotopies* from f_1 to f_2.

HoTT has a general schema for HITs that would let us simply postulate A-colimits. We, however, explicitly construct A-colimits with just the machinery of MLTT augmented with pushouts (Section 5.3).[1] We take this different approach to reveal the connection between A-colimits and their underlying colimits in $\mathcal{U}$. In fact, our construction is *not* a case of a general method to encode higher-dimensional HITs with pushouts but rather tailored to reveal this connection.

Why do we care about this connection? It sheds light on three established areas of synthetic homotopy theory. We preview them now and return to them in Sections 6–8.

The universality of colimits (Section 6)

The *universality* of colimits is a special feature of locally cartesian closed (LCC) ∞-categories, such as that of spaces. The main connection will establish a well-known classical result inside type theory: The forgetful functor $A/\mathcal{U} \to \mathcal{U}$ *creates* colimits of diagrams over contractible graphs (Theorem 18), which make up a large subclass of graphs.[2] Examples of such colimits include sequential colimits [25]. With the forgetful functor creating colimits, we can transfer universality of ordinary colimits to A-colimits over contractible graphs (Corollary 21). This is notable as LCC ∞-categories are not closed under coslices.

The categories of higher groups are cocomplete (Section 7)

A striking feature of colimits is their interaction with (orthogonal) factorization systems. In Section 7, we use the main connection to show that colimits in $A/\mathcal{U}$ preserve left classes of maps of factorization systems on $\mathcal{U}$. It is significant that we consider factorization systems on $\mathcal{U}$ rather than $A/\mathcal{U}$. We could derive a similar preservation theorem for systems on $A/\mathcal{U}$ directly from the universal property of an A-colimit. In practice, however, the factorization

[1] A theoretical advantage of such a construction is that pushouts, the simplest nontrivial HITs, can be postulated with a less powerful schema than that required to postulate A-colimits.

[2] For a definition of *creating (co)limits*, see [16].

systems we tend to care about are on $\mathcal{U}$. Since the main connection relates the action of A-colimits on maps to the action of the underlying ordinary colimits on maps (Section 5.4), we manage to deduce the preservation theorem for systems on $\mathcal{U}$.

To prove this theorem, we find it useful to develop the theory of factorization systems in a more general setting than $\mathcal{U}$. In Section 4.1, we study such systems on *wild categories*, which make up one approach to category theory in HoTT. We prove that if a functor F of well-behaved wild categories with factorization systems has a right adjoint G, then F preserves the left class when G preserves the right class (Theorem 13). We combine this result with the main connection to deduce the desired preservation property.

When we focus on the (n-connected, n-truncated) factorization system on $\mathcal{U}$ [27, Chapter 7.6] and take A as the unit type, the main connection shows that the colimit of every diagram of pointed n-connected types is n-connected. One useful corollary of this is that the higher category $(n, k)\,\mathsf{GType}$ of k-tuply groupal n-groupoids considered by [2] is cocomplete on (directed) graphs for all truncation levels $-2 \leq n \leq \infty$ and $-1 \leq k < \infty$ (Example 23).

Cohomology sends colimits to weak limits (Section 8)

Finally, we examine how colimits interact with cohomology theories, which are important algebraic invariants of spaces. To do so, we consider *weak limits*, which are key ingredients in the Brown representability theorem (BRT). A weak colimit in a category need not satify the uniqueness property required of a colimit. The BRT specifies conditions for a presheaf on the homotopy category $\mathbf{Ho}(\mathbf{Top}_{*,c})$ of pointed connected spaces to be representable. The standard proof of the BRT requires the presheaf to send countable homotopy colimits in $\mathbf{Top}_{*,c}$ to weak limits in $\mathbf{Set}$ [14, Section 1.4.1]. Eilenberg-Steenrod cohomology theories enjoy this property as set-valued functors.

In Section 8, we use the main connection to establish a restricted, type-theoretic version of this property. From the main connection we derive another construction of A-colimits, as pushouts of coproducts (Corollary 26), which mirrors a well-known classical lemma. We take A as the unit type and combine the new construction with the Mayer-Vietoris sequence to find that cohomology takes finite colimits to weak limits assuming the axiom of choice.

2 Additional related work

2.1 Construction of nonrecursive 2-HITs

The HITs we consider are nonrecursive 2-HITs, in the sense that they have only nonrecursive constructors of points and of paths of dimension one or two. Van Doorn et al. explicitly construct nonrecursive 2-HITs in MLTT augmented with pushouts [28, Section 5]. When specialized to A-colimits, however, their construction has a significantly different form from ours and does not directly lead to the properties of A-colimits we derive. Moreover, they do not prove the full induction principle enjoyed by the 2-HIT for their construction, whereas we do for ours. The full induction principle is necessary (and sufficient) to characterize the 2-HIT uniquely.

2.2 Orthogonal factorization systems

Our work also builds on the theory of factorization systems. Such systems play important roles in model category theory [20], a key framework for classical homotopy theory. Moreover, in type theory, Rijke et al. have shown that factorization systems on $\mathcal{U}$ are closely connected

to modalities [22], which are important in logic. We extend factorization systems to wild categories other than $\mathcal{U}$. Moreover, we lift factorization systems on $\mathcal{U}$ to wild categories of $\mathcal{U}$-valued diagrams (Lemma 22).

3 Background on type theory and colimits

Before describing the main connection and its applications, we need to review the type system we work in. For us, the most important data type of this system is the *ordinary colimit*, i.e., the colimit of a diagram of types over a graph. We define this type in Section 3.3 and call it "ordinary" to distinguish it from the notion of *coslice colimit*. The latter takes place in coslices of a universe rather than the universe itself, and we will construct coslice colimits out of ordinary colimits.

3.1 Type system

We assume the reader is familiar with MLTT and HITs in the style of [27]. We will work in MLTT augmented with ordinary colimits and denote this system by $\mathsf{MLTT} + \mathsf{Colim}$. In fact, we need only augment MLTT with pushouts as they let us construct all nonrecursive 1-HITs, including ordinary colimits, with all of their computational properties. Notably, $\mathsf{MLTT} + \mathsf{Colim}$ comes with strong function extensionality for free. This property is critical for reasoning about functions in type theory and underlies our entire development (see, for example, Lemma 8). Overall, we carry out our proofs inside $\mathsf{MLTT} + \mathsf{Colim}$ until Section 7. For Section 7 and Section 8, we also assume Voevodsky's univalence axiom.

Before reviewing ordinary colimits, we recall two essential constructions in our type system. The first is the function $\mathsf{ap} : (x = y) \to (f(x) = f(y))$ defined by path induction for all functions $f : X \to Y$ and $x, y : X$. (We use $=$ for the identity type and $\equiv$ for definitional equality.) If we view X as an ∞-groupoid, then ap is the action of f on morphisms of X (thereby exhibiting f as a functor). The second is the *transport* function $\mathsf{transp}^Y : \prod_{x,y:X} \prod_{p:x=y} Y(x) \to Y(y)$ for any type family Y over X. This notion also gives us a dependent version of ap: If $f : \prod_{x:X} Y(x)$, then we have a function $\mathsf{apd}_f : \prod_{x,y:X} \prod_{p:x=y} \mathsf{transp}^Y(p, f(x)) = f(y)$. The transport function is essential for stating the induction principle for HITs, e.g., the colimits in Section 3.3. It also satisfies the following coherence law, which we need for our construction of A-colimits.

▶ **Lemma 1.** *Let $f, g : X \to Y$. For all $x, y : X$, $p : x = y$, and $H : f \sim g$, we have a commuting square of identities*

$$\begin{array}{ccc} f(x) & \xRightarrow{H(x)} & g(x) \\ {\scriptstyle \mathsf{ap}_f(p)}\Big\Downarrow & & \Big\Downarrow{\scriptstyle \mathsf{ap}_g(p)} \\ f(y) & \xRightarrow[\mathsf{transp}^{z \mapsto f(z) = g(z)}(p, H(x))]{} & g(y) \end{array}$$

Finally, a remark on notation: we may use the Agda notation $(x : X) \to Y(x)$ for the type $\prod_{x:X} Y(x)$ for any type family Y over X.

3.2 Graphs

Let $\mathcal{U}$ be a universe and $A : \mathcal{U}$. In classical 1-category theory, a diagram in a category $\mathcal{C}$ is a functor $F : \mathcal{I} \to \mathcal{C}$, where $\mathcal{I}$ is the shape of the diagram. As long as $\mathcal{C}$ is cocomplete, we can form the functor $\mathsf{colim}_{\mathcal{I}}$ sending each diagram over $\mathcal{I}$ to its colimit in $\mathcal{C}$. We, however, want the colimit of a diagram in the ∞-category $A/\mathcal{U}$. This requires the diagram to be an ∞-functor: the functor laws must satisfy coherence laws up to homotopy, which themselves must satisfy higher coherence laws, and so on at arbitrarily high levels. It is unknown whether such ∞-functors are definable in HoTT.

To avoid infinite coherence conditions, we specialize $\mathcal{I}$ to the free category generated by a *graph*. A graph Γ is a pair (Γ_0, Γ_1) consisting of a type $\Gamma_0 : \mathcal{U}$ of vertices and a family $\Gamma_1 : \Gamma_0 \to \Gamma_0 \to \mathcal{U}$ of edges. A Γ-*shaped diagram* F *in* $A/\mathcal{U}$ is a pair (F_0, F_1) consisting of a function $F_0 : \Gamma_0 \to A/\mathcal{U}$ and a family of maps $F_1 : (i, j : \Gamma_0) \to \Gamma_1(i, j) \to F_0(i) \to_A F_0(j)$. We may write F for F_0 and F_1. The induced diagram in $A/\mathcal{U}$ satisfies all the infinite coherence laws because its domain is freely generated by the points and edges of Γ.

▶ **Example 2.** For each graph Γ and $D : \mathsf{Ob}(A/\mathcal{U})$, the *constant diagram* $\mathsf{const}_\Gamma(D)$ *at* D is defined by $(\mathsf{const}_\Gamma(D))_0(i) :\equiv D$ and $(\mathsf{const}_\Gamma(D))_1(i, j, g) :\equiv \mathsf{id}_D$. We often refer to $\mathsf{const}_\Gamma(D)$ simply by D.

We will see that A-colimits interact nicely with *trees*. A tree is a graph without non-directed cycles. Formally, a graph Γ is a *tree* if the quotient Γ_0/Γ_1 is contractible. Both $\mathbb{N}$ and $\mathbb{Z}$ are trees when equipped with the successor ordering:

$$\mathbb{N} \equiv 0 \longrightarrow 1 \longrightarrow 2 \longrightarrow \cdots \qquad \mathbb{Z} \equiv \cdots \longrightarrow -1 \longrightarrow 0 \longrightarrow 1 \longrightarrow \cdots$$

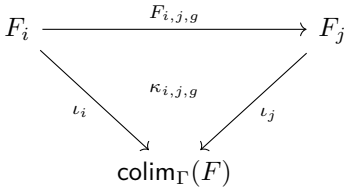
Another example of a tree is a span $\bullet \leftarrow \bullet \rightarrow \bullet$, on which pushouts are defined.

Rijke has defined the notion of *directed tree* and has defined an interpretation function sending an element of a W-type to a directed tree [23, The underlying trees of elements of W-types]. Intuitively, a directed tree is a rooted graph such that for each vertex v, there is a single directed path (including the trivial path) from v to the root. Every directed tree is a tree in our sense [6, Corollary 4.0.6]. Thus, elements of a W-type can be realized as trees.

3.3 Colimits in $\mathcal{U}$

Let F be a Γ-shaped diagram in $\mathcal{U}$. The *(ordinary) colimit of* F is the HIT $\mathsf{colim}_\Gamma(F)$ generated by

$$\begin{aligned} \iota &: (i : \Gamma_0) \to F_i \to \mathsf{colim}_\Gamma(F) \\ \kappa &: (i, j : \Gamma_0)\,(g : \Gamma_1(i, j)) \to \iota_j \circ F_{i,j,g} \sim \iota_i \end{aligned}$$



These two constructors make $\mathsf{colim}_\Gamma(F)$ a *cocone under* F (or F-*cocone*): a type C equipped with maps $u : \prod_{i:\Gamma_0} F_i \to C$ and homotopies $K : \prod_{i,j,g} u_j \circ F_{i,j,g} \sim u_i$. What characterizes $\mathsf{colim}_\Gamma(F)$ as a colimit of F is that κ is a *(homotopy) initial* F-cocone [24]. Equivalently, for every $X : \mathcal{U}$, the function

$$\begin{aligned} &\mathsf{postcomp} : (\mathsf{colim}_\Gamma(F) \to X) \to \mathsf{Cocone}_F(X) \\ &\mathsf{postcomp}(f) :\equiv \big(\lambda i.f \circ \iota_i, \lambda i \lambda j \lambda g \lambda(x : F_i).\, \mathsf{ap}_f(\kappa_{i,j,g}(x))\big) \end{aligned}$$

is an equivalence, where $\mathsf{Cocone}_F(X)$ denotes the type of F-cocones on X.

Our proof of Theorem 15 will use the induction principle for $\mathsf{colim}_\Gamma(F)$. This states that for every type family E over $\mathsf{colim}_\Gamma(F)$ together with data

$$r \;:\; \prod_{i:\Gamma_0}\prod_{x:F_i} E(\iota_i(x)) \qquad R \;:\; \prod_{i,j:\Gamma_0}\prod_{g:\Gamma_1(i,j)}\prod_{x:F_i} \mathsf{transp}^E(\kappa_{i,j,g}(x), r(j, F_{i,j,g}(x))) = r(i,x)$$

we have a function $\mathsf{ind}(E,r,R) : \prod_{z:\mathsf{colim}_\Gamma(F)} E(z)$ that satisfies $\mathsf{ind}(E,r,R)(\iota_i(x)) \equiv r(i,x)$ and is equipped with an identity $\rho_{\mathsf{ind}(E,r,R)}(i,j,g,x) : \mathsf{apd}_{\mathsf{ind}(E,r,R)}(\kappa_{i,j,g}(x)) = R(i,j,g,x)$.

4 Wild categories

Any universe, along with its coslices, fits into the framework of *wild categories*. This is one approach to category theory in HoTT and is used by other works of synthetic homotopy theory [3, 5, 11]. It is useful for the relationship between A-colimits and factorization systems we establish in Section 7. This requires us to formulate factorization systems on categories other than universes, namely the category of type-valued diagrams over a graph.

The key distinction between wild categories and (pre-)categories [27, Chapter 9.1] is that the latter have 0-truncated hom types. This means that instead of trivializing the higher coherence data for morphisms, wild categories simply ignore them. We choose them over pre-categories because we will focus on universes and their coslices (see Example 7), which are wild categories but not pre-categories.

▶ **Definition 3** ([6, Definition 3.1.1])**.** *A* wild category (in a universe $\mathcal{U}$) *is a tuple consisting of a type* $\mathsf{Ob} : \mathcal{U}$ *of objects, a family* $\mathsf{hom} : \mathsf{Ob} \to \mathsf{Ob} \to \mathcal{U}$ *of hom types, identity morphisms* id*, composition* $\circ$*, left* LId *and right* RId *unit laws for* $\circ$*, and associativity laws* assoc *for* $\circ$*.*

By itself, the data of a wild category is insufficient for our work on factorization systems. We need two extra ingredients. The first is the notion of a wild bicategory. The second is a wild-categorical version of univalence.

▶ **Definition 4.** *A wild category* $\mathcal{C}$ *is a* (wild) bicategory *if it is equipped with identities*

(a) $\mathsf{ap}_{-\circ f}(\mathsf{assoc}(k,g,h)) \cdot \mathsf{assoc}(k, g\circ h, f) \cdot \mathsf{ap}_{k\circ -}(\mathsf{assoc}(g,h,k)) = \mathsf{assoc}(k\circ g, h, f) \cdot \mathsf{assoc}(k,g,h\circ f)$ *for all composable morphisms* k*,* g*,* h*, and* f

(b) $\mathsf{assoc}(g, \mathsf{id}, h) \cdot \mathsf{ap}_{g\circ -}(\mathsf{LId}(h)) = \mathsf{ap}_{-\circ h}(\mathsf{RId}(g))$ *for all composable morphisms* g *and* h*.*[3]

▶ Remark. For us, a bicategory is always a wild $(2,1)$-category since the 2-cells, which are identities in $\mathcal{U}$, are invertible.

Before moving to univalence, we transfer a well-known lemma of classical 2-category theory to type theory. This was first proved for monoidal categories [10], but the proof is applicable to all bicategories. (The type-theoretic version also appears as [3, Lemma 4.3].)

▶ **Lemma 5** ([6, Lemma 3.1.3])**.** *Let* $\mathcal{C}$ *be a bicategory. For all* $A, B, C : \mathsf{Ob}(\mathcal{C})$*,* $f : \mathsf{hom}_{\mathcal{C}}(A,B)$*,* $g : \mathsf{hom}_{\mathcal{C}}(B,C)$*, we have* $\mathsf{LId}(g\circ f)^{-1} \cdot \mathsf{assoc}(\mathsf{id}, g, f)^{-1} \cdot \mathsf{ap}_{-\circ f}(\mathsf{LId}(g)) = \mathsf{refl}_{g\circ f}$*.*

▶ **Definition 6.** *We say that a wild category* $\mathcal{C}$ *is* univalent *if the canonical function* $(A =_{\mathsf{Ob}(\mathcal{C})} B) \to (A \simeq_{\mathcal{C}} B)$ *is an equivalence. Here, elements of the righthand type are equivalences, defined as bi-invertible morphisms (in the manner of [27, Definition 4.3.1]).*

[3] A wild bicategory is called a *wild* 2*-precategory* by [3].

▶ **Example 7.** The following are univalent bicategories assuming the univalence axiom.

- The category $\mathcal{U}$ of types and functions
- For each $A : \mathcal{U}$, the coslice $A/\mathcal{U}$ of $\mathcal{U}$ under A
- The category $\mathsf{Diag}(\Gamma, A/\mathcal{U})$ of Γ-shaped diagrams in $A/\mathcal{U}$. We define its hom types (natural transformations) when we present the action of the A-colimit on maps (Section 5.4).

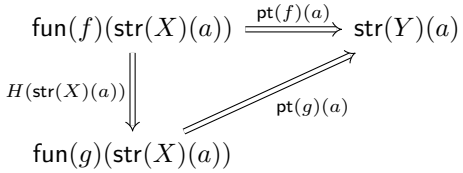
Our ultimate interest is in colimits in the wild category $A/\mathcal{U}$. This category is defined by

$$\mathsf{Ob}(A/\mathcal{U}) \coloneqq \textstyle\sum_{X:\mathcal{U}} A \to X \qquad \mathsf{hom}_{A/\mathcal{U}}(X, Y) \coloneqq X \to_A Y$$

For each $X : \mathsf{Ob}(A/\mathcal{U})$, the identity morphism on X is $\big(\mathsf{id}_{\mathsf{pr}_1(X)}, \lambda a.\mathsf{refl}_{\mathsf{pr}_2(X)(a)}\big)$. Composition is defined by $(g, g_p) \circ (f, f_p) \coloneqq \big(g \circ f, \lambda a.\, \mathsf{ap}_g(f_p(a)) \cdot g_p(a)\big)$. The associativity and unit laws follow from routine path algebra. Note that the categories $\mathbf{0}/\mathcal{U}$ and $\mathcal{U}$ are equivalent.

We write ty and str for the functions $\mathsf{pr}_1 : \mathsf{Ob}(A/\mathcal{U}) \to \mathcal{U}$ and $\mathsf{pr}_2 : (Z : \mathsf{Ob}(A/\mathcal{U})) \to A \to \mathsf{pr}_1(Z)$, i.e., the underlying type and structure map of an object in $A/\mathcal{U}$, respectively. Also, we write fun and pt for the functions $\mathsf{pr}_1 : \mathsf{hom}_{A/\mathcal{U}}(W, Z) \to \mathsf{ty}(W) \to \mathsf{ty}(Z)$ and $\mathsf{pr}_2 : \big(h : \mathsf{hom}_{A/\mathcal{U}}(W, Z)\big) \to \mathsf{pr}_1(h) \circ \mathsf{str}(W) \sim \mathsf{str}(Z)$, respectively.

▶ **Lemma 8.** *Let $f, g : X \to_A Y$. Define $f \sim_A g$ as the type of homotopies $H : \mathsf{fun}(f) \sim \mathsf{fun}(g)$ equipped with a commuting triangle*



for each $a : A$. The canonical function $f = g \to f \sim_A g$ is an equivalence, with inverse denoted by $\langle -, - \rangle : f \sim_A g \to f = g$.

Elements of $f \sim_A g$ are called *A-homotopies* between f and g.

4.1 Orthogonal factorization systems

We now introduce (orthogonal) factorization systems on wild categories. For us, the key property of such systems is that they interact nicely with adjunctions. In Section 7, we deduce from this property, combined with the main connection, that A-colimits preserve the left classes of factorization systems on $\mathcal{U}$.

▶ **Definition 9.** *Let $\mathcal{C}$ be a wild category. An* orthogonal factorization system (OFS) *on $\mathcal{C}$ consists of predicates $\mathcal{L}, \mathcal{R} : \prod_{A,B:\mathcal{C}} \mathsf{hom}_{\mathcal{C}}(A, B) \to \mathsf{Prop}$ such that*

1. *both $\mathcal{L}$ and $\mathcal{R}$ are closed under composition and have all identities*
2. *for every $h : \mathsf{hom}_{\mathcal{C}}(A, B)$, the following type is contractible:*

$$\mathsf{fact}_{\mathcal{L},\mathcal{R}}(h) \coloneqq \sum_{D:\mathsf{Ob}(\mathcal{C})} \sum_{f:\mathsf{hom}_{\mathcal{C}}(A,D)} \sum_{g:\mathsf{hom}_{\mathcal{C}}(D,B)} (g \circ f = h) \times \mathcal{L}(f) \times \mathcal{R}(g)$$

For the next lemma, where $\mathcal{C}$ is a univalent bicategory, $\mathcal{C}$ is similar enough to $\mathcal{U}$ that the proof of the lemma for $\mathcal{U}$ can be transferred to $\mathcal{C}$.[4] Indeed, univalence lets us characterize the identity types of $\mathsf{fact}_{\mathcal{L},\mathcal{R}}(h)$ via the fundamental theorem of identity types [21, Theorem 11.2.2]. Moreover, Lemma 5 gives us a suitable diagonal filler for the key commuting square used by the proof. Before stating the next lemma, we need a definition.

[4] For the proof of this lemma for $\mathcal{U}$, see [22, Lemma 1.46].

▶ **Definition 10.** *Let $\mathcal{C}$ be a wild category. Let $l : \mathsf{hom}_{\mathcal{C}}(A, B)$ and $\mathcal{H}$ be a property of morphisms in $\mathcal{C}$. We say that l has the* left lifting property against $\mathcal{H}$ *if for every $r : \mathsf{hom}_{\mathcal{C}}(C, D)$ such that $\mathcal{H}(r)$ and every commuting square*

$$\begin{array}{ccc} A & \xrightarrow{f} & C \\ {\scriptstyle l}\downarrow & S & \downarrow{\scriptstyle r} \\ B & \xrightarrow[g]{} & D \end{array}$$

the type of diagonal fillers

$$\sum_{d:\mathsf{hom}_{\mathcal{C}}(B,C)} \sum_{H_f : d \circ l = f} \sum_{H_g : r \circ d = g} \mathsf{assoc}(r, d, l) \cdot \mathsf{ap}_{r \circ -}(H_f) = \mathsf{ap}_{- \circ l}(H_g) \cdot S$$

is contractible.

▶ **Lemma 11** ([6, Corollary 3.3.6])**.** *Suppose that $\mathcal{C}$ is a univalent bicategory with an OFS $(\mathcal{L}, \mathcal{R})$. A map is in $\mathcal{L}$ if and only if it has the left lifting property against $\mathcal{R}$.*

This alternative definition of $\mathcal{L}$ is useful for the proof of Theorem 13, below. For this theorem, we need to introduce adjoint pairs of functors between wild categories.

▶ **Definition 12.** *Let $L : \mathcal{C} \to \mathcal{D}$ and $R : \mathcal{D} \to \mathcal{C}$ be functors of wild categories. An* adjunction *$L \dashv R$ consists of an equivalence $\alpha : \mathsf{hom}_{\mathcal{D}}(LA, X) \simeq \mathsf{hom}_{\mathcal{C}}(A, RX)$ for all $A : \mathsf{Ob}(\mathcal{C})$ and $X : \mathsf{Ob}(\mathcal{D})$ along with naturality proofs:*

$$\mathsf{n}_1 \ : \prod_{A:\mathsf{Ob}(\mathcal{C})} \prod_{X,Y:\mathsf{Ob}(\mathcal{D})} \prod_{g:\mathsf{hom}_{\mathcal{D}}(X,Y)} \prod_{h:\mathsf{hom}_{\mathcal{D}}(LA,X)} R(g) \circ \alpha(h) = \alpha(g \circ h)$$

$$\mathsf{n}_2 \ : \prod_{Y:\mathsf{Ob}(\mathcal{D})} \prod_{A,B:\mathsf{Ob}(\mathcal{C})} \prod_{f:\mathsf{hom}_{\mathcal{C}}(A,B)} \prod_{h:\mathsf{hom}_{\mathcal{D}}(LB,Y)} \alpha(h) \circ f = \alpha(h \circ L(f))$$

▶ **Theorem 13** ([6, Corollary 3.3.9])**.** *Consider an adjunction $L \dashv R$ where both $\mathcal{C}$ and $\mathcal{D}$ are univalent bicategories. If R preserves $\mathcal{R}$, then L preserves $\mathcal{L}$.*

5 The main connection

Let $\mathcal{U}$ be a universe. Let Γ be a graph and suppose F is a diagram in $A/\mathcal{U}$ over Γ. Working in $\mathsf{MLTT} + \mathsf{Colim}$, we want to construct the *A-colimit* of F so as to show the connection between A-colimits and ordinary colimits. After defining *A-colimit*, we mention a reasonable yet wrong approach to constructing it. Then, we explain another construction and prove it is correct by exhibiting it as left adjoint to the constant diagram functor. The Agda proof of this adjunction is found in the folder [7, `Colimit-code/Main-Theorem`].

5.1 Definition of A-colimits

We can generalize ordinary colimits in Section 3 to all coslices $A/\mathcal{U}$. For each $Y : \mathsf{Ob}(A/\mathcal{U})$, an F-cocone on Y consists of a family of maps $h : (i : \Gamma_0) \to F_i \to_A Y$ in $A/\mathcal{U}$ together with an identity $H_{i,j,g} \ : \ h_j \circ F_{i,j,g} = h_i$ for all $i, j : \Gamma_0$ and $g : \Gamma_1(i, j)$. In this situation, we say that Y is a *colimit of F* if (h, H) is initial in the category of F-cocones. This means that for each $X : \mathsf{Ob}(A/\mathcal{U})$, the function

$$\mathsf{postcomp}(h, H) \ : \ (Y \to_A X) \ \to \ \mathsf{Cocone}_F(X)$$
$$\mathsf{postcomp}(h, H, f) \ := \ \big(\lambda i.f \circ h_i, \lambda i \lambda j \lambda g.\mathsf{assoc}(f, h_j, F_{i,j,g}) \cdot \mathsf{ap}_{f \circ -}(H_{i,j,g})\big)$$

is an equivalence. We must include the associativity term since associativity of maps does not hold judgmentally in $A/\mathcal{U}$ (whereas it does in $\mathcal{U}$).

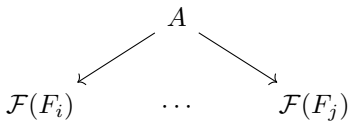
Observe that by a variant of Lemma 8, $h_j \circ F_{i,j,g} = h_i$ is equivalent to the type of homotopies $\eta_{i,j,g} : \mathsf{fun}(h_j) \circ \mathsf{fun}(F_{i,j,g}) \sim \mathsf{fun}(h_i)$ equipped with a commuting square

$$\begin{array}{ccc} \mathsf{fun}(h_j)(\mathsf{fun}(F_{i,j,g})(\mathsf{str}(F_i)(a))) & \xRightarrow{\eta_{i,j,g}(\mathsf{str}(F_i)(a))} & \mathsf{fun}(h_i)(\mathsf{str}(F_i)(a)) \\ \Big\Downarrow{\scriptstyle \mathsf{ap}_{\mathsf{fun}(h_j)}(\mathsf{pt}(F_{i,j,g})(a))} & & \Big\Downarrow{\scriptstyle \mathsf{pt}(h_i)(a)} \\ \mathsf{fun}(h_j)(\mathsf{str}(F_j)(a)) & \xRightarrow[\mathsf{pt}(h_j)(a)]{} & \mathsf{str}(Y)(a) \end{array} \tag{2-c}$$

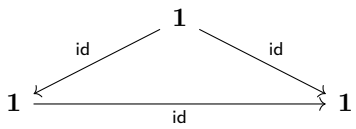
of paths for each $a : A$. It is this family of 2-cells which distinguishes the colimit of F, in $A/\mathcal{U}$, from $\mathsf{colim}_\Gamma(\mathcal{F}(F))$. Here, we reuse $\mathcal{F}$ to denote the evident forgetful functor from Γ-shaped diagrams in $A/\mathcal{U}$ to those in $\mathcal{U}$. The 2-cells affect $\mathsf{colim}_\Gamma(\mathcal{F}(F))$ by collapsing its nontrivial loops formed by paths of the form $\eta(\mathsf{str}(F_i)(a))$. We call such loops *distinguished loops* in $\mathsf{colim}_\Gamma(\mathcal{F}(F))$. For example, if $i \equiv j$ and $F_{i,j,g} \equiv \mathsf{id}_{F_i}$, then (2-c) is equivalent to $\eta(\mathsf{str}(F_i)(a)) = \mathsf{refl}_{\mathsf{fun}(h_i)(\mathsf{str}(F_i)(a))}$. In this case, it fills the loop $\eta(\mathsf{str}(F_i)(a))$.

5.2 Misleading approach

If our setting behaved like the classical one, the colimit of F in $A/\mathcal{U}$ would arise as the ordinary colimit of the *augmented diagram*: $\mathcal{F}(F)$ augmented with the canonical arrow from A to $\mathcal{F}(F_i)$ for each $i : \Gamma_0$ [17, Proposition 4.6]. If Γ is *discrete*, i.e., Γ_1 is the empty relation, then the A-colimit of F inside HoTT is, in fact, the colimit of



In general, though, this construction is wrong inside HoTT. For example, the pointed colimit of the diagram $\mathbf{1} \xrightarrow{\mathsf{id}} \mathbf{1}$ is trivial, but the colimit of the augmented diagram

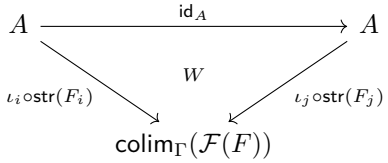


is the circle S^1. The reason for the discrepancy between the classical case and ours is that unless Γ is discrete, the augmented diagram inside HoTT adds arrows that are intended as composites but are not interpreted as such in the model of HoTT. Rather, the model sees them as freely added to the diagram.

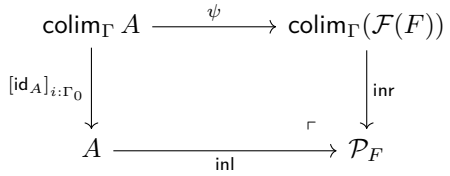
5.3 Our approach

Our approach to building the colimit of F never creates an augmented diagram, thereby avoiding the problem of Section 5.2. We start with the ordinary colimit $\mathsf{colim}_\Gamma(\mathcal{F}(F))$ which ignores the coslice structure of F. Then, we glue onto this colimit the 2-cells required by the coslice colimit. We do this via a quotient of $\mathsf{colim}_\Gamma(\mathcal{F}(F))$ that fills its distinguished loops.

To this end, define $\mathsf{colim}_\Gamma A \xrightarrow{\psi} \mathsf{colim}_\Gamma(\mathcal{F}(F))$ by colimit induction, as the function induced by the cocone



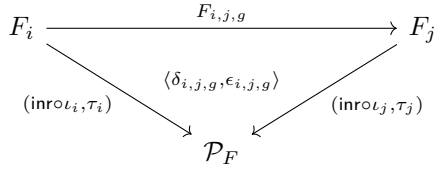
under the constant diagram at A. The homotopy $W : \iota_j \circ \mathsf{str}(F_j) \sim \iota_i \circ \mathsf{str}(F_i)$ is defined by $W(a) \coloneqq \mathsf{ap}_{\iota_j}(\mathsf{pt}(F_{i,j,g})(a))^{-1} \cdot \kappa_{i,j,g}(\mathsf{str}(F_i)(a))$. Intutively, ψ finds the distinguished loops of $\mathsf{colim}_\Gamma(\mathcal{F}(F))$. Next, form the pushout square



which, by the definition of pushout types, comes with a homotopy $\mathsf{glue}_{\mathcal{P}_F} : \mathsf{inl} \circ [\mathsf{id}_A] \sim \mathsf{inr} \circ \psi$. This pushout is our approach to forming the desired quotient of $\mathsf{colim}_\Gamma(\mathcal{F}(F))$.

▶ **Example 14.** Suppose that Γ has a single vertex v and a single loop ℓ at v. Let $\mathbf{2}$ denote the type of booleans. Define the pointed diagram F over Γ by $F_v \coloneqq (\mathbf{2}, \mathsf{true})$ and $F_{v,v,\ell} \coloneqq (\mathsf{id}_{\mathbf{2}}, \mathsf{refl})$. Then $\mathsf{colim}_\Gamma(\mathcal{F}(F)) \simeq S^1 + S^1$. In this case, the function ψ traces the left copy of S^1, the distinguished loop, exactly once. The pushout $\mathcal{P}_F$ is formed from $S^1 + S^1$ by filling this loop, which collapses the left copy of S^1 to a point. As a result, $\mathcal{P}_F \simeq \mathbf{1} + S^1$.

Back to the general case, with the equivalence $\langle -, - \rangle$ of Lemma 8, we can form an F-cocone on $(\mathcal{P}_F, \mathsf{inl})$



$(\tau_i(a) \coloneqq \mathsf{glue}_{\mathcal{P}_F}(\iota_i(a))^{-1})$

as follows. We have a homotopy $\delta_{i,j,g} \coloneqq \lambda(x : \mathsf{ty}(F_i)).\, \mathsf{ap}_{\mathsf{inr}}(\kappa_{i,j,g}(x))$ from $\mathsf{inr} \circ \iota_j \circ \mathsf{fun}(F_{i,j,g})$ to $\mathsf{inr} \circ \iota_i$. Further, for each $a : A$, we have a chain $\epsilon_{i,j,g}(a)$ of identities

$$
\begin{aligned}
& \mathsf{ap}_{\mathsf{inr}}(\kappa_{i,j,g}(\mathsf{str}(F_i)(a)))^{-1} \cdot \mathsf{ap}_{\mathsf{inr}\circ\iota_j}(\mathsf{pt}(F_{i,j,g})(a)) \cdot \tau_j(a) \\
=\ & \mathsf{ap}_{\mathsf{inr}}(\mathsf{ap}_{\iota_j}(\mathsf{pt}(F_{i,j,g})(a))^{-1} \cdot \kappa_{i,j,g}(\mathsf{str}(F_i)(a)))^{-1} \cdot \tau_j(a) \cdot \mathsf{refl}_{\mathsf{inl}(a)} \\
=\ & \mathsf{ap}_{\mathsf{inr}}(\mathsf{ap}_{\psi}(\kappa_{i,j,g}(a)))^{-1} \cdot \tau_j(a) \cdot \mathsf{refl}_{\mathsf{inl}(a)} && (\textit{via } \rho_\psi(i,j,g,a)) \\
=\ & \mathsf{ap}_{\mathsf{inr}}(\mathsf{ap}_{\psi}(\kappa_{i,j,g}(a)))^{-1} \cdot \tau_j(a) \cdot \mathsf{ap}_{\mathsf{inl}}(\mathsf{ap}_{[\mathsf{id}_A]}(\kappa_{i,j,g}(a))) && (\textit{via } \rho_{[\mathsf{id}_A]}(i,j,g,a)) \\
=\ & \mathsf{transp}^{\mathsf{inr}\circ\psi\sim\mathsf{inl}\circ[\mathsf{id}_A]}(\kappa_{i,j,g}(a), \tau_j(a)) && (\textit{Lemma } 1) \\
=\ & \tau_i(a) && (\textit{by } \mathsf{apd}_{\mathsf{glue}(-)^{-1}}(\kappa_{i,j,g}(a)))
\end{aligned}
$$

Let $\mathcal{K}(\mathcal{P}_F)$ denote this F-cocone structure on $(\mathcal{P}_F, \mathsf{inl})$.

▶ **Theorem 15** ([6, Theorem 5.4.3])**.** *The function*

$$\mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), T, f_T) \ :\ ((\mathcal{P}_F, \mathsf{inl}) \to_A (T, f_T)) \ \to\ \mathsf{Cocone}_F(T, f_T)$$

is an equivalence for every $(T, f_T) : \mathsf{Ob}(A/\mathcal{U})$.

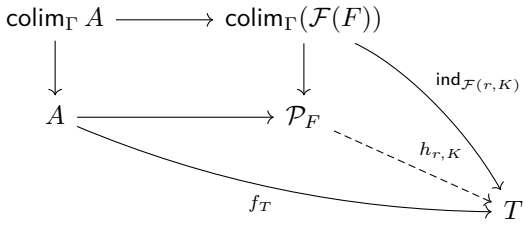
Proof. We construct an inverse $\mathsf{Cocone}_F(T, f_T) \xrightarrow{\Theta} ((\mathcal{P}_F, \mathsf{inl}) \to_A (T, f_T))$ of $\mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), T, f_T)$ as follows. Let $(r, K) : \mathsf{Cocone}_F(T, f_T)$. The forgetful functor $\mathcal{F}$ from cocones under F to ordinary cocones under $\mathcal{F}(F)$ gives rise to the function $\mathsf{ind}_{\mathcal{F}(r,K)} : \mathsf{colim}_\Gamma(\mathcal{F}(F)) \to T$ by colimit induction. For all $i : \Gamma_0$ and $a : A$, we have

$$f_T(a) \overset{\mathsf{pt}(r_i)(a)^{-1}}{=\!=} \mathsf{fun}(r_i)(\mathsf{str}(F_i(a))) \equiv\!\equiv \mathsf{ind}_{\mathcal{F}(r,K)}(\mathsf{str}(F_i(a)))$$

Further, for all $i, j : \Gamma_0$, $g : \Gamma_1(i, j)$, and $a : A$, we have a chain of identities

$$\begin{aligned}
&\mathsf{transp}^{f_T \circ [\mathsf{id}_A] = \mathsf{ind}_{\mathcal{F}(r,K)} \circ \psi(x)}(\kappa_{i,j,g}(a), \mathsf{pr}_2(r_j)(a)^{-1}) \\
&= \mathsf{ap}_{f_T}(\mathsf{ap}_{[\mathsf{id}_A]}(\kappa_{i,j,g}(a)))^{-1} \cdot \mathsf{pr}_2(r_j)(a)^{-1} \cdot \mathsf{ap}_{\mathsf{ind}_{\mathcal{F}(r,K)}}(\mathsf{ap}_\psi(\kappa_{i,j,g}(a))) && (\textit{Lemma } 1) \\
&= \mathsf{ap}_{f_T}(\mathsf{ap}_{[\mathsf{id}_A]}(\kappa_{i,j,g}(a)))^{-1} \cdot \mathsf{pr}_2(r_j)(a)^{-1} \cdot \mathsf{ap}_{\mathsf{pr}_1(r_j)}(\mathsf{pt}(F_{i,j,g})(a))^{-1} \cdot \mathsf{pr}_1(K_{i,j,g})(\mathsf{str}(F_i)(a)) \\
&\qquad && (\textit{via } \rho_\psi(i,j,g,a) \textit{ and then } \rho_{\mathsf{ind}_{\mathcal{F}(r,K)}}(i,j,g,\mathsf{str}(F_i)(a))) \\
&= \Big(\mathsf{pr}_1(K_{i,j,g})(\mathsf{str}(F_i)(a))^{-1} \cdot \mathsf{ap}_{\mathsf{pr}_1(r_j)}(\mathsf{pt}(F_{i,j,g})(a)) \cdot \mathsf{pr}_2(r_j)(a)\Big)^{-1} && (\textit{via } \rho_{[\mathsf{id}_A]}(i,j,g,a)) \\
&= \mathsf{pr}_2(r_i)(a)^{-1} && (\textit{by } \mathsf{ap}_{-^{-1}}(\mathsf{pr}_2(K_{i,j,g})(a)))
\end{aligned}$$

By induction on $\mathsf{colim}_\Gamma A$, this gives us a homotopy $f_T \circ [\mathsf{id}_A] \sim \mathsf{ind}_{\mathcal{F}(r,K)} \circ \psi$ and thus a function $h_{r,K} : \mathcal{P}_F \to T$



defined by pushout induction on $\mathcal{P}_F$. Finally, since $h(\mathsf{inl}(a)) \equiv f_T(a)$ for all $a : A$, we have

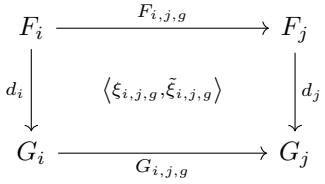
$$\Theta(r, K) \coloneqq \big(h_{r,K}, \lambda a.\mathsf{refl}_{f_T(a)}\big) \;:\; (\mathcal{P}_F, \mathsf{inl}) \to_A (T.f_T)$$

Each of the homotopies $\mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), T, f_T) \circ \Theta \sim \mathsf{id}$ and $\Theta \circ \mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), T, f_T) \sim \mathsf{id}$ requires intricate computations to prove. We leave their proofs to the formalization (see the folders [7, `Colimit-code/R-L-R`] and [7, `Colimit-code/L-R-L`], respectively). ◀

5.4 Action on maps

So far, we have defined a function $\mathsf{colim}^A_\Gamma \coloneqq \mathcal{P} : \mathsf{Ob}(\mathsf{Diag}(\Gamma, A/\mathcal{U})) \to \mathsf{Ob}(A/\mathcal{U})$ sending a Γ-shaped diagram in $A/\mathcal{U}$ to its A-colimit. We now make $\mathcal{P}$ a functor by describing its action on maps of diagrams. We want to describe this action in terms of the action of the ordinary colimit functor by using the special form of $\mathcal{P}$'s object function. Moreover, we must verify that such a description is correct by proving that $\mathcal{P}$ is left adjoint to the constant diagram functor, i.e., enjoys the universal property of the colimit functor.

Suppose that F and G are Γ-shaped diagrams in $A/\mathcal{U}$. The type of *natural transformations* from F to G consists of families $d : (i : \Gamma_0) \to \mathsf{ty}(F_i) \to_A \mathsf{ty}(G_i)$ of maps equipped with an A-homotopy $G_{i,j,g} \circ d_i \sim_A d_j \circ F_{i,j,g}$ for all i, j, g, where $\sim_A$ is as in Lemma 8. Consider a natural transformation $\delta \coloneqq \big(d, \langle \xi, \tilde{\xi} \rangle\big)$



from F to G, where $\langle -, - \rangle$ is as in Lemma 8. We form a map $\mathsf{colim}^A_\Gamma(\delta) : \mathsf{colim}^A_\Gamma(F) \to_A \mathsf{colim}^A_\Gamma(G)$ as follows. Start with the function $\mathsf{colim}_\Gamma(\mathcal{F}(F)) \xrightarrow{\bar{\delta}} \mathsf{colim}_\Gamma(\mathcal{F}(G))$ induced by the following map of $\mathcal{U}$-valued diagrams over Γ:

$$\begin{array}{ccc} \mathsf{ty}(F_i) & \xrightarrow{\mathsf{fun}(F_{i,j,g})} & \mathsf{ty}(F_j) \\ {\scriptstyle \mathsf{fun}(d_i)}\downarrow & \xi_{i,j,g} & \downarrow{\scriptstyle \mathsf{fun}(d_j)} \\ \mathsf{ty}(G_i) & \xrightarrow[\mathsf{fun}(G_{i,j,g})]{} & \mathsf{ty}(G_j) \end{array}$$

Note that for each $a : A$,

$$\tilde{\xi}_{i,j,g}(a) \; : \; \xi_{i,j,g}(\mathsf{str}(F_i)(a))^{-1} \cdot \mathsf{ap}_{\mathsf{fun}(G_{i,j,g})}(\mathsf{pt}(d_i)(a)) \cdot \mathsf{str}(G_{i,j,g})(a) = \mathsf{ap}_{\mathsf{fun}(d_j)}(\mathsf{pt}(F_{i,j,g})(a)) \cdot \mathsf{pt}(d_j)(a)$$

We may assume that $\tilde{\xi}_{i,j,g}(a)$ instead has the equivalent type

$$\xi_{i,j,g}(\mathsf{str}(F_i)(a)) = \underbrace{\mathsf{ap}_{\mathsf{fun}(G_{i,j,g})}(\mathsf{pt}(d_i)(a)) \cdot \mathsf{str}(G_{i,j,g})(a) \cdot \mathsf{pt}(d_j)(a)^{-1} \cdot \mathsf{ap}_{\mathsf{fun}(d_j)}(\mathsf{pt}(F_{i,j,g})(a))^{-1}}_{E_{i,j,g}(a)}$$

Here we abbreviate the right endpoint of the path by $E_{i,j,g}(a)$. Now, the triangle

$$\begin{array}{ccc} & \mathsf{colim}_\Gamma A & \\ {\scriptstyle \psi_F}\swarrow & & \searrow{\scriptstyle \psi_G} \\ \mathsf{colim}_\Gamma(\mathcal{F}(F)) & \xrightarrow[\bar{\delta}]{} & \mathsf{colim}_\Gamma(\mathcal{F}(G)) \end{array} \qquad (\psi\text{-tri})$$

commutes by induction on $\mathsf{colim}_\Gamma A$. Indeed, the computation rules of these functions give us

$$C_i(a) :\equiv \mathsf{ap}_{\iota_i}(\mathsf{pt}(d_i)(a)) \; : \; \bar{\delta}(\psi_F(\iota_i(a))) = \psi_G(\iota_i(a))$$

for all $i : \Gamma_0$ and $a : A$. Further, by defining $\Lambda_{i,j,g}(a) :\equiv C_j(a) \cdot \mathsf{ap}_{\psi_G}(\kappa_{i,j,g}(a))$, we have a chain of identities

$$\begin{aligned} & \mathsf{transp}^{\bar{\delta} \circ \psi_F \sim \psi_G}(\kappa_{i,j,g}(a), C_j(a)) \\ = \; & \mathsf{ap}_{\bar{\delta}}(\mathsf{ap}_{\psi_F}(\kappa_{i,j,g}(a)))^{-1} \cdot C_j(a) \cdot \mathsf{ap}_{\psi_G}(\kappa_{i,j,g}(a)) && (\textit{Lemma 1}) \\ = \; & \left(\mathsf{ap}_{\iota_j}(\xi_{i,j,g}(\mathsf{str}(F_i)(a)))^{-1} \cdot \kappa_{i,j,g}(\mathsf{fun}(d_i)(\mathsf{str}(F_i)(a)))\right)^{-1} \cdot \mathsf{ap}_{\iota_j \circ \mathsf{fun}(d_j)}(\mathsf{pt}(F_{i,j,g})(a)) \cdot \Lambda_{i,j,g}(a) \\ & && (\textit{via } \rho_{\psi_F}(i,j,g,a) \textit{ and then } \rho_{\bar{\delta}}(i,j,g,\mathsf{str}(F_i)(a))) \\ = \; & \left(\mathsf{ap}_{\iota_j}(E_{i,j,g}(\mathsf{str}(F_i)(a)))^{-1} \cdot \kappa_{i,j,g}(\mathsf{fun}(d_i)(\mathsf{str}(F_i)(a)))\right)^{-1} \cdot \mathsf{ap}_{\iota_j \circ \mathsf{fun}(d_j)}(\mathsf{pt}(F_{i,j,g})(a)) \cdot \Lambda_{i,j,g}(a) \\ & && (\textit{by } \mathsf{ap}_{\left(\mathsf{ap}_{\iota_j}(-)^{-1} \cdot \kappa_{i,j,g}(\mathsf{fun}(d_i)(\mathsf{str}(F_i)(a)))\right)^{-1} \cdot \cdots}(\tilde{\xi}_{i,j,g}(a))) \\ = \; & C_i(a) && (\textit{via } \rho_{\psi_G}(i,j,g,a)) \end{aligned}$$

for all $i, j : \Gamma_0$, $g : \Gamma_1(i,j)$, and $a : A$, so (ψ-tri) commutes. We now have a map

$$\begin{array}{ccccc} A & \longleftarrow & \mathsf{colim}_\Gamma A & \longrightarrow & \mathsf{colim}_\Gamma(\mathcal{F}(F)) \\ {\scriptstyle \mathsf{id}}\downarrow & \lambda x.\mathsf{refl}_{[\mathsf{id}](x)} & \downarrow{\scriptstyle \mathsf{id}} & \lambda x.C(x)^{-1} & \downarrow{\scriptstyle \bar{\delta}} \\ A & \longleftarrow & \mathsf{colim}_\Gamma A & \longrightarrow & \mathsf{colim}_\Gamma(\mathcal{F}(G)) \end{array}$$

of spans, which induces a function $\Psi_\delta : \mathcal{P}_F \to \mathcal{P}_G$ by the universal property of pushouts. Since $\Psi_\delta(\mathsf{inl}(a)) \equiv \mathsf{inl}(a)$ for all $a : A$, we may take $\mathsf{colim}^A_\Gamma(\delta)$ as $\left(\Psi_\delta, \lambda a.\mathsf{refl}_{\mathsf{inl}(a)}\right) : \mathcal{P}_F \to_A \mathcal{P}_G$.

To verify that the functor $\mathsf{Diag}(\Gamma, A/\mathcal{U}) \xrightarrow{\mathsf{colim}^A_\Gamma} A/\mathcal{U}$ we've defined is correct, we must show that it is left adjoint to the constant diagram functor. To do so, we construct the terms n_1 and n_2 required by Definition 12.

▶ **Lemma 16** ([6, Lemma 5.4.5]). *For every map $s : V \to_A U$, the following square commutes:*

$$\begin{array}{ccc} \mathsf{colim}^A_\Gamma(F) \to_A V & \xrightarrow{s\circ -} & \mathsf{colim}^A_\Gamma(F) \to_A U \\ \downarrow{\scriptstyle \mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F),V)} & & \downarrow{\scriptstyle \mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F),U)} \\ \mathsf{Cocone}_F(V) & \xrightarrow[\mathsf{Cocone}_F(s\circ -)]{} & \mathsf{Cocone}_F(U) \end{array}$$

▶ **Lemma 17** ([6, Lemma 5.4.12]). *For every $V : \mathsf{Ob}(A/\mathcal{U})$ and $\delta : F \Rightarrow_A G$, the following square commutes:*

$$\begin{array}{ccc} \mathsf{colim}^A_\Gamma(G) \to_A V & \xrightarrow{-\circ\mathsf{colim}^A_\Gamma(\delta)} & \mathsf{colim}^A_\Gamma(F) \to_A V \\ \downarrow{\scriptstyle \mathsf{postcomp}(\mathcal{K}(\mathcal{P}_G),V)} & & \downarrow{\scriptstyle \mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F),V)} \\ \mathsf{Cocone}_G(V) & \xrightarrow[\mathsf{Cocone}^V(-\circ\delta)]{} & \mathsf{Cocone}_F(V) \end{array}$$

The two lower horizontal functions are induced by post-composition with s and pre-composition with δ [6, Definition 5.4.11], respectively.

Lemma 16 is a routine computation, whereas Lemma 17 is quite difficult. The proof of Lemma 17 is easier for the map $\mathsf{colim}^A_\Gamma(F) \to_A \mathsf{colim}^A_\Gamma(G)$ obtained by applying the inverse of $\mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), \mathcal{P}_G, \mathsf{inl})$ to the canonical F-cocone on $\mathcal{P}_G$ induced by δ. Therefore, we decide to reduce the goal to an equality between this map and $\mathsf{colim}^A_\Gamma(\delta)$. We achieve this by showing that they belong to the same fiber of $\mathsf{postcomp}(\mathcal{K}(\mathcal{P}_F), \mathcal{P}_G, \mathsf{inl})$, which is contractible by Theorem 15. Though much easier than a direct approach to Lemma 17, this method requires intricate computations. We have formalized both Lemma 16 and Lemma 17 (see [7, `Colimit-code/Map-Nat/CosColimitPstCmp.agda`] and [7, `Colimit-code/Map-Nat/CosColimitPreCmp.agda`], respectively).

6 Creation of colimits

Classically, if $\mathcal{D}$ is an ∞-category, then all forgetful functors of ∞-coslices create $\mathcal{D}$-shaped colimits when the ∞-groupoid obtained by freely inverting all morphisms of $\mathcal{D}$ is contractible (see [15, Tag 02KS]). Theorem 18 expresses the same result inside HoTT.

▶ **Theorem 18.** *The forgetful functor $A/\mathcal{U} \to \mathcal{U}$ creates colimits over trees.*

Proof. The idea is that a tree has no cycles, and thus we have no distinguished loops to fill. As a result, coslice colimits over trees look the same as their underlying colimits in $\mathcal{U}$.

To be precise, suppose that Γ is a tree and let F be a diagram in $A/\mathcal{U}$ over Γ. Then the function $[\mathsf{id}_A] : \mathsf{colim}_\Gamma A \to A$ is an equivalence, and one can check that

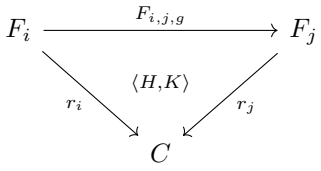
$$\begin{array}{ccc} \mathsf{colim}_\Gamma A & \xrightarrow{\psi} & \mathsf{colim}_\Gamma(\mathcal{F}(F)) \\ \downarrow{\scriptstyle [\mathsf{id}_A]} & & \downarrow{\scriptstyle \mathsf{id}} \\ A & \xrightarrow[\psi\circ[\mathsf{id}_A]^{-1}]{} & \mathsf{colim}_\Gamma(\mathcal{F}(F)) \end{array}$$

is a pushout square. By uniqueness of pushouts, this gives us an equivalence $\gamma : \mathcal{P}_F \xrightarrow{\simeq} \mathsf{colim}_\Gamma(\mathcal{F}(F))$ such that $\gamma(\mathsf{inr}(\iota_i(x))) \equiv \iota_i(x)$ for all $i : \Gamma_0$ and $x : \mathsf{ty}(F_i)$. We also see that

$$\mathsf{ap}_\gamma(\mathsf{ap}_{\mathsf{inr}}(\kappa_{i,j,g}(x))) = \mathsf{ap}_{\gamma\circ\mathsf{inr}}(\kappa_{i,j,g}(x)) \equiv \mathsf{ap}_{\mathsf{id}}(\kappa_{i,j,g}(x)) = \kappa_{i,j,g}(x)$$

for all $i, j : \Gamma_0$, $g : \Gamma_1(i, j)$, and $x : \mathsf{ty}(F_i)$. This means that γ is a morphism of cocones under $\mathcal{F}(F)$. It follows that the forgetful functor preserves colimits over Γ.

It remains to prove that the forgetful functor reflects colimits over Γ. Consider an A-cocone $\mathcal{J}$ under F



as well as the cocone $\mathcal{F}(\mathcal{J}) \coloneqq (\mathsf{ty}(C), \mathsf{fun} \circ r, H)$ under $\mathcal{F}(F)$ obtained by applying the forgetful functor to $\mathcal{J}$. Suppose that $\mathcal{F}(\mathcal{J})$ is colimiting in $\mathcal{U}$. By the universal property of colimits in $A/\mathcal{U}$, we have a morphism $(\mathcal{P}_F, \mathsf{inl}) \xrightarrow{\tau} C$ of A-cocones, which induces a morphism $\mathcal{P}_F \xrightarrow{\mathcal{F}(\tau)} \mathsf{ty}(C)$ of cocones in $\mathcal{U}$. This morphism is unique by the universal property of ordinary colimits. Moreover, by the uniqueness of ordinary colimits, there is a cocone equivalence from $\mathcal{P}_F$ to $\mathsf{ty}(C)$ as both of them are colimiting. This implies $\mathcal{F}(\tau)$ is an equivalence. Thus, τ is an A-cocone morphism whose underlying function $\mathcal{P}_F \to \mathsf{ty}(C)$ is an equivalence. This means that τ is an A-cocone equivalence, so that $\mathcal{J}$ is colimiting. ◀

▶ Remark. The fact that the forgetful functor $\mathcal{U}^* \to \mathcal{U}$ from pointed types creates *pushouts* appears in the agda-unimath library, though without proof [23, Pushouts of pointed types].

Theorem 18 lets us lift powerful features of ordinary colimits to A-colimits. For example, let Γ be a graph and F be an A-diagram over Γ. We say that $\mathsf{colim}^A_\Gamma(F)$ is *universal*, or *pullback-stable* [19], if for every pullback square

$$\begin{array}{ccc} \mathsf{colim}^A_\Gamma(F) \times_V Y & \xrightarrow{\pi_2} & Y \\ {\scriptstyle \pi_1}\downarrow & \lrcorner & \downarrow{\scriptstyle h} \\ \mathsf{colim}^A_\Gamma(F) & \xrightarrow[f]{} & V \end{array} \tag{pb}$$

in $A/\mathcal{U}$, the canonical map

$$\sigma_{f,h} \;:\; \mathsf{colim}^A_{i:\Gamma}(F_i \times_V Y) \to_A \mathsf{colim}^A_\Gamma(F) \times_V Y$$

is an equivalence.[5] The distinguishing feature of a LCC ∞-category, such as $\mathcal{U}$, is that all of its colimits are universal. Although the coslice of a LCC category need not be LCC, we now show that all of its colimits over trees are universal.

▶ **Lemma 19.** *The forgetful functor $\mathcal{F} : A/\mathcal{U} \to A$ preserves limits.*

Proof. The functor $\mathcal{F}$ is right adjoint to the functor $X \mapsto X + A$, so it preserves limits. ◀

▶ **Theorem 20.** *All colimits in $\mathcal{U}$ are universal.*

[5] We show how to construct pullbacks in $A/\mathcal{U}$ in [6, Note 6.0.4].

We have formalized Theorem 20 in Agda (see the folder [7, `Pullback-stability`]).

▶ **Corollary 21.** *For each tree* Γ *and each* A*-diagram* F *over* Γ*, the colimit* $\mathsf{colim}^A_\Gamma(F)$ *is universal.*

Proof. Suppose that Γ is a tree and consider the pullback square (`pb`). By Theorem 18 combined with Theorem 20, the function

$$\mathsf{ty}(\mathsf{colim}^A_{i:\Gamma}(F_i \times_V Y)) \xrightarrow{\mathsf{fun}(\sigma_{f,h})} \mathsf{ty}(\mathsf{colim}^A_\Gamma(F)) \times_{\mathsf{ty}(V)} \mathsf{ty}(Y)$$

is an equivalence. The codomain is in this form because $\mathcal{F}$ preserves pullbacks by Lemma 19. It follows that $\sigma_{f,h}$ is an equivalence. ◀

7 Preservation of the left class of an OFS

In this section, we combine our construction of $\mathsf{colim}^A_\Gamma(\delta)$ from Section 5.4 with Theorem 13 to prove that colim^A_Γ always preserves the left class of an OFS on $\mathcal{U}$. We assume the univalence axiom to have access to the the tools of univalent bicategories developed in Section 4.

Let $(\mathcal{L}, \mathcal{R})$ be an OFS on $\mathcal{U}$. For all diagrams $F, G : \mathcal{D}_\Gamma \coloneqq \mathsf{Diag}(\Gamma, \mathcal{U})$ and natural transformations $(H, \gamma) : F \Rightarrow G$, define the predicates $\widehat{\mathcal{L}}(H,\gamma) \coloneqq (i : \Gamma_0) \to \mathcal{L}(H_i)$ and $\widehat{\mathcal{R}}(H,\gamma) \coloneqq (i : \Gamma_0) \to \mathcal{R}(H_i)$.

▶ **Lemma 22** ([6, Theorem 7.0.8])**.** *Let* $Q : F \Rightarrow G$*. The following type is contractible:*

$$\mathsf{fact}_{\widehat{\mathcal{L}},\widehat{\mathcal{R}}}(Q) \coloneqq \sum_{M:\mathcal{D}_\Gamma} \sum_{S:F\Rightarrow M} \sum_{T:M\Rightarrow G} (T \circ S = Q) \times \widehat{\mathcal{L}}(S) \times \widehat{\mathcal{R}}(T).$$

By Lemma 22, we see that $(\mathcal{L}, \mathcal{R})$ lifts levelwise to $\mathcal{D}_\Gamma$. Since the functor $\mathsf{const}_\Gamma : \mathcal{U} \to \mathcal{D}_\Gamma$ clearly takes $\mathcal{R}$ to $\widehat{\mathcal{R}}$, we deduce that $\mathsf{colim}_\Gamma(-)$ takes $\widehat{\mathcal{L}}$ to $\mathcal{L}$ by Theorem 13.[6]

For each $X, Y : \mathsf{Ob}(A/\mathcal{U})$, consider the predicate $\mathcal{L}_A(f,p) \coloneqq \mathcal{L}(f)$ on $X \to_A Y$. Also, define the predicate $\widehat{\mathcal{L}}_A$ on maps of A-diagrams over Γ by $\widehat{\mathcal{L}}_A(H,\gamma) \coloneqq \prod_{i:\Gamma_0} \mathcal{L}_A(H_i)$. Then the functor colim^A_Γ takes $\widehat{\mathcal{L}}_A$ to $\mathcal{L}_A$. Indeed, consider a map $\delta : \mathcal{A} \Rightarrow_A \mathcal{B}$ of A-diagrams over Γ. The underlying function of $\mathsf{colim}^A_\Gamma(\delta)$ is induced by the morphism of spans

$$\begin{array}{ccccc} A & \longleftarrow & \mathsf{colim}_\Gamma A & \longrightarrow & \mathsf{colim}_\Gamma(\mathcal{F}(\mathcal{A})) \\ {\scriptstyle \mathsf{id}}\downarrow & & \downarrow{\scriptstyle \mathsf{id}} & & \downarrow{\scriptstyle \bar{\delta}} \\ A & \longleftarrow & \mathsf{colim}_\Gamma A & \longrightarrow & \mathsf{colim}_\Gamma(\mathcal{F}(\mathcal{B})) \end{array}$$

Thus, if δ is in $\widehat{\mathcal{L}}_A$, then all three vertical functions are in $\mathcal{L}$. Since a map of spans is a map of diagrams, we see that $\mathsf{colim}^A_\Gamma(\delta)$ is in $\mathcal{L}_A$.

Recall that a type X is $(\mathcal{L}, \mathcal{R})$-*connected* if the function $X \to \mathbf{1}$ is in $\mathcal{L}$. If F is a diagram of pointed types over Γ such that each $\mathsf{ty}(F_i)$ is $(\mathcal{L}, \mathcal{R})$-connected, then the type $\mathsf{colim}^*_\Gamma(F) \coloneqq \mathsf{colim}^{\mathbf{1}}_\Gamma(F)$ is also $(\mathcal{L}, \mathcal{R})$-connected. Indeed, we can deduce that $\mathsf{colim}^*_\Gamma \mathbf{1} = \mathbf{1}$ from the construction of $\mathcal{P}_F$. Thus, colim^*_Γ takes the unique map $F \Rightarrow_* \mathbf{1}$ of pointed diagrams to $(c, c_p) : \mathsf{colim}^*_\Gamma(F) \to_* \mathsf{colim}^*_\Gamma \mathbf{1}$ where $c : \mathsf{colim}^*_\Gamma(F) \to \mathbf{1}$ is the constant map. In addition, $\mathcal{L}(c)$ holds because colim^*_Γ takes $\widehat{\mathcal{L}}_{\mathbf{1}}$ to $\mathcal{L}_{\mathbf{1}}$.

[6] The adjunction $\mathsf{colim}_\Gamma \dashv \mathsf{const}_\Gamma$ follows directly from the equivalence `postcomp` in Section 3.3.

▶ **Example 23.** Let n be a truncation level [27, Chapter 7]. The archetypal OFS in HoTT has n-connected functions as its left class and n-truncated ones as its right. Thus, by our preceding argument, if each $\mathsf{ty}(F_i)$ is n-connected, then so is $\mathsf{colim}^*_\Gamma(F)$.

Now, let $-1 \leq k < \infty$ also be a truncation level. Recall the category $(n,k)\,\mathsf{GType}$ of *k-tuply groupal n-groupoids* [2]. (These are examples of *higher groups*, in the sense of group theory). This is equivalent to the full subcategory $\mathcal{U}^*_{\geq k,\leq n+k}$ of $\mathcal{U}^*$ on $(k-1)$-connected, $(n+k)$-truncated pointed types. For each truncation level m, consider the m-truncation functor $\|-\|_m : A/\mathcal{U} \to A/\mathcal{U}$, which takes a type X to the universal m-type admitting a function from X [27, Section 7.3]. This functor preserves colimits as a left adjoint [6, Proposition 3.4.6], and its associated counit is an isomorphism. It follows that $\mathcal{U}^*_{\geq k,\leq n+k}$, hence $(n,k)\,\mathsf{GType}$, inherits colimits over graphs from $\mathcal{U}^*$.

It is a special feature of *pointed* colimits that they always preserve n-connectedness. If Γ is not a tree, then colim_Γ may fail to preserve n-connectedness. Indeed, let Γ be the graph with a single point $*$ and a single edge from $*$ to itself. Define the diagram F over Γ by $F_0(*) \coloneqq \mathbf{1}$ and $F_{*,*,*} \coloneqq \mathsf{id}_{\mathbf{1}}$. Then $\mathsf{colim}_\Gamma(F)$ is equivalent to the circle S^1, which is not 1-connected.

▶ Remark 24. Although Example 23 is only about pointed types, we do benefit from the formulation of the main connection for general coslices. Indeed, for each object G of $\mathcal{U}^*_{\geq k,\leq n+k}$, the coslice $G/\mathcal{U}^*_{\geq k,\leq n+k}$ is a reflective subcategory of $\mathsf{ty}(G)/\mathcal{U}_{\geq k,\leq n+k}$ for which the reflector is 2-*coherent* [6, Definition B.0.1]. (Here $\mathcal{U}_{\geq k,\leq n+k}$ denotes the subuniverse of $(k-1)$-connected, $(n+k)$-truncated types.) Hence $G/\mathcal{U}^*_{\geq k,\leq n+k}$ inherits colimits over graphs from the coslice $\mathsf{ty}(G)/\mathcal{U}_{\geq k,\leq n+k}$ [6, Corollary 7.1.3]. In its full generality, Section 5 gives us an explciit construction of such colimits.

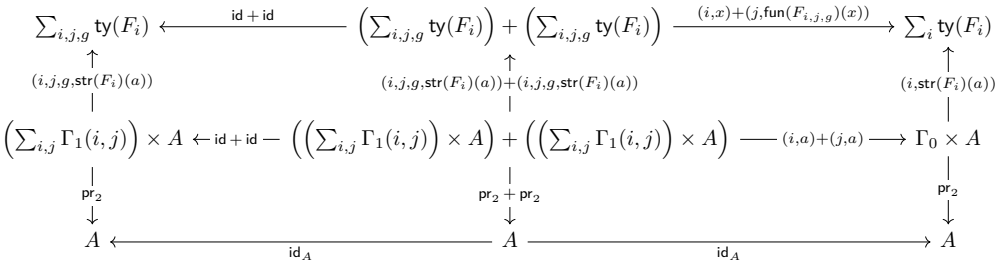
In particular, let $n, m : \mathbb{N}$ with $n > 0$ and $m < n$. The Eilenberg-MacLane space $K(\mathbb{Z}, n)$ [13] is the free group on one generator in the category $(n,m)\,\mathsf{GType}$. Therefore, when $m > 0$, we may view $K(\mathbb{Z},n)/\mathcal{U}^*_{\geq m,\leq n+m}$ as a higher version of the category of *pointed abelian groups* [18]. We see, then, that Section 5 lets us build colimits of higher pointed abelian groups inside HoTT.

8 Mapping colimits to weak limits

Finally, we look at the interaction between colimits and (Eilenberg-Steenrod) cohomology theories. Specifically, we apply the 3×3 lemma to the main connection to obtain the familiar construction of $\mathsf{colim}^A_\Gamma(F)$ as a pushout of coproducts in $A/\mathcal{U}$. Afterward, we apply this new construction to the Mayer-Vietoris sequence to prove that cohomology theories send finite colimits to weak limits in **Set** assuming the axiom of choice.

8.1 Decomposition of A-colimits into simpler pieces

To make use of the 3×3 lemma, we first form the following grid of commuting squares:



Call the pushouts of the left, middle, and right vertical spans V_1, V_2, and V_3, respectively. Call the pushouts of the top, middle, and bottom horizontal spans H_1, H_2, and H_3, respectively. We can form two additional pushouts from this grid:

$$\begin{array}{ccc} V_2 & \xrightarrow{\delta_2} & V_3 \\ {\scriptstyle\delta_1}\downarrow & \ulcorner & \downarrow{\scriptstyle\mathsf{inr}} \\ V_1 & \xrightarrow[\mathsf{inl}]{} & P_V \end{array} \qquad \begin{array}{ccc} H_2 & \xrightarrow{\eta_1} & H_1 \\ {\scriptstyle\eta_2}\downarrow & \ulcorner & \downarrow{\scriptstyle\mathsf{inr}} \\ H_3 & \xrightarrow[\mathsf{inl}]{} & P_H \end{array}$$

- δ_1 denotes the function induced by the middle-to-left map of spans;
- δ_2 denotes the function induced by the middle-to-right map of spans;
- η_1 denotes the function induced by the middle-to-top map of spans; and
- η_2 denotes the function induced by the middle-to-bottom map of spans.

The 3×3 lemma now gives us an equivalence $\tau_1 : P_H \xrightarrow{\simeq} P_V$ of types defined by double induction on pushouts [12, Section VII].

▶ Note. Let Δ be a discrete graph and G an A-diagram over Δ. The pushout

$$\begin{array}{ccc} \Delta_0 \times A & \xrightarrow{(i,a)\mapsto(i,\mathsf{str}(G_i)(a))} & \sum_{i:\Delta_0} \mathsf{ty}(G_i) \\ {\scriptstyle\mathsf{pr}_2}\downarrow & & \downarrow{\scriptstyle\mathsf{inr}} \\ A & \xrightarrow[\mathsf{inl}]{} & D \end{array}$$

together with inl is the coproduct of the G_i in $A/\mathcal{U}$. We denote D by $\bigvee_{i:\Delta_0} \mathsf{ty}(G_i)$.

▶ **Lemma 25.** *We have two equivalences of spans*

$$\begin{array}{ccccc} A & \xleftarrow{[\mathsf{id}_A]} & \mathsf{colim}_\Gamma A & \xrightarrow{\psi} & \mathsf{colim}_\Gamma(\mathcal{F}(F)) \\ \downarrow{\scriptstyle\simeq} & & \downarrow{\scriptstyle\simeq} & & \downarrow{\scriptstyle\simeq} \\ H_3 & \xleftarrow[\eta_2]{} & H_2 & \xrightarrow[\eta_1]{} & H_1 \end{array}$$

$$\begin{array}{ccccc} V_1 & \xleftarrow{\delta_1} & V_2 & \xrightarrow{\delta_2} & V_3 \\ \| & & \downarrow{\scriptstyle\simeq} & & \| \\ \bigvee_{i,j,g} \mathsf{ty}(F_i) & \xleftarrow[\mathsf{id}\vee\mathsf{id}]{} & \left(\bigvee_{i,j,g} \mathsf{ty}(F_i)\right) \vee \left(\bigvee_{i,j,g} \mathsf{ty}(F_i)\right) & \xrightarrow{\nu} & \bigvee_i \mathsf{ty}(F_i) \end{array}$$

where ν is defined by double induction on pushouts through the commuting square

$$\begin{array}{ccc} A & \longrightarrow & \bigvee_{i,j,g} \mathsf{ty}(F_i) \\ \downarrow & {\scriptstyle\mathsf{refl}_{\mathsf{inl}(a)}} & \downarrow{\scriptstyle(i,j,g,x)\mapsto\mathsf{inr}(j,\mathsf{fun}(F_{i,j,g})(x))} \\ \bigvee_{i,j,g} \mathsf{ty}(F_i) & \xrightarrow[(i,j,g,x)\mapsto\mathsf{inr}(i,x)]{} & \bigvee_i \mathsf{ty}(F_i) \end{array}$$

Notice that the pushout of the topmost span appearing in Lemma 25 is exactly $\mathcal{P}_F$. As a result, the equivalence supplied by the 3×3 lemma gives us $\mathsf{colim}_\Gamma^A(F)$ as a familiar pushout of coproducts.

▶ **Corollary 26** ([6, Corollary 5.5.3]). *We have a pushout square*

$$\begin{array}{ccc} \left(\bigvee_{i,j,g} \mathsf{ty}(F_i)\right) \vee \left(\bigvee_{i,j,g} \mathsf{ty}(F_i)\right) & \xrightarrow{\nabla} & \bigvee_i \mathsf{ty}(F_i) \\ \downarrow{\scriptstyle \mathsf{id} \vee \mathsf{id}} & \ulcorner & \downarrow \\ \bigvee_{i,j,g} \mathsf{ty}(F_i) & \longrightarrow & \mathsf{colim}^A_\Gamma(F) \end{array}$$

8.2 Weak continuity of cohomology

With this new construction of colim^A_Γ, we can transfer the weak continuity of cohomology to HoTT. This application is described in detail in [6, Section 8].

Let **Ab** denote the category of abelian groups. Suppose that $H^* : (\mathcal{U}^*)^{\mathrm{op}} \to \mathbf{Ab}$ is a cohomology theory.[7] Consider the pushout

$$\begin{array}{ccc} Z & \xrightarrow{g} & Y \\ {\scriptstyle f}\downarrow & \ulcorner & \downarrow{\scriptstyle \mathsf{inr}} \\ X & \xrightarrow[\mathsf{inl}]{} & P \end{array}$$

of a span of pointed maps. In [4], Cavallo constructs the *Mayer-Vietoris sequence* for P, a long exact sequence (LES) of the form

$$\cdots \quad H^{n-1}(P) \longrightarrow H^{n-1}(X) \times H^{n-1}(Y) \longrightarrow H^{n-1}(Z)$$

$$H^n(P) \xrightarrow[(H^n(\mathsf{inl}),H^n(\mathsf{inr}))]{} H^n(X) \times H^n(Y) \xrightarrow[H^n(f)-H^n(g)]{} H^n(Z) \quad \cdots$$

Let F be a diagram of pointed types over a *finite* graph Γ, which means that Γ_0 is finite and $\Gamma_1(i,j)$ is finite for all $i,j : \Gamma_0$. As cohomology preserves finite wedges [4, Section 4.2], Corollary 26 combined with this LES gives us an exact squence

$$H^n(\mathsf{colim}^*_\Gamma(F)) \xrightarrow{\zeta_n} \prod_{i,j,g} H^n(F_i) \times \prod_i H^n(F_i) \xrightarrow{\mu_n - \nu_n} \prod_{i,j,g} H^n(F_i) \times \prod_{i,j,g} H^n(F_i) \qquad (\texttt{prod-cohom})$$

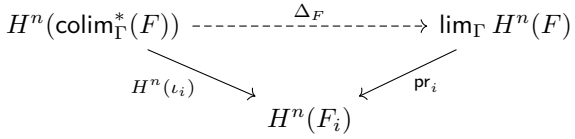
for each $n : \mathbb{Z}$.[8] Here, ζ_n is defined as the composite

$$\begin{array}{ccc} H^n(\mathsf{colim}^*_\Gamma(F)) & \dashrightarrow^{\zeta_n} & \prod_{i,j,g} H^n(F_i) \times \prod_i H^n(F_i) \\ {\scriptstyle (H^n(\mathsf{inl}),H^n(\mathsf{inr}))}\downarrow & \nearrow_{\cong \times \cong} & \\ H^n(\bigvee_{i,j,g} F_i) \times H^n(\bigvee_i F_i) & & \end{array}$$

and μ_n and ν_n are defined by $(f,h) \mapsto (f, \lambda i \lambda j \lambda g.H^n(F_{i,j,g})(h_j))$ and $(f,h) \mapsto (f, \lambda i \lambda j \lambda g.h_i)$, respectively. Moreover, the universal property of limits in **Ab** induces a commuting triangle

[7] See [1, Section 6] for a description of Eilenberg-Steenrod cohomology theory inside HoTT. A slightly more general definition, which works in our setting, is found in [6, Section 8.1].

[8] When H^* is a singular cohomology theory, we may extend the class of graphs to those satisfying the set-level axiom of choice, in the sense of [1, Definition 6.1].



for each $i : \Gamma_0$, induced by the cone $(H^n(\mathsf{colim}^*_\Gamma(F)), H^n(\iota))$ over $H^n(F)$. One can check that the exactness of (`prod-cohom`) implies that Δ_F is epic as a map of sets.

At this stage, if we were in a classical system, then it would follow that Δ_F has a section, which in turn would imply that $H^n(\mathsf{colim}^*_\Gamma(F))$ is a weak limit in **Set**. Inside HoTT, we may assume the axiom of choice [27, Chapter 3.8] to conclude that Δ_F is *merely* a weak limit.[9] In this sense, H^* enjoys a restricted version of weak continuity inside HoTT.

9 Conclusion and future work

We explored colimits inside HoTT. The heart of our work was the connection between A-colimits and ordinary colimits, i.e., *the main connection*. To derive the main connection, we found an explicit construction of A-colimits that was tailored to reveal the connection. We used the main connection to prove that the forgetful functor from a coslice creates colimits over trees and that A-colimits over trees are universal. We also used the main connection to examine how colimits interact with factorization systems. As a result, we found that all pointed colimits preserve n-connectedness, which implies that higher groups are closed under colimits on directed graphs. Finally, we used the main connection to see that cohomology takes finite colimits to weak limits in **Set** assuming the axiom of choice.

A natural direction is to extend our development to colimits of diagrams over 2-computads [26]. To our knowledge, colimits of type-valued diagrams over higer-dimensional graphs have not been developed in the untruncated setting. We believe both Section 6 and Section 7 can be generalized to the setting of 2-computads.

References

1. Ulrik Buchholtz and Kuen-Bang Hou (Favonia). Cellular Cohomology in Homotopy Type Theory. *Logical Methods in Computer Science*, Volume 16, Issue 2, 2020. `doi:10.23638/LMCS-16(2:7)2020`.
2. Ulrik Buchholtz, Floris van Doorn, and Egbert Rijke. Higher Groups in Homotopy Type Theory. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 205–214, New York, NY, USA, 2018. Association for Computing Machinery. `doi:10.1145/3209108.3209150`.
3. Paolo Capriotti and Nicolai Kraus. Univalent higher categories via complete Semi-Segal types. *Proc. ACM Program. Lang.*, 2(POPL), 2017. `doi:10.1145/3158132`.
4. Evan Cavallo. Synthetic Cohomology in Homotopy Type Theory. Master's thesis, Carnegie Mellon University, 2015. URL: `https://staff.math.su.se/evan.cavallo/works/thesis15.pdf`.
5. J Daniel Christensen and Luis Scoccola. The Hurewicz theorem in homotopy type theory. *Algebraic and Geometric Topology*, 23(5):2107–2140, 2023. `doi:10.2140/agt.2023.23.2107`.
6. Perry Hart and Kuen-Bang Hou. Coslice Colimits in Homotopy Type Theory, 2024. `arXiv:2411.15103`.
7. Perry Hart and Kuen-Bang Hou (Favonia). A formalized construction of coslice colimits, 2024. v0.1.0. URL: `https://github.com/PHart3/colimits-agda/tree/v0.1.0`.

[9] As in [27, Chapter 3.10], the adverb *merely* refers to propositional truncation.

8 Kuen-Bang Hou (Favonia), Eric Finster, Daniel R. Licata, and Peter LeFanu Lumsdaine. A Mechanization of the Blakers-Massey Connectivity Theorem in Homotopy Type Theory. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '16, pages 565–574, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2933575.2934545.

9 Krzysztof Kapulkin and Peter LeFanu Lumsdaine. The simplicial model of Univalent Foundations (after Voevodsky). *J. Eur. Math. Soc.*, 23(6):2071–2126, 2021. doi:10.4171/JEMS/1050.

10 Max Kelly. On MacLane's Conditions for Coherence of Natural Associativities, Commutativities, etc. *Journal of Algebra*, 1(4):397–402, 1964. doi:10.1016/0021-8693(64)90018-3.

11 Nicolai Kraus and Jakob von Raumer. Path Spaces of Higher Inductive Types in Homotopy Type Theory . In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, Los Alamitos, CA, USA, June 2019. IEEE Computer Society. doi:10.1109/LICS.2019.8785661.

12 Daniel R. Licata and Guillaume Brunerie. A Cubical Approach to Synthetic Homotopy Theory. In *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 92–103, 2015. doi:10.1109/LICS.2015.19.

13 Daniel R. Licata and Eric Finster. Eilenberg-MacLane spaces in homotopy type theory. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, CSL-LICS '14, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2603088.2603153.

14 Jacob Lurie. Higher Algebra. Unpublished, 2017. URL: https://www.math.ias.edu/~lurie/papers/HA.pdf.

15 Jacob Lurie. Kerodon. https://kerodon.net, 2024.

16 nLab authors. created limit. https://ncatlab.org/nlab/show/created+limit, 2024. Revision 21.

17 nLab authors. (infinity,1)-limit. https://ncatlab.org/nlab/show/%28%E2%88%9E%2C1%29-limit, 2024. Revision 78.

18 nLab authors. pointed abelian group. https://ncatlab.org/nlab/show/pointed+abelian+group, November 2024. Revision 3.

19 nLab authors. pullback-stable colimit. https://ncatlab.org/nlab/show/pullback-stable+colimit, October 2024. Revision 18.

20 Emily Riehl. *Categorical Homotopy Theory*. New Mathematical Monographs. Cambridge University Press, 2014. doi:10.1017/CBO9781107261457.

21 Egbert Rijke. Introduction to Homotopy Type Theory, 2022. arXiv:2212.11082.

22 Egbert Rijke, Michael Shulman, and Bas Spitters. Modalities in homotopy type theory. *Logical Methods in Computer Science*, Volume 16, Issue 1, January 2020. doi:10.23638/LMCS-16(1:2)2020.

23 Egbert Rijke, Elisabeth Stenholm, Jonathan Prieto-Cubides, Fredrik Bakke, and others. The agda-unimath library. URL: https://github.com/UniMath/agda-unimath/.

24 Kristina Sojakova. Higher Inductive Types as Homotopy-Initial Algebras. *SIGPLAN Not.*, 50(1):31–42, 2015. doi:10.1145/2775051.2676983.

25 Kristina Sojakova, Floris van Doorn, and Egbert Rijke. Sequential Colimits in Homotopy Type Theory. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '20, pages 845–858, 2020. doi:10.1145/3373718.3394801.

26 Ross Street. Limits indexed by category-valued 2-functors. *Journal of Pure and Applied Algebra*, 8(2):149–181, 1976. doi:10.1016/0022-4049(76)90013-X.

27 The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

28 Floris van Doorn, Jakob von Raumer, and Ulrik Buchholtz. Homotopy Type Theory in Lean. In *Interactive Theorem Proving*, pages 479–495. Springer International Publishing, 2017. doi:10.1007/978-3-319-66107-0_30.

A Rewriting Theory for Quantum λ-Calculus

Claudia Faggian
IRIF, CNRS, Université Paris Cité, France

Gaetan Lopez
IRIF, CNRS, Université Paris Cité, France

Benoît Valiron
Université Paris-Saclay, CNRS, CentraleSupélec, ENS Paris-Saclay, Inria,
Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France

Abstract

Quantum lambda calculus has been studied mainly as an idealized programming language – the evaluation essentially corresponds to a deterministic abstract machine. Very little work has been done to develop a rewriting theory for quantum lambda calculus. Recent advances in the theory of probabilistic rewriting give us a way to tackle this task with tools unavailable a decade ago. Our primary focus are standardization and normalization results.

2012 ACM Subject Classification Theory of computation → Quantum computation theory; Theory of computation → Operational semantics; Theory of computation → Equational logic and rewriting; Theory of computation → Linear logic

Keywords and phrases quantum lambda-calculus, probabilistic rewriting, operational semantics, asymptotic normalization, principles of quantum programming languages

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.47

Related Version *Extended Version*: `http://arxiv.org/abs/2411.14856` [25]

Funding This work has been partially funded by the French National Research Agency (ANR) by the projects PPS ANR-19-CE48-0014, TaQC ANR-22-CE47-0012 and within the framework of "Plan France 2030", under the research projects EPIQ ANR-22-PETQ-0007, OQULUS ANR-23-PETQ-0013, HQI-Acquisition ANR-22-PNCQ-0001 and HQI-R&D ANR-22-PNCQ-0002.

1 Introduction

Quantum computation is a model of computation in which one has access to data coded on the state of objects governed by the law of quantum physics. Due to the unique nature of quantum mechanics, quantum data exhibits several non-intuitive properties [37]: it is non-duplicable, it can exist in superposition, and reading the memory exhibits a probabilistic behavior. Nonetheless, the mathematical formalization is well-established: the state of a quantum memory and the available manipulations thereof can be expressed within the theory of Hilbert spaces.

Knill's QRAM model [34] describes a generic interface for interacting with such a quantum memory. The memory is modeled with uniquely identified quantum registers, each holding one quantum bit – also called a qubit. The interface should make it possible to create and discard registers and apply elementary operations on arbitrary registers. These operations consist of *unitary gates* and *measurements*. The former are internal, local modifications of the memory state, represented by a quantum circuit, while the latter are the operations for reading the memory. Measurements are *probabilistic operations* returning a classical bit of information.

Quantum algorithms are typically designed with a model akin to Knill's QRAM [37]. A quantum algorithm consists of the description of a sequence of quantum operations, measurements, and classical processing. The control flow is purely classical, and generally, the algorithm's behavior depends on the results of past measurements. An algorithm, therefore, mixes classical processing and interaction with quantum memory in a potentially arbitrary way: Quantum programming languages should be designed to handle it.

Quantum λ-Calculus and Linear Logics. In the last 20 years, many proposals for quantum programming languages have emerged [28, 30, 40, 39, 8, 11]. Similar to classical languages, the paradigm of higher-order, functional quantum programming languages have been shown to be a fertile playground for the development of well-founded, formal quantum languages aiming at the formal analysis of quantum programs [40, 11].

The *quantum λ-calculus* of Selinger&Valiron [42] lies arguably at the foundation of the development of higher-order, quantum functional programming languages [14, 13, 38, 12, 35]. Akin to other extensions of lambda-calculus with probabilistic [17, 15, 21] or non-deterministic behavior [16], the quantum lambda calculus extends the regular lambda calculus – core of functional programming languages – with a set interface to manipulate a quantum memory. Due to the properties of the quantum memory, quantum lambda-calculi should handle non-duplicable data and probabilistic behavior.

One of the critical points that a quantum programming language should address is the problem of the *non-duplicability* of quantum data. In the literature, non-duplicable data is usually captured with tools coming from linear logic. The first approach [42, 38, 12] consists in using types, imposing all functions to be linear by default, and with the use of a special type $!A$ to explicitly pinpoint duplicable objects of type A. An alternative – untyped – approach [13, 14] considers instead an untyped lambda calculus, augmented with a special term construct "!" and validity constraints to forbid the duplication of qubits.

Probabilistic and Infinitary Behavior. A quantum λ-calculus featuring all of quantum computation should not only permit the manipulation of quantum register with unitary operations but should also give the possibility to *measure* them, and retrieve a classical bit of information. As the latter is a probabilistic operation, an operational semantics for a quantum λ-calculus is inherently probabilistic. As in the non-quantum case, probabilistic choice and unbounded recursion yield subtle behaviors.

Fair Coin. Consider the following program L, written in a mock-up ML language with quantum features, similar to the language of [42]:

$$L \coloneqq \texttt{if}\ \ \texttt{meas}(H\texttt{new})\ \texttt{then}\ I\ \texttt{else}\ \Omega.$$

For this introduction, we only describe its behavior informally. The term L above produces a qubit[1] in state $\frac{\sqrt{2}}{2}(|0\rangle + |1\rangle)$ by creating a fresh qubit in state $|0\rangle$ (this is the role of `new`), and applying the Hadamard gate H. Measuring this qubit amounts to flipping a fair coin with equal probability $\frac{1}{2}$. In one case, the program returns the identity function I; otherwise, it diverges – the term Ω stands for the usual, non-terminating looping term.

[1] The reader unfamiliar with the notation should not worry, as the formal details are not essential at this point: just retain that *the state of our qubit is a superposition of two (basis) states*, which play the role of head and tail. When needed, in Section 3 we provide a brief introduction to the mathematical formalism for quantum computation [37].

The program L, therefore, uses the quantum memory only once (at the beginning of the run of the program), and it terminates with probability $\frac{1}{2}$.

Unbounded Use of Fair Coin. In the context of probabilistic behavior, unbounded loops might terminate asymptotically: A program may terminate *with probability* 1, but only at the limit (*almost sure termination*). A simple example suffices to illustrate this point.

Consider a quantum process R that flips a coin by creating and measuring a fresh qubit. If the result is head, the process stops, outputting I. If the result is tail, it starts over. In our mock-up ML, the program R is

$$R \coloneqq \texttt{letrec}\ fx = \big(\texttt{if}\ (\texttt{meas}\ x)\texttt{then}\ I\ \texttt{else}\ f(H\texttt{new})\big)\ \texttt{in}\ f(H\texttt{new}). \tag{1}$$

After n iterations, the program R is in normal form with probability $\frac{1}{2} + \frac{1}{2^2} + \cdots + \frac{1}{2^n}$. Even if the termination probability is 1, this probability of termination is not reached in a finite number of steps but *as a limit*. The program in Equation (1) is our leading example: we formalize it in Example 4.3.

Operational Semantics of Quantum Programs. As it is customary when dealing with *choice effects*, the probabilistic behavior is dealt with by fixing an *evaluation strategy*. Think of tossing a (quantum) coin and duplicating the result, versus tossing the coin twice, which is indeed one key issue at the core of confluence failure in such settings (as observed in [16, 13]). Following the standard approach adopted for functional languages with side effects, the evaluation strategy in quantum λ-calculi such as [42, 38, 12] is a deterministic call-by-value strategy: an argument is reduced to a value before being passed to a function.

> *One aspect that has been seldom examined is however the* properties *of the* general reduction *associated to the quantum lambda-calculus: this is the purpose of this paper.*

A Rewriting Theory for the Quantum λ-Calculus. Lambda calculus has a rich, powerful notion of reduction, whose properties are studied by a vast amount of literature. Such a general *rewriting theory* provides a *sound framework* for reasoning about programs transformations, such as compiler optimizations or parallel implementations, and a base on which to reason about program equivalence. The two fundamental operational properties of lambda calculus are *confluence* and *standardization*. Confluence guarantees that normal forms are unique, standardization that if a normal form exists, there is a strategy that is guaranteed to terminate in such a form.

As pioneered by Plotkin [41], standardization allows to bridge between the general reduction (where programs transformation can be studied), and a specific evaluation strategy, which implements the execution of an idealized programming language. Summarizing the situation, for programming languages, there are two kinds of term rewriting: run-time rewriting ("evaluation") and compile-time rewriting (program transformations).

In the context of quantum lambda-calculi, the only line of research discussing *rewriting* (rather than fixing a deterministic strategy) has been pursued by Dal Lago, Masini, and Zorzi [14, 13]: working with an untyped quantum lambda-calculus, they establish confluence results (and also a form of standardization, but only for the sub-language without measurement [13] – therefore, without the probabilistic behavior).

In this paper, we study not only confluence but also *standardization and normalization results* for a quantum λ-calculus featuring *measurement*, and where β reduction (the engine of λ-calculus) is fully unrestricted. Recent advances in probabilistic and monadic rewriting theory [9, 12, 19, 33, 6, 23, 27, 32] allow us to tackle this task with a formalism and powerful

techniques unavailable a decade ago. Still, quantum rewriting is *more challenging* than probabilistic rewriting because we need to manage the states of the quantum memory. The *design of the language* is, therefore, also delicate: we need to control the duplication of qubits while allowing the full power of β-reduction.

Contributions. We can summarize the contributions of the paper as follows. These are described in more details in Section 5, once all the necessary materials have been set up.

- An untyped quantum lambda-calculus, closely inspired by [14] but re-designed to allow for a more general reduction, now encompassing the full strength of β-reduction; validity constraints make it quantum-compatible.
- The calculus is equipped with a rich operational semantics, which is sound with respect to quantum computation. The *general reduction* enables arbitrary β-reduction; surface reduction (in the spirit of [43] and other calculi based on Linear Logic) plays the role of an *evaluation strategy*.
- We study the rewriting theory for the system, proving *confluence* of the reduction, and *standardization*.
- We obtain a normalization result that scales to the *asymptotic* case, defining a *normalizing strategy* w.r.t. termination at the limit.

Missing proofs and some more technical details are given in the extended version [25].

2 Setting the Scene: the Rewriting Ingredients

This section is devoted to a more detailed (but still informal) discussion of two key elements: the style of λ-calculus we adopt, and what standardization results are about. The calculus is then defined in Section 3, its operational semantics in Section 4; standardization and normalization in the following sections.

Untyped Quantum λ-Calculus. Our quantum calculus is built on top of Simpson's calculus $\Lambda^!$ [43], a variant of untyped λ-calculus inspired by Girard's Linear Logic [29]. In this choice, we follow [14, 13, 12]. Indeed, the fine control of duplication which $\Lambda^!$ inherits from linear logic makes it an ideal base for quantum computation.

The Bang operator ! plays the role of a marker for non-linear management: duplicability and discardability of resources. Abstraction is refined into linear abstraction $\lambda x.M$ and non-linear abstraction $\lambda^! x.M$. The latter allows duplication of the argument, which is required to be suspended as thunk $!N$, behaving as the !-box of linear logic.

▶ **Example 2.1** (duplication, or not). $(\lambda x.Hx)(\mathtt{new})$ is a valid term, but $(\lambda x.\langle x, x\rangle)(\mathtt{new})$ which would duplicate the qubit created by `new` is not. Instead, we can validly write $(\lambda^! x.\mathsf{CNOT}\langle Hx, x\rangle)(!\mathtt{new})$ which thunks `new` and then duplicate it, yielding $\mathsf{CNOT}\langle H\mathtt{new}, \mathtt{new}\rangle$. Notice that this term produces *an entangled pair* of qubits.

In our paper, as well as in [14, 13, 12], surface reduction (*i.e.*, no reduction is allowed in the scope of the ! operator) is the key ingredient to allow for the coexistence of quantum bits with duplication and erasing. Unlike previous work however, in our setting β-reduction – the engine of λ-calculus – is unconstrained. We prove that only quantum operations needs to be surface, making ours a conservative extension of the usual λ-calculus, with its full power.

Table 1 Summarizing Standard Factorization and Normalization Results.

Call-by-name λ-calculus	**Call-by-value λ_v-calculus**	**Linear $\lambda_!$-calculus**
General reduction: $(\to_\beta)$	General reduction: $(\to_{\beta_v})$	General reduction: $(\to_{\beta_!})$
Evaluation: head $(\to_h)$	Evaluation: weak-left $(\to_l)$	Evaluation: surface $(\to_s)$
1. Head factorization:	*1. Weak-left factorization*:	*1. Surface factorization*:
$M \to_\beta^* N$ iff $M \to_h^* \cdot \to_{\neg h}^* N$	$M \to_{\beta_v}^* N$ iff $M \to_l^* \cdot \to_{\neg l}^* N$	$M \to_{\beta_!}^* N$ iff $M \to_s^* \cdot \to_{\neg s}^* N$
2. Head normalization:	*2. Convergence to a value*:	*2. Surface normalization*:
$M \to_\beta^* H$ iff $M \to_h^* H'$	$M \to_{\beta_v}^* V$ iff $M \to_l^* V'$	$M \to_{\beta_!}^* S$ iff $M \to_s^* S'$

Call-by-Value... or rather, Call-by-Push-Value. The reader may have recognized that reduction in our calculus follows the Call-by-Push-Value paradigm, with the Bang operator *thunking* a computation. In fact, Simpson's calculus [43], more precisely the fragment without linear abstraction, is essentially an untyped version of Call-by-Push-Value, and it has been extensively studied in the literature of Linear Logic also with the name of *Bang calculus* [20, 31, 10], as a unifying framework which subsumes both Call-by-Name (CbN) and Call-by-Value(CbV) calculi.

A Taste of Standardization and Normalization: Pure λ-Calculi. Termination and confluence concern the existence and the uniqueness of normal forms, which are the results of a computation. Standardization and normalization are concerned with *how to compute* a given outcome. For example, is there a strategy which guarantees termination, if possible? The desired outcome is generally a specific kind of terms. In the classical theory of λ-calculus (à la Barendregt), the terms of interest are *head normal forms*. In the Call-by-Value approach, the terms of computational interest are values.

Classical λ-calculus. The simplest form of standardization is *factorization*: any reduction sequence can be re-organized so as to first performing specific steps and then everything else. A paradigmatic example is the head factorization theorem of classical λ-calculus (theorem 11.4.6 in [7]): every β-reduction sequence $M \to_\beta^* N$ can be re-organized/factorized so as to first reducing head redexes and then everything else – in symbols $M \to_h^* \cdot \to_{\neg h}^* N$.

A major consequence is *head normalization*, relating arbitrary β reduction with *head* reduction, w.r.t. *head normal forms*, the terms of computational interest in classical λ-calculus. A term M has head normal form if and only if head reduction terminates:

$$M \to_\beta^* H(\text{hnf}) \Leftrightarrow M \to_h^* H'(\text{hnf})$$

Plotkin's Call-by-Value. This kind of results takes its full computational meaning in Plotkin's [41] Call-by-Value λ-calculus The terms of interest are here *values*. Plotkin relates arbitary β reduction $(\to_{\beta_v})$ and the evaluation strategy $\to_l$ which only performs *weak-left* steps (no reduction in the scope of abstractions), by establishing

$$M \to_{\beta_v}^* V(\text{value}) \Leftrightarrow M \to_l^* V'(\text{value})$$

In words: the unconstrained reduction $(\to_{\beta_v})$ returns a value if and only if the evaluation strategy $(\to_l)$ returns a value. The proof relies on a factorization: $M \to_{\beta_v}^* N$ iff $M \to_l^* \cdot \to_{\neg l}^* N$.

Simpson's pure calculus. Standardization and Normalization results have been established by Simpson also for its calculus $\Lambda^!$ [43]. Here, the evaluation strategy is *surface reduction*, *i.e.* no reduction is allowed in the scope of a ! operator.

Summary. The table in Table 1 summarize the factorization and normalization result for the three calculi (respectively based on $\beta, \beta_v, \beta_!$) which we have discussed.

3 Untyped Quantum λ-Calculus

Quantum lambda-calculus is an idealization of functional quantum programming language: following Selinger and Valiron [42], it consists of a regular λ-calculus together with specific constructs for manipulating quantum data and quantum operations. One of the problems consists in accomodating the non-duplicability of quantum information: in a typed setting [42] one can rely on a linear type system. In our untyped setting, we instead base our language on Simpson's λ-calculus [43], extended with constructs corresponding to quantum data and quantum operations.

Due of entanglement, the state of an array of quantum bits cannot be separated into states of individual qubits: the information is *non-local*. A corollary is that quantum data cannot easily be written inside lambda-terms: unlike Boolean values or natural numbers, one cannot put in the grammar of terms a family of constants standing for all of the possible values a quantum bit could take. A standard procedure [42] relies on an external memory with register identifiers used as placeholders for qubits inside the lambda-term. As they stands for qubits, these registers are taken as non-duplicable.

In the original quantum lambda-calculus [42], regular free variables of type qubit were used to represent registers. In this work, being untyped we prefer to consider two kinds of variables: regular term variables, and special variables, called *registers* and denoted by r_i with $i \in \mathbb{N}$, corresponding to the qubit number i in the quantum memory. The language is also equipped with three term constructs to manipulate quantum information. The first term construct is `new`, producing the allocation of a fresh qubit[2]. The second term construct is $\texttt{meas}(r_i, M_0, M_1)$, corresponding to a destructive measurement of the qubit r_i. The evaluation then *probabilistically* continues as M_0 or M_1, depending on the measure being $|0\rangle$ or $|1\rangle$. Finally, assuming that the memory comes with a set of built-in unitary gates ranged over by letters A, B, C, the term U_A corresponds to a function applying the gate A to the indicated qubits.

Raw Terms. Formally, *raw terms* $M, N, P, \ldots$ are built according to the following grammar.

$$M, N, P ::= x \mid !M \mid \lambda x.M \mid \lambda^! x.M \mid MN \mid r_i \mid U_A \mid \texttt{new} \mid \texttt{meas}(P, M, N) \qquad (\textbf{terms } \Lambda_q)$$

where x ranges over a countable set of *variables*, r_i over a disjoint set of *registers* where $i \in \mathbb{N}$ is called the *identifier* of the register, and U_A over a set of build-in n-ary *gates*. In this paper, we limit the arity n to be 1 or 2. Pairs do not appear as a primitive construct, but we adopt the standard encoding, writing $\langle M, N\rangle$ as sugar for $\lambda f.\,(f\ M)\,N$. We also use the shorthand $\langle M_1, \ldots, M_n\rangle$ for $\langle M_1 \langle M_2, \ldots\rangle\rangle$. The variable x is bound in both $\lambda x.P$ and $\lambda^! x.P$. As usual, we silently work modulo α-equivalence. Given a term of shape $\texttt{meas}(P, M, N)$, we call M and N its *branches*. As usual, the set of free variables of a term M are denoted with $\texttt{FV}(M)$. The set of registers identifiers for a term M is denoted with $\texttt{Reg}(M)$.

▶ Remark 3.1. Without any constraints, one could write terms such as $\langle r_0, r_0\rangle$ or $\lambda x.\langle x, x\rangle$. Both are invalid: the former since a qubit cannot be duplicated, the latter since λ-abstractions are meant to be linear in Simpson's calculus.

[2] Unlike the original quantum λ-calculus [42], the term `new` literally evaluates to a qubit.

Terms Validity. To deal with the problem in Remark 3.1, we need to introduce the notions of context and surface context, to speak of occurrences and surface occurrences of subterms.

A context is a term with a hole. We define general *contexts* where the hole can appear anywhere, and *surface contexts* for contexts where holes do not occur in the scope of a ! operator, nor in the branches of a $\texttt{meas}(-,-,-)$. They are generated by the grammars

$$\begin{aligned}\mathbf{C} ::= {} & (\!|\,|\!) \mid M\mathbf{C} \mid \mathbf{C}M \mid \lambda x.\mathbf{C} \mid \lambda^! x.\mathbf{C} \mid \texttt{meas}(\mathbf{C},M,N) \mid && \textbf{(contexts)}\\ & \texttt{meas}(M,\mathbf{C},N) \mid \texttt{meas}(M,N,\mathbf{C}) \mid !\mathbf{C},\\ \mathbf{S} ::= {} & (\!|\,|\!) \mid M\mathbf{S} \mid \mathbf{S}M \mid \lambda x.\mathbf{S} \mid \lambda^! x.\mathbf{S} \mid \texttt{meas}(\mathbf{S},M,N), && \textbf{(surface contexts)}\end{aligned}$$

where $(\!|\,|\!)$ denotes the *hole* of the corresponding context. The notation $\mathbf{C}(\!|R|\!)$ (or $\mathbf{S}(\!|R|\!)$) stands for the term where the only occurrence of a hole $(\!|\,|\!)$ in $\mathbf{C}$ (or $\mathbf{S}$) is replaced with the term R, potentially capturing free variables of R.

Contexts and surface contexts allow us to formalize two notions of occurence of a subterm T. The pair $(\mathbf{C},T)$ (resp. $(\mathbf{S},T)$) is an *occurence* (resp. *surface occurence*) of T in M whever $M=\mathbf{C}(\!|T|\!)$ (resp. $M=\mathbf{S}(\!|T|\!)$). By abuse of notation, we will simply speak of occurrence of a subterm in M, the context being implicit.

We can now define a notion of valid terms, agreeing with the quantum principle of no-cloning.

▶ **Definition 3.2** (Valid Terms, and Linearity)**.** A term M is **valid** whenever

- no register occurs in M more than once, and every occurrences of registers are surface;
- for every subterm $\lambda x.P$ of M, x is *linear* in P, *i.e.* x occurs free exactly once in P and, moreover, this occurrence of x is surface.

▶ Remark 3.3. The validity conditions for registers and linear variables do not allow us to put registers inside branches. So for instance a term such as

$$\lambda z.\,\texttt{meas}(r_0, z\,(U_A\,r_1), z\,(U_B\,r_1)).$$

is invalid in our syntax. This function would measure r_0 and performs an action on r_1 based on the result. If one cannot write such a term directly with the constraints we have set on the language, one can however encode the corresponding behavior as follows:

$$(\lambda^! f.fzr_1)\,\texttt{meas}(r_0, !(\lambda ux.u(U_A x)), !(\lambda ux.u(U_B x)).$$

The action on the register r_1 is the function f whose definition is based on the result of the measurement of r_0.

Quantum Operations. Before diving into the definition of the notion of program, we briefly recall here the mathematical formalism for quantum computation [37].

The basic unit of information in quantum computation is a quantum bit or *qubit*. The state of a single qubit is a normalized vector of the 2-dimensional Hilbert space $\mathbb{C}^2$. We denote the standard basis of $\mathbb{C}^2$ as $\{|0\rangle,|1\rangle\}$, so that the general state of a single qubit can be written as $\alpha|0\rangle+\beta|1\rangle$, where $|\alpha|^2+|\beta|^2=1$.

The basic operations on quantum states are unitary operations and measurements. A *unitary operation* maps an n-qubit state to an n-qubit state, and is described by a *unitary* $2^n\times 2^n$-matrix. We assume that the language provides a set of built-in unitary operations, including for example the *Hadamard gate* H and the *controlled not gate* CNOT:

$$H := \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} \qquad \mathsf{CNOT} := \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix} \tag{2}$$

Measurement acts as a projection. When a qubit $\alpha|0\rangle + \beta|1\rangle$ is measured, the observed outcome is a classical bit: either 0 or 1, with probabilities $|\alpha|^2$ and $|\beta|^2$, respectively. Moreover, the state of the qubit collapses to $|0\rangle$ (resp. $|1\rangle$) if 0 (resp. 1) was observed.

Programs. In order to associate quantum states to registers in lambda-terms, we introduce the notion of program, consisting of a *quantum memory* and a lambda-term of Λ_q. A program is closed under permutation of register identifiers.

The *state* of one qubit is a normalized vector in $\mathcal{E} = \mathbb{C}^2$. The *state* of a quantum memory (also called *qubits state* in the remainder of the document) consisting of n qubits is a normalized vector $\mathtt{Q} \in \mathcal{E}^n = (\mathbb{C}^2)^{\otimes n}$, the n-fold Kronecker product of $\mathcal{E}$. The size of $\mathtt{Q}$ is written $|\mathtt{Q}| := n$. We identify a canonical basis element of $\mathcal{E}^n$ with a string of bits of size n, denoted with $|b_0 \ldots b_{n-1}\rangle$. A state $\mathtt{Q} \in \mathcal{E}^n$ is therefore in general of the form $\mathtt{Q} = \sum_{b_0,\ldots b_{n-1} \in \{0,1\}} \alpha_{b_0 \ldots b_{n-1}} |b_0 \ldots b_{n-1}\rangle$. If σ is a permutation of $\{0, \ldots n-1\}$, we define $\sigma(\mathtt{Q})$ as $\sigma(\mathtt{Q}) = \sum_{b_0,\ldots b_{n-1} \in \{0,1\}} \alpha_{b_0 \ldots b_{n-1}} |b_{\sigma(0)} \ldots b_{\sigma(n-1)}\rangle$.

A *raw program* $\mathbf{p}$ is then a pair $[\mathtt{Q}; M]$, where $\mathtt{Q} \in \mathcal{E}^n$ and where M is a valid term such that $\mathtt{Reg}(M) = \{0, \ldots, n-1\}$. We call n the size of $\mathtt{Q}$ and we denote it with $|\mathtt{Q}|$. If σ is a permutation over the set $\{0..n-1\}$, the *re-indexing* $\sigma(\mathbf{p})$ of $\mathbf{p}$ is the pair $[\sigma(\mathtt{Q}); \sigma(M)]$ where $\sigma(M)$ is M with each occurence of r_i replaced by $r_{\sigma(i)}$.

▶ **Definition 3.4** (Program). A *program* is an equivalence class of raw programs under re-indexing. We identify programs with their representative elements. The set of all programs is denoted with $\mathcal{P}$.

▶ **Example 3.5.** The following two raw programs are equal modulo re-indexing: $[|\psi\rangle \otimes |\phi\rangle; \langle r_0, r_1\rangle] = [|\phi\rangle \otimes |\psi\rangle; \langle r_1, r_0\rangle]$. In both cases, $|\psi\rangle$ is the first qubit in the pair and $|\phi\rangle$ the second one. Re-indexing is agnostic with respect to entanglement, and we also have the raw program $[\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle; \langle r_0, r_1\rangle]$ being a re-indexation of the raw program $[\alpha|00\rangle + \gamma|01\rangle + \beta|10\rangle + \delta|11\rangle; \langle r_1, r_0\rangle]$: they are two representative elements of the same program.

4 Operational Semantics

The operational semantics of λ-calculus is usually formalized by means of a *rewriting system*. In the setting of λ-calculus, the rewriting rules are also known as *reductions*.

Rewriting System. We recall that a *rewriting system* is a pair $(\mathcal{A}, \to)$ consisting of a set $\mathcal{A}$ and a binary relation $\to$ on $\mathcal{A}$ whose pairs are written $t \to s$ and called *reduction steps*. We denote $\to^*$ (resp. $\to^=$) the transitive-reflexive (resp. reflexive) closure of $\to$. We write $t \leftarrow u$ if $u \to t$. If $\to_1, \to_2$ are binary relations on $\mathcal{A}$ then $\to_1 \cdot \to_2$ denotes their composition.

Probabilistic Rewriting. In order to define the operational semantics of the quantum lambda calculus, we need to formalize probabilistic reduction. We do so by means of a rewrite system over *multidistributions*, adopting the *monadic* approach recently developed in the literature of probabilistic rewriting (see *e.g.* [6, 12, 19, 23]). Reduction is here defined not simply on programs, but on (monadic) structures representing probability distributions over programs, called *multidistributions*.

The operational semantics of the language is defined by specifying the probabilistic behavior of programs, then lifting reduction to multidistributions of programs. Let us recall the notions.

Probability Distributions. Given a countable set Ω, a function $\mu\colon \Omega \to [0,1]$ is a probability *subdistribution* if $\|\mu\| := \sum_{\omega\in\Omega} \mu(\omega) \leq 1$ (a *distribution* if $\|\mu\| = 1$). Subdistributions allow us to deal with partial results. We write $\mathtt{D}(\Omega)$ for the set of subdistributions on Ω, equipped with the pointwise order on functions: $\mu \leq \rho$ if $\mu(\omega) \leq \rho(\omega)$ for all $\omega \in \Omega$. $\mathtt{D}(\Omega)$ has a bottom element (the subdistribution $\mathbf{0}$) and maximal elements (all distributions).

Multidistributions. We use multidistributions [6] to *syntactically* represent distributions. A *multidistribution* $\mathtt{m} = \{\!\{ q_i\mathbf{p}_i \}\!\}_{i\in I}$ on the set of programs $\mathcal{P}$ is a finite multiset of pairs of the form $q_i\mathbf{p}_i$, with $q_i \in]0,1]$, $\mathbf{p}_i \in \mathcal{P}$, and $\sum_i q_i \leq 1$. The set of all multidistributions on $\mathcal{P}$ is $\mathtt{MD}(\mathcal{P})$. The sum of two multidistributions is noted $+$, and is simply the union of the multisets. The product $q \cdot \mathtt{m}$ of a scalar q and a multidistribution $\mathtt{m}$ is defined pointwise $q \cdot \{\!\{ p_i\mathbf{p}_i \}\!\}_{i\in I} := \{\!\{ (q\cdot p_i)\mathbf{p}_i \}\!\}_{i\in I}$. We write $\{\!\{ \mathbf{p} \}\!\}$ for $\{\!\{ 1\mathbf{p} \}\!\}$.

4.1 The Rewrite System $\mathcal{Q}$

$\mathcal{Q}$ is the rewrite system $(\mathtt{MD}(\mathcal{P}), \Rightarrow)$ where the relation $\Rightarrow \subseteq \mathtt{MD}(\mathcal{P}) \times \mathtt{MD}(\mathcal{P})$ is monadically defined in two phases. First, we define *one-step reductions* from a program to a multidistribution. For example, if $\mathbf{p}$ is the program $[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle); \mathtt{meas}(r_0, M, N)]$, the program $\mathbf{p}$ reduces in one step to $\{\!\{ \frac{1}{2}[|\rangle; M], \frac{1}{2}[|\rangle; N] \}\!\}$. Then, we *lift* the definition of reduction to a binary relation on $\mathtt{MD}(\mathcal{P})$, in the natural way. So for instance, reusing $\mathbf{p}$ above, $\{\!\{ \frac{1}{2}\mathbf{p}, \frac{1}{2}[\mathtt{Q}; (\lambda x.xx)F] \}\!\}$ reduces in one step to $\{\!\{ \frac{1}{4}[|\rangle; M], \frac{1}{4}[|\rangle; N], \frac{1}{2}[\mathtt{Q}; FF] \}\!\}$. Let us see the details.

I. Programs Reduction. We first define the reduction of a program $\mathbf{p}$ to a multidistribution. The operational behavior of $\mathbf{p}$ is given by beta reduction, denoted with $\to_\beta$, and specific rules for handling quantum operations – the corresponding reduction is denoted with $\to_q$. Formally, the relations $\to_\beta$ and $\to_q$ (also called reduction steps) are subsets of $\mathcal{P} \times \mathtt{MD}(\mathcal{P})$ and are defined in Figure 2 by *contextual closure* of the *root rules* $\mapsto_\beta$ and $\mapsto_q$, given in Figure 1. The relation $\to$ is then the union $\to_\beta \cup \to_q$.

The root rules. They are given in Figure 1. We call *redex* the term on the left-hand side of a rule. Beta rules come in two flavors, the linear one (b), which does not allow for duplication, and the non-linear one (b!), which possibly duplicate boxes (*i.e.* terms of shape $!M$). Quantum rules act on the qubits state, exactly implement the operation which we had informally described in Section 3. Notice that the rule (m) has a probabilistic behaviour. The qubit which has been measured can be discharged (as we actually do).

Contextual Closures. They are defined in Figure 2. Observe that while the β rules are closed under arbitrary contexts $\mathbf{C}$, while the quantum rules are restricted to surface contexts $\mathbf{S}$ (no reduction in the scope of a ! operator, nor in the branches of $\mathtt{meas}(-,-,-)$). This constraints guarantee that qubits are neither duplicated nor delated.

▶ Remark 4.1 (Reindexing). As in [42], reduction is defined on programs, which are equivalence classes. We define the rules on a convenient representative. For example, in Figure 1 rule (u_1) reduces the redex $U_A r_0$. Modulo reindexing, the same rules can be applied to any other register.

II. Lifting. The lifting of a relation $\to_r \subseteq \mathcal{P} \times \mathtt{MD}(\mathcal{P})$ to a relation on multidistributions is defined in Figure 3. In particular, $\to, \to_\beta, \to_q$, lift to $\Rightarrow, \Rightarrow_\beta, \Rightarrow_q$, respectively.

Reduction Sequences. A $\Rightarrow$-sequence (or *reduction sequence*) from $\mathtt{m}$ is a sequence $\mathtt{m} = \mathtt{m}_0, \mathtt{m}_1, \mathtt{m}_2, \ldots$ such that $\mathtt{m}_i \Rightarrow \mathtt{m}_{i+1}$ for every i. We write $\mathtt{m}_0 \Rightarrow^* \mathtt{m}$ to indicate the existence of a finite reduction sequence from $\mathtt{m}_0$, and $\mathtt{m}_0 \Rightarrow^k \mathtt{m}$ to specify the number k of $\Rightarrow$-steps. Given a program $\mathbf{p}$ and $\mathtt{m}_0 = \{\!\{ \mathbf{p} \}\!\}$, the sequence $\mathtt{m}_0 \Rightarrow \mathtt{m}_1 \Rightarrow \cdots$ naturally models the evaluation of $\mathbf{p}$; each $\mathtt{m}_k$ expresses the "expected" state of the system after k steps.

β rules

(b) $[\mathtt{Q}; (\lambda x.M)N] \mapsto_\beta \{\!\{ [\mathtt{Q}; M\{N/x\}] \}\!\}$
$(!b)$ $[\mathtt{Q}; (\lambda^! x.M)!N] \mapsto_\beta \{\!\{ [\mathtt{Q}; M\{N/x\}] \}\!\}$

Quantum rules

(n) $[\mathtt{Q}; \mathtt{new}] \mapsto_q \{\!\{ [\mathtt{Q} \otimes |0\rangle; r_n] \}\!\}$ where $|\mathtt{Q}| = n$
(m) $[\mathtt{Q}; \mathtt{meas}(r_n, M, N)] \mapsto_q \{\!\{ |\alpha_0|^2[\mathtt{Q}_0; M], |\alpha_1|^2[\mathtt{Q}_1; N] \}\!\}$ where $\mathtt{Q} = \alpha_0\mathtt{Q}_0 \otimes |0\rangle + \alpha_1\mathtt{Q}_1 \otimes |1\rangle$ and Q has $n+1$ qubits
(u_1) for A unary operator: $[\mathtt{Q}; U_A\, r_0] \mapsto_q \{\!\{ [\mathtt{Q}'; r_0] \}\!\}$ where $\mathtt{Q}'$ is $(A \otimes \mathsf{Id})\mathtt{Q}$
(u_2) for A binary operator: $[\mathtt{Q}; (U_A\, \langle r_0, r_1\rangle)] \mapsto_q \{\!\{ [\mathtt{Q}'; \langle r_0, r_1\rangle] \}\!\}$ where $\mathtt{Q}'$ is $(A \otimes \mathsf{Id})\mathtt{Q}$.

Figure 1 Root rules ($\mapsto$).

β steps

$$\frac{[\mathtt{Q}; M] \mapsto_\beta \{\!\{ [\mathtt{Q}; M'] \}\!\}}{[\mathtt{Q}; \mathbf{C}(\!|M|\!)] \to_\beta \{\!\{ [\mathtt{Q}; \mathbf{C}(\!|M'|\!)] \}\!\}}$$

Quantum steps

$$\frac{[\mathtt{Q}; M] \mapsto_q \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}}{[\mathtt{Q}; \mathbf{S}(\!|M|\!)] \to_q \{\!\{ p_i[\mathtt{Q}_i; \mathbf{S}(\!|M_i|\!)] \}\!\}}$$

$$\to := \to_\beta \cup \to_q$$

Figure 2 Contextual closure of root rules: ($\to$).

Validity. Validity of terms is preserved:

▶ **Proposition 4.2.** *If M is a valid term, and $[\mathtt{Q}; M] \to \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}$, then M_i is a valid term.*

Notice that the restriction of $\to_q$ to surface contexts is necessary to respect the linearity of quantum computation, avoiding duplication or deletion of qubits.

Examples. Let us see the definitions at work in a few examples. We first formalize the recursive program from Equation (1). Recall that H is the Hadamar gate, and $I := \lambda x.x$.

▶ **Example 4.3** (Flipping the quantum coin)**.** The program in Equation (1) can be written as $\mathbf{p} := [\,|\rangle; \Delta!\Delta]$, where $|\rangle$ is just the empty memory and $\Delta := \lambda^! x.\, \mathtt{meas}(H\mathtt{new}, I, x!x)$. A reduction from $\mathbf{p}$ behaves as follows. At every reduction step, we underline the redex.

$$\begin{aligned}\{\!\{ [\,|\rangle; \underline{\Delta!\Delta}] \}\!\} &\Rightarrow \{\!\{ [\,|\rangle; \mathtt{meas}(U_H\underline{\mathtt{new}}, I, \Delta!\Delta)] \}\!\} \Rightarrow \{\!\{ [\,|0\rangle; \mathtt{meas}(\underline{U_H r_0}, I, \Delta!\Delta)] \}\!\} \\ &\Rightarrow \{\!\{ [\,\frac{\sqrt{2}}{2}(|0\rangle + |1\rangle); \underline{\mathtt{meas}(r_0, I, \Delta!\Delta)}] \}\!\} \Rightarrow \{\!\{ \tfrac{1}{2}[|\rangle; I], \tfrac{1}{2}[|\rangle; \Delta!\Delta] \}\!\} \\ &\Rightarrow \ldots \Rightarrow \{\!\{ \tfrac{1}{2}[|\rangle; I], \tfrac{1}{4}[|\rangle; I], \tfrac{1}{4}[|\rangle; \Delta!\Delta] \}\!\} \Rightarrow \ldots\end{aligned}$$

Notice that the first step is a non-linear β reduction. The reduction of `new` allocates a fresh qubit in the memory, corresponding to the register r_0. The redex $U_H r_0$ applies the Hadamar gate H to that qubit. The last reduction performs *measurement*, yielding a probabilistic outcome.

▶ **Example 4.4** (Entangled pair)**.** Let $\mathbf{p} := [|\rangle; \mathtt{let}\ \langle x, y\rangle = U_{\mathsf{CNOT}}\langle U_H\mathtt{new}, \mathtt{new}\rangle\ \mathtt{in}\ \mathtt{meas}(y, I, I)x]$ (where $\mathtt{let}\ \langle x, y\rangle \ldots$ is sugar for an opportune encoding). This program produces an entangled pair of qubits (notice how CNOT is applied to a pair of registers) and then measures one of the qubits. Let us formalize its behaviour:

$$\begin{aligned}\{\!\{ \mathbf{p} \}\!\} &\Rrightarrow_{\mathsf{s}}^* \{\!\{ [\frac{\sqrt{2}}{2}(|0\rangle + |1\rangle) \otimes |0\rangle; \mathtt{let}\ \langle x, y\rangle = \underline{U_{\mathsf{CNOT}}\langle r_0, r_1\rangle}\ \mathtt{in}\ \mathtt{meas}(y, I, I)x] \}\!\} \\ &\Rrightarrow_{\mathsf{s}} \{\!\{ [\frac{\sqrt{2}}{2}|00\rangle + \frac{\sqrt{2}}{2}|11\rangle; \mathtt{let}\ \langle x, y\rangle = \langle r_0, r_1\rangle\ \mathtt{in}\ \mathtt{meas}(y, I, I)x] \}\!\} \\ &\Rrightarrow_{\mathsf{s}}^* \{\!\{ [\frac{\sqrt{2}}{2}|00\rangle + \frac{\sqrt{2}}{2}|11\rangle; \mathtt{meas}(r_1, I, I)r_0] \}\!\} \Rrightarrow_{\mathsf{s}} \{\!\{ \tfrac{1}{2}[|0\rangle; Ir_0], \tfrac{1}{2}[|1\rangle; Ir_0] \}\!\}\end{aligned}$$

$$\frac{}{\{\!\{ \mathbf{p} \}\!\} \Rightarrow \{\!\{ \mathbf{p} \}\!\}} \quad \frac{\mathbf{p} \to \mathtt{m}}{\{\!\{ \mathbf{p} \}\!\} \Rightarrow \mathtt{m}} \quad \frac{(\{\!\{ \mathbf{p}_i \}\!\} \Rightarrow \mathtt{m}_i)_{i\in I}}{\{\!\{ p_i\mathbf{p}_i \mid i \in I \}\!\} \Rightarrow \sum_{i\in I} p_i \cdot \mathtt{m}_i}$$

Figure 3 Lifting of $\to$.

4.2 Surface Reduction and Surface Normal Forms

So far, we have defined a very liberal notion of reduction, in which β is unrestricted – it can validly be performed even inside a !-box. What shall we adopt as evaluation strategy?

In the setting of calculi based on linear logic, as Simpson's calculus [43] and the Bang calculus [20], the natural candidate is **surface reduction**: the restriction of *beta* to surface contexts ($\underset{s}{\to}_\beta$) plays a role akin to that of head reduction in classical λ-calculus, yielding to similar factorization and normalization results which relate $\to_\beta$ and $\underset{s}{\to}_\beta$ (as recalled in Table 1). The *terms of interest* are here **surface normal forms** (snf), such as x or $!M$. They are the analog of values in Plotkin's Call-by-Value λ-calculus and of head normal forms in classical λ-calculus – such an analogy can indeed be made precise [20, 31, 10][3].

In our setting, *surface reduction* and *surface normal forms*(snf) also play a privileged role.

Surface Reduction. Surface steps $\underset{s}{\to} \subseteq \mathcal{P} \times \mathtt{MD}(\mathcal{P})$ (Figure 5) are the union $\to_q \cup \underset{s}{\to}_\beta$ of quantum steps together with $\underset{s}{\to}_\beta$, *i.e.* the closure *under surface contexts* $\mathbf{S}$ of the β rules. A program $\mathbf{p}$ is a **surface normal form** (snf) if $\mathbf{p} \not\underset{s}{\to}$, *i.e.* no surface reduction is possible from it.

A $\to$-step which is not surface is noted $\underset{\neg s}{\to}$. The lifting of $\underset{s}{\to}, \underset{\neg s}{\to}$ to relations on multidistributions is denoted $\underset{s}{\Rightarrow}, \underset{\neg s}{\Rightarrow}$ respectively.

▶ Remark 4.5. Notice that $\underset{\neg s}{\to}$ steps do not act on the qubits state, since they are β steps.

Strict Lifting. To guarantee normalization results (Section 7), we will need a stricter form of lifting, noted $\underset{s}{\rightrightarrows}$ (Figure 4), forcing a reduction step to be performed in each program of the multidistribution $\mathtt{r}$, if a redex exists. Clearly $\underset{s}{\rightrightarrows} \subseteq \underset{s}{\Rightarrow}$.

$$\frac{\mathbf{p} \not\underset{s}{\to}}{\{\!\{ \mathbf{p} \}\!\} \underset{s}{\rightrightarrows} \{\!\{ \mathbf{p} \}\!\}} \quad \frac{\mathbf{p} \underset{s}{\to} \mathtt{m}}{\{\!\{ \mathbf{p} \}\!\} \underset{s}{\rightrightarrows} \mathtt{m}} \quad \frac{(\{\!\{ \mathbf{p}_i \}\!\} \underset{s}{\rightrightarrows} \mathtt{m}_i)_{i\in I}}{\{\!\{ p_i\mathbf{p}_i \mid i \in I \}\!\} \underset{s}{\rightrightarrows} \sum_{i\in I} p_i \cdot \mathtt{m}_i}$$

Figure 4 Strict lifting of $\underset{s}{\to}$.

▶ **Example 4.6.** We will prove that the strict lifting $\underset{s}{\rightrightarrows}$ guarantees to reach snf, if any exist. This is obviously not the case for $\underset{s}{\Rightarrow}$-sequences:

$$\{\!\{ \tfrac{1}{2} I\mathtt{new}, \tfrac{1}{2}\underline{(\lambda^! x.x!x)!(\lambda^! x.x!x)} \}\!\} \underset{s}{\Rightarrow} \{\!\{ \tfrac{1}{2} I\mathtt{new}, \tfrac{1}{2}\underline{(\lambda^! x.x!x)!(\lambda^! x.x!x)} \}\!\} \underset{s}{\Rightarrow} \{\!\{ \tfrac{1}{2} I\mathtt{new}, \tfrac{1}{2}\underline{(\lambda^! x.x!x)!(\lambda^! x.x!x)} \}\!\} \underset{s}{\Rightarrow} \dots$$

[3] A consequence of Girard's translation of Call-by-Name and Call-by-Value λ-calculi into Linear Logic.

On the Interest of Surface Normal Forms. What is the result of running a quantum program? In general, since computation is probabilistic, the result of executing a program will be a distribution over some outcomes of interest. A natural choice are programs of shape $\mathbf{p} \coloneqq [\mathtt{Q}; S]$, with S in surface normal form, ensuring that at this point, the qubits state $\mathtt{Q}$ is a *stable piece of information* (it will not further evolve in the computation). Indeed:

$$\text{a program } \mathbf{p} \not\xrightarrow[\mathsf{s}]{} \text{ (i.e. in } \mathsf{snf}\text{) will no longer modify the qubits state.}$$

▶ Remark 4.7. Notice instead that a program $\mathbf{p} \not\to_q$ (no quantum step is possible) is not necessarily done in manipulating the quantum memory. Further β reductions may unblock further quantum steps. Think of $(\lambda^! x.\mathsf{CNOT}\langle Hx, x\rangle)(!\mathtt{new})$ from Example 2.1.

4.3 Sum-up Tables

Let us conclude the section summarizing the reduction relations at play.

Relations.

$\mathcal{P} \times \mathtt{MD}(\mathcal{P})$	Definition	Lifted to $\mathtt{MD}(\mathcal{P}) \times \mathtt{MD}(\mathcal{P})$	Strict lifting
$\to_\beta$	contextual closure of β-rules	$\Rightarrow_\beta$	
$\xrightarrow[\mathsf{s}]{}_\beta$	closure by **surface context** of β-rules	$\underset{\mathsf{s}}{\Rightarrow}_\beta$	$\underset{\mathsf{s}}{\rightrightarrows}_\beta$
$\to_q$	closure by **surface context** of q-rules	$\Rightarrow_q$	$\rightrightarrows_q$
$\to$	$\to_\beta \cup \to_q$	$\Rightarrow$	
$\xrightarrow[\mathsf{s}]{}$	$\xrightarrow[\mathsf{s}]{}_\beta \cup \to_q$	$\underset{\mathsf{s}}{\Rightarrow}$	$\underset{\mathsf{s}}{\rightrightarrows}$
$\xrightarrow[\neg\mathsf{s}]{}$	$\to - \xrightarrow[\mathsf{s}]{}$	$\underset{\neg\mathsf{s}}{\Rightarrow}$	

Reduction Sequences.

Finite reduction sequence	
$\mathtt{m} \Rightarrow^* \mathtt{n}$	there is a $\Rightarrow$-sequence from $\mathtt{m}$ to $\mathtt{n}$
$\mathtt{m} \underset{\mathsf{s}}{\Rightarrow}^* \mathtt{n}$	there is a $\underset{\mathsf{s}}{\Rightarrow}$-sequence from $\mathtt{m}$ to $\mathtt{n}$
$\mathtt{m} \underset{\mathsf{s}}{\rightrightarrows}^* \mathtt{n}$	there is a $\underset{\mathsf{s}}{\rightrightarrows}$-sequence from $\mathtt{m}$ to $\mathtt{n}$

5 Rewriting Theory for $\mathcal{Q}$: Overview of the Results

We are now going to study reduction on multidistributions of programs, namely the *general reduction* $\Rightarrow$ (corresponding to the lifting of $\to$) and *surface reductions* (corresponding to the lifting of $\xrightarrow[\mathsf{s}]{}$), and the relation between the two. Let us discuss each point.

1. The reduction $\Rightarrow$ allows for *unrestricted* β reduction. For example, we can rewrite in the scope of a Bang operator ! (perhaps to optimize the thunked code before copying it several times). We prove that $\Rightarrow$ is confluent, providing a general framework for rewriting theory. This (very liberal) reduction has a foundational role, in which to study the equational theory of the calculus and to analyze programs transformations.
2. Surface reduction $\underset{\mathsf{s}}{\Rightarrow} \subseteq \Rightarrow$ plays the role of an evaluation strategy, in which however the scheduling (how redexes should be fired) is not fully specified[4]. For example $\mathbf{p} = [|\rangle; \langle \mathtt{new}, H\mathtt{new}\rangle]$ has two surface redexes, enabling two different steps. We will prove (by

[4] This is not only convenient, as it allows for parallel implementation, but it is necessary for standardization [26]

proving a diamond property) that surface reduction ($\rightrightarrows_s$) is "essentially deterministic" in the sense that while the choice of the redex to fire is non-deterministic, the order in which such choices are performed are irrelevant to the final result.

3. The two reductions are related by a standardization result (Theorem 6.7) stating that if $\mathtt{m} \Rightarrow^* \mathtt{n}$ then $\mathtt{m} \Rightarrow_s^* \cdot \rightrightarrows_{\neg s}^* \mathtt{n}$. Standardization is the base of normalization results, concerning properties such as "program $\mathbf{p}$ terminates with probability p."
4. We prove that $\rightrightarrows_s$ is a *normalization strategy* for $\Rightarrow$, namely if $\mathbf{p}$ *may* converge to surface normal form with probability p using the general reduction $\Rightarrow$, then $\rightrightarrows_s$ reduction *must* converge to surface normal form with probability p. Informally, we can write that $\mathtt{m} \Downarrow p$ implies $\mathtt{m} \Downarrow_s p$ (corresponding to the last line in Table 1). To formalize and prove such a claim we will need more tools, because probabilistic termination is *asymptotic*, *i.e.* it appears as a limit of a possibly infinite reduction. We treat this in Section 7, where we rely on techniques from [2, 23, 24].

6 Confluence and Finitary Standardization

We first recall standard notions which we are going to use.

Confluence, Commutation, and all That (a quick recap). The relation $\rightarrow$ is *confluent* if $\leftarrow^* \cdot \rightarrow^* \subseteq \rightarrow^* \cdot \leftarrow^*$. A stricter form is the diamond $\leftarrow \cdot \rightarrow$ implies $\rightarrow \cdot \leftarrow$, which is well known to imply confluence. Two relations $\rightarrow_\circ$ and $\rightarrow_\bullet$ on A *commute* if : $\leftarrow_\circ^* \cdot \rightarrow_\bullet^*$ implies $\rightarrow_\bullet^* \cdot \leftarrow_\circ^*$. Confluence and factorization are both commutation properties: a relation is confluent if it commutes with itself.

An element $u \in \mathcal{A}$ is a $\rightarrow$-*normal form* if there is no t such that $u \rightarrow t$ (written $u \not\rightarrow$).

On normalization. In general, a term may or may not reduce to a normal form. And if it does, not all reduction sequences necessarily lead to normal form. How do we compute a normal form? This is the problem tackled by *normalization*: by repeatedly performing *only specific steps*, a normal form will be computed, provided that t can reduce to any. Intuitively, a *normalizing strategy* for $\rightarrow$ is a reduction strategy which, given a term t, is guaranteed to reach normal form, if any exists.

6.1 Surface Reduction has the Diamond Property

In this section, we first prove that surface reduction ($\Rightarrow_s$ and $\rightrightarrows_s$) has the *diamond property*:

$$\mathtt{r} \Leftarrow_s \mathtt{m} \Rightarrow_s \mathtt{s} \text{ implies } \mathtt{r} \Rightarrow_s \mathtt{n} \Leftarrow_s \mathtt{s} \text{ (for some } \mathtt{n}) \qquad \text{(Diamond)}$$

then we show that $\Rightarrow$ is confluent.

Here we adapt techniques used in probabilistic rewriting [6, 22, 26]. Proving the diamond property is however significantly *harder* than in the case of probabilistic λ-calculi, because we need to take into account also the *qubits state*, and the corresponding registers. If a program $\mathbf{p} = [\mathtt{Q}; M]$ has two different reductions, we need to join in one step not only the terms, but also their qubits states, working up to re-indexing of the registers (recall that programs are equivalence classes modulo re-indexing, see also Example 3.5). The following is an example, just using the simple construct `new`. Measurement makes the situation even more delicate.

▶ **Example 6.1.** Let $\mathbf{p} = [|\rangle; \langle \mathtt{new}, (H\mathtt{new})\rangle]$. The following are two different reduction sequences form $\mathbf{p}$. The two normal forms are the same program (Definition 3.4). Here, $|+\rangle := \frac{\sqrt{2}}{2}(|0\rangle + |1\rangle)$.

$$[|\rangle; \langle \underline{\texttt{new}}, (H\texttt{new})\rangle] \to_{\texttt{s}} \{\!\{ [|0\rangle; \langle r_0, (H\underline{\texttt{new}})\rangle] \}\!\} \rightrightarrows_{\texttt{s}} \{\!\{ [|0\rangle; \langle r_0, (\underline{Hr_1})\rangle] \}\!\} \rightrightarrows_{\texttt{s}} \{\!\{ [|0\rangle \otimes |+\rangle; \langle r_0, r_1\rangle] \}\!\}$$

$$[|\rangle; \langle \texttt{new}, (\underline{H\texttt{new}})\rangle] \to_{\texttt{s}} \{\!\{ [|0\rangle; \langle \texttt{new}, (\underline{Hr_0})\rangle] \}\!\} \rightrightarrows_{\texttt{s}} \{\!\{ [|+\rangle; \langle \underline{\texttt{new}}, (r_0)\rangle] \}\!\} \rightrightarrows_{\texttt{s}} \{\!\{ [|+\rangle \otimes |0\rangle; \langle r_1, r_0\rangle] \}\!\}$$

The key result is the following version of diamond (commutation). The proof – quite technical – is given in the extended version [25]. Recall that $\rightrightarrows_{\texttt{s}} \subseteq \Rightarrow_{\texttt{s}}$.

▶ **Lemma 6.2** (Pointed Diamond). *Assume* $\mathbf{p} = [\mathtt{Q}; M]$ *and that* M *has two distinct redexes, such that* $\mathbf{p} \to_{\texttt{s}} \mathtt{m}_1$ *and* $\mathbf{p} \to_{\texttt{s}} \mathtt{m}_2$. *Then there exists* $\mathtt{n}$ *such that* $\mathtt{m}_1 \rightrightarrows_{\texttt{s}} \mathtt{n}$ *and* $\mathtt{m}_2 \rightrightarrows_{\texttt{s}} \mathtt{n}$. *Moreover, no term* M_i *in* $\mathtt{m}_1 = \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}_{i\in I}$ *and no term* M_j *in* $\mathtt{m}_2 = \{\!\{ p_j[\mathtt{Q}_j; M_j] \}\!\}_{j\in J}$ *is in* snf.

From the above result we obtain the diamond property.

▶ **Proposition 6.3** (Diamond). *Surface reductions* $\Rightarrow_{\texttt{s}}$ *and* $\rightrightarrows_{\texttt{s}}$ *have the diamond property.*

In its stricter form, the diamond property guarantees that the non determinism in the choice of the redex is *irrelevant* – hence the reduction $\rightrightarrows_{\texttt{s}}$ is essentially deterministic. The technical name for this property is Newman's *random descent* [36]: no matter the choice of the redex, all reduction sequences behave the same way, *i.e.* have the same length, and if terminating, they do so in the same normal form. Formalized by Theorem 7.3, we use this fact to establish that $\rightrightarrows_{\texttt{s}}$ is a normalizing strategy for $\Rightarrow$.

6.2 Confluence of ⇒

We modularize the proof of confluence by using a classical technique, Hindley-Rosen lemma, stating that if $\Rightarrow_1$ and $\Rightarrow_2$ are binary relations on the same set $\mathcal{R}$, then their union $\Rightarrow_1 \cup \Rightarrow_2$ is confluent if both $\Rightarrow_1$ and $\Rightarrow_2$ are confluent, and $\Rightarrow_1$ and $\Rightarrow_2$ commute.

▶ **Theorem 6.4.** *The reduction* $\Rightarrow$ *satisfies confluence.*

Proof. The proof that $\Rightarrow_\beta \cup \Rightarrow_q$ is confluent, is easily obtained from Lemma 6.2, by using Hindley-Rosen Lemma. We already have most of the elements: $\Rightarrow_\beta$ is confluent: because $\to_\beta$ is; $\Rightarrow_q$ is confluent: because it is diamond (Proposition 6.3); $\Rightarrow_q$ and $\Rightarrow_\beta$ commute: by Lemma 6.2, we already know that $\Rightarrow_q$ and $\Rightarrow_{\texttt{s}\beta}$ commute, hence we only need to verify that $\Rightarrow_q$ and $\rightrightarrows_{\neg\texttt{s}\beta}$ commute, which is easily done. ◀

6.3 Surface Standardization

We show that any sequence $\Rightarrow^*$ can be factorized as $\Rightarrow_{\texttt{s}}^* \cdot \rightrightarrows_{\neg\texttt{s}}^*$ (Theorem 6.7). Standardization is proved via the modular technique proposed in [1], which in our notation can be stated as follows:

▶ **Lemma 6.5** (Modular Factorization [1]). $\Rightarrow^* \subseteq \Rightarrow_{\texttt{s}}^* \cdot \rightrightarrows_{\neg\texttt{s}}^*$ *if the following conditions hold:*

1. $\Rightarrow_\beta^* \subseteq \Rightarrow_{\texttt{s}\beta}^* \cdot \rightrightarrows_{\neg\texttt{s}\beta}^*$, *and*
2. $\rightrightarrows_{\neg\texttt{s}\beta} \cdot \Rightarrow_{\texttt{s}q} \subseteq \Rightarrow_{\texttt{s}q} \cdot \Rightarrow_\beta$.

Condition 1. in Lemma 6.5 is immediate consequence of Simpson's surface standardization for the $\Lambda^!$ calculus [43] stating that $\to_\beta^* \subseteq \to_{\texttt{s}\beta}^* \cdot \to_{\neg\texttt{s}\beta}^*$. Condition 2. in Lemma 6.5 is obtained from the following *pointed* version:

▶ **Lemma 6.6.** $[\![\mathtt{Q}; M]\!] \rightrightarrows_{\neg \mathsf{s}}^{\beta} \{\!\{ [\![\mathtt{Q}; P]\!] \}\!\}$ *and* $[\![\mathtt{Q}; P]\!] \to_q \mathtt{n}$ *implies* $[\![\mathtt{Q}; M]\!] \to_q \cdot \Rightarrow_\beta \mathtt{n}$.

Proof. By induction on the context $\mathbf{S}$ such that $P = \mathbf{S}(\!|R|\!)$ and $[\![\mathtt{Q}; \mathbf{S}(\!|R|\!)]\!] \to_q \{\!\{ p_i[\![\mathtt{Q}_i; \mathbf{S}(\!|R_i|\!)]\!] \}\!\} = \mathtt{n}$. We exploit in an essential way the fact that M and P have the same shape. ◀

By Lemmas 6.5 and 6.6, we obtain the main result of this section:

▶ **Theorem 6.7** (Surface Standardization). $\mathtt{m} \Rightarrow^* \mathtt{n}$ *implies* $\mathtt{m} \underset{\mathsf{s}}{\Rightarrow}^* \cdot \underset{\neg \mathsf{s}}{\Rightarrow}^* \mathtt{n}$

▶ Remark 6.8 (Strict vs non-strict). Please observe that standardization is stated in terms of the *non-strict* lifting ($\underset{\mathsf{s}}{\Rightarrow}$) of $\underset{\mathsf{s}}{\to}$, as $\underset{\mathsf{s}}{\rightrightarrows}$ could reduce more than what is desired. Dually, normalization holds in terms of the *strict* lifting $\underset{\mathsf{s}}{\rightrightarrows}$, for the reasons already discussed in Example 4.6.

A Reading of Surface Standardization. A program $\mathbf{p}$ in snf will no longer modify the qubits state. Intuitively, $\mathbf{p}$ has already produced the maximal amount of quantum data that it could possibly do. We can read Surface Standardization as follows. Assume $\{\!\{ \mathbf{p} \}\!\} \Rightarrow^* \mathtt{n}$ where all terms in $\mathtt{n}$ are in snf (we use metavariables S_i, S_i' to indicate terms in snf). Standardization guarantees that surface steps suffice to reach a multidistribution $\mathtt{n}'$ whose programs have the exact *same information content* as $\mathtt{n}$:

$$\{\!\{ \mathbf{p} \}\!\} \Rightarrow^* \mathtt{n} = \{\!\{ p_i[\![\mathtt{Q}_i; S_i]\!] \}\!\}_{i \in I} \quad \text{implies} \quad \{\!\{ \mathbf{p} \}\!\} \underset{\mathsf{s}}{\Rightarrow}^* \mathtt{n}' = \{\!\{ p_i[\![\mathtt{Q}_i; S_i']\!] \}\!\}_{i \in I}.$$

This because Theorem 6.7 implies $\{\!\{ \mathbf{p} \}\!\} \underset{\mathsf{s}}{\Rightarrow}^* \mathtt{n}' \underset{\neg \mathsf{s}}{\Rightarrow}^* \mathtt{n}$, and from $\mathtt{n}' \underset{\neg \mathsf{s}}{\Rightarrow}^* \mathtt{n}$ we deduce that each element $p_i[\![\mathtt{Q}_i; S_i]\!]$ in $\mathtt{n}$ must come form an element $p_i[\![\mathtt{Q}_i; S_i']\!]$ in $\mathtt{n}'$ where S_i' is in snf and where the qubits state $\mathtt{Q}_i$ (and the associated probability p_i) are *exactly the same*.

7 Probabilistic Termination and Asymptotic Normalization

What does it mean for a program to reach surface normal form (snf)? Since measurement makes the reduction probabilistic, we need to give a quantitative answer.

Probabilistic Termination. The probability that the system described by the multidistribution $\mathtt{m} = \{\!\{ p_i[\![\mathtt{Q}_i; M_i]\!] \mid i \in I \}\!\}$ is in surface normal form is expressed by a scalar $p = \mathbb{P}(\mathtt{m}) \in [0, 1]$ which is defined as follows:

$$\mathbb{P}(\mathtt{m}) = \sum_{i \in I} q_i \qquad q_i = \begin{cases} p_i & \text{if } M_i \ \mathsf{snf} \\ 0 & \text{otherwise} \end{cases}$$

Let $\mathbf{p} = [\![\mathtt{Q}; M]\!]$ and $\mathtt{m}_0 = \{\!\{ \mathbf{p} \}\!\}$. Let $\mathtt{m}_0 \Rightarrow \mathtt{m}_1 \Rightarrow \mathtt{m}_2 \Rightarrow \cdots$ a reduction sequence. $\mathbb{P}(\mathtt{m}_k)$ expresses the probability that after k steps $\mathbf{p}$ is in snf. The *probability that* $\mathbf{p}$ *reaches* snf along the (possibly infinite) reduction sequence $\langle \mathtt{m}_n \rangle_{n \in \mathbb{N}}$ is easily defined as a limit: $\sup_n \{\mathbb{P}(\mathtt{m}_n)\}$. We also say that the sequence $\langle \mathtt{m}_n \rangle_{n \in \mathbb{N}}$ *converges* with probability $\sup_n \{\mathbb{P}(\mathtt{m}_n)\}$.

▶ **Example 7.1** (Recursive coin, cont.). Consider again Example 4.3. After 4 steps, the program terminates with probability $\frac{1}{2}$. After 4 more steps, it terminates with probability $\frac{1}{2} + \frac{1}{4}$, and so on. At the limit, the reduction sequence *converges* with probability $\sum_{k:1}^{\infty} \frac{1}{2^k} = 1$.

Table 2 Limit of (possibly infinite) reduction sequences.

Convergence (Def.n7.2)	
$\mathtt{m} \Downarrow p$	there is a $\Rightarrow$-sequence from $\mathtt{m}$ which converges with probability p
$\mathtt{m} \Downarrow_s p$	there is a $\underset{s}{\Rightarrow}$-sequence from $\mathtt{m}$ which converges with probability p
$\mathtt{m} \downdownarrows_s p$	there is a $\underset{s}{\rightrightarrows}$-sequence from $\mathtt{m}$ which converges with probability p

7.1 Accounting for Several Possible Reduction Sequences

Since $\Rightarrow$ is not a deterministic reduction, given a multidistribution $\mathtt{m}$, there are *several possible reduction sequences* from $\mathtt{m}$, and therefore several outcomes (limits) are possible. Following [23], we adopt the following terminology:

▶ **Definition 7.2** (Limits). *Given* $\mathtt{m}$*, we write*

- $\mathtt{m} \Downarrow p$*, if there exists a* $\Rightarrow$*-sequence* $\langle \mathtt{m}_n \rangle_{n\in\mathbb{N}}$ *from* $\mathtt{m}$ *whose limit is* p*.*
- $\mathtt{Lim}(\mathtt{m}, \Rightarrow) \coloneqq \{p \mid \mathtt{m} \Downarrow p\}$ *is the set of limits of* $\mathtt{m}$*.*
- $[\![\mathtt{m}]\!]$ *denotes the greatest element of* $\mathtt{Lim}(\mathtt{m}, \Rightarrow)$*, if any exists.*

Intuitively, $[\![\mathbf{p}]\!]$ is the *best result* that any $\Rightarrow$-sequence from $\mathbf{p}$ can *effectively* produce. If the set $\mathtt{Lim}(\mathbf{p}, \Rightarrow)$ has a sup α but not a greatest element (think of the open interval $[0, 1)$), it means that in fact, no reduction can produce α as a limit. Notice also that, when reduction is deterministic, from any $\mathbf{p}$ there is only one maximal reduction sequence, and so it is always the case that $[\![\mathbf{p}]\!] = \sup_n\{\mathbb{P}(\mathbf{p}_n)\}$. Below we exploit the interplay between different rewriting relations, and their limit; it is useful to summarize our notations in Table 2.

7.2 Asymptotic Normalization

Given a quantum program $\mathbf{p}$, does $[\![\mathbf{p}]\!]$ exists? If this is the case, can we define a *normalizing strategy* which is guarantee to converge to $[\![\mathbf{p}]\!]$? The answer is positive. The main result of this section is that such a normalizing strategy does exist, and it is $\underset{s}{\rightrightarrows}$. More precisely, we show that *any* $\underset{s}{\rightrightarrows}$-reduction sequence from $\mathbf{p}$ converges to the same limit, which is exactly $[\![\mathbf{p}]\!]$. We establish the following results, for any arbitrary $\mathtt{m} \in \mathtt{MD}(\mathcal{P})$. Theorem 7.3 is a direct – and the most important – consequence of the diamond property of $\underset{s}{\rightrightarrows}$. The proof uses both point 1. and point 2. of Lemma 6.2. For Theorem 7.4, the proof relies on an abstract technique from [24].

▶ **Theorem 7.3** (Random Descent). *All* $\underset{s}{\rightrightarrows}$*-sequences from* $\mathtt{m}$ *converge to the same limit.*

▶ **Theorem 7.4** (Asymptotic completeness). $\mathtt{m} \Downarrow p$ *implies* $\mathtt{m} \downdownarrows_s q$*, with* $p \leq q$*.*

Theorem 7.4 states that, for each $\mathtt{m}$, if $\Rightarrow$ reduction *may* converge to snf with probability p, then $\underset{s}{\rightrightarrows}$ reduction *must* converge to snf with probability (at least) p. Theorem 7.3 states that, for each $\mathtt{m}$, the limit q of strict surface reductions ($\underset{s}{\rightrightarrows}$) from $\mathtt{m}$ is *unique*.

Summing-up, the limit q of $\underset{s}{\rightrightarrows}$ reduction is the best convergence result that any sequence from $\mathtt{m}$ can produce. Since $\underset{s}{\rightrightarrows} \subseteq \Rightarrow$, then q is also the greatest element in $\mathtt{Lim}(\mathtt{m}, \Rightarrow)$, *i.e.* $[\![\mathtt{m}]\!] = q$. We hence have proved the following, where item (2.) is the *asymptotic analogue* of the normalization results in Table 1.

▶ **Theorem 7.5** (Asymptotic normalization). *For each* $\mathbf{p} \in \mathcal{P}$*, (1.) the limit* $\mathtt{Lim}(\mathbf{p}, \Rightarrow)$ *has a greatest element* $[\![\mathbf{p}]\!]$*, and (2.)* $\mathbf{p} \downdownarrows_s [\![\mathbf{p}]\!]$*.*

8 Related Work and Discussion

In this paper, we propose a foundational notion of (untyped) quantum λ-calculus with a general reduction, encompassing the full strength of β-reduction while staying compatible with quantum constraints. We then introduce an evaluation strategy, and derive standardization and confluence results. We finally discuss normalization of programs at the limit.

Related Works. For quantum λ-calculi *without measurement*, hence without probabilistic behavior, *confluence* [13, 5] (and even a special form of standardization [13]) have been studied since early work. When dealing with measurement, the analysis is far more challenging. To our knowledge, only confluence has been studied, in pioneering work by Dal Lago, Masini and Zorzi [14]. Remarkably, in order to deal with probabilistic and asymptotic behavior, well before the advances in probabilistic rewriting of which we profit, the authors introduce a very elaborated technique. Notice that in [14] reduction is non-deterministic, but restricted to *surface reduction*. In our paper, their result roughly corresponds to the diamond property of $\overset{*}{\underset{s}{\Rightarrow}}$, together with Theorem 7.3.

No "standard" *standardization* results (like the classical ones we recall in Table 1) exist in the literature for the quantum setting. Notice that the form of standardization in [13] is a reordering of the (surface) *measurement-free* reduction steps, so to perform first beta steps, then quantum steps, in agreement with the idea that a quantum computer consists of a classical device "setting up" a quantum circuit, which is then fed with an input. A similar refinement is also possible for the corresponding fragment of our calculus (namely measurement-free $\underset{s}{\rightarrow}$), but clearly does not scale: think of $(\lambda x.x)\,\mathtt{meas}(U_H\,\mathtt{new}, M, N)$, where the argument of a function is guarded by a measurement.

Our term language is close to [14]. How such a calculus relate with a Call-by-Value λ-calculus such as [42]? A first level of answer is that our setting is an *untyped* λ-calculus; linear abstraction, together with well forming rules, allows for the management of quantum data. In [42], the same role is fulfilled by the (Linear Logic based) *typing system*.

Despite these differences, we do expect that our results can be transferred. As already mentioned, the redex $(\lambda^! x.M)!N$ reflects a Call-by-Push-Value mechanism, which in *untyped form* has been extensively studied in the literature with the name of *Bang calculus* [20, 31, 10], as a uniform framework to encode both Call-by-Name (CbN) and Call-by-Value (CbV). Semantical but also syntactical properties, including *confluence* [20, 31] and *standardization* [24, 3] are analyzed in the Bang setting, and then transferred via *reverse simulation* to both CbV and CbN. More precisely, a CbV (resp. CbN) translation maps forth-and-back weak (resp. head) reduction into surface reduction. Surface normal forms are the CbV image of values (and the CbN image of head normal forms). Since the *Bang calculus* is exactly the fragment of Simpson's calculus [43] without linear abstraction, one may reasonably expect that our calculus can play a similar role in the quantum setting. It seems however that a back-and forth translation of CbV (or CbN) will need to encompass types.

A last line of works worth mentioning is the series of works based on Lineal [5, 4, 18]. However, these works differ from our approach in the sense that the λ-terms themselves are subject to superposition: the distinction between classical and quantum data in an untyped setting is unclear.

References

1 Beniamino Accattoli, Claudia Faggian, and Giulio Guerrieri. Factorize factorization. In *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, January 25-28, 2021, Ljubljana, Slovenia (Virtual Conference)*, volume 183 of *LIPIcs*, pages 6:1–6:25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CSL.2021.6.

2 Zena M. Ariola and Stefan Blom. Skew confluence and the lambda calculus with letrec. *Annals of Pure and Applied Logic*, 117(1):95–168, 2002. doi:10.1016/S0168-0072(01)00104-X.

3 Victor Arrial, Giulio Guerrieri, and Delia Kesner. The benefits of diligence. *International Joint Conference on Automated Reasoning, IJCAR 2024,*, 2024.

4 Pablo Arrighi, Alejandro Díaz-Caro, and Benoît Valiron. The vectorial lambda-calculus. *Information and Computation*, 254:105–139, 2017. doi:10.1016/j.ic.2017.04.001.

5 Pablo Arrighi and Gilles Dowek. Lineal: A linear-algebraic lambda-calculus. *Logical Methods in Computer Science*, 13(1), 2017. doi:10.23638/LMCS-13(1:8)2017.

6 Martin Avanzini, Ugo Dal Lago, and Akihisa Yamada. On probabilistic term rewriting. *Sci. Comput. Program.*, 185, 2020. doi:10.1016/J.SCICO.2019.102338.

7 Hendrik Pieter Barendregt. *The Lambda Calculus – Its Syntax and Semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1984.

8 Benjamin Bichsel, Maximilian Baader, Timon Gehr, and Martin T. Vechev. Silq: a high-level quantum language with safe uncomputation and intuitive semantics. In Alastair F. Donaldson and Emina Torlak, editors, *Proceedings of the 41st ACM SIGPLAN International Conference on Programming Language Design and Implementation, PLDI'20*, pages 286–300. ACM, 2020. doi:10.1145/3385412.3386007.

9 Olivier Bournez and Florent Garnier. Proving positive almost sure termination under strategies. In *Rewriting Techniques and Applications, RTA*, pages 357–371, 2006. doi:10.1007/11805618_27.

10 Antonio Bucciarelli, Delia Kesner, Alejandro Ríos, and Andrés Viso. The bang calculus revisited. *Inf. Comput.*, 293:105047, 2023. doi:10.1016/J.IC.2023.105047.

11 Christophe Chareton, Sébastien Bardin, Franccois Bobot, Valentin Perrelle, and Benoît Valiron. An automated deductive verification framework for circuit-building quantum programs. In Nobuko Yoshida, editor, *Proceedings of the 30th European Symposium on Programming Languages and Systems, ESOP 2021*, volume 12648 of *Lecture Notes in Computer Science*, pages 148–177. Springer, 2021. doi:10.1007/978-3-030-72019-3_6.

12 Ugo Dal Lago, Claudia Faggian, Benoît Valiron, and Akira Yoshimizu. The geometry of parallelism: classical, probabilistic, and quantum effects. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 833–845. ACM, 2017. doi:10.1145/3009837.

13 Ugo Dal Lago, Andrea Masini, and Margherita Zorzi. On a measurement-free quantum lambda calculus with classical control. *Math. Struct. Comput. Sci.*, 19(2):297–335, 2009. doi:10.1017/S096012950800741X.

14 Ugo Dal Lago, Andrea Masini, and Margherita Zorzi. Confluence results for a quantum lambda calculus with measurements. *Electr. Notes Theor. Comput. Sci.*, 270(2):251–261, 2011. doi:10.1016/j.entcs.2011.01.035.

15 Ugo Dal Lago and Margherita Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO Theor. Informatics Appl.*, 46(3):413–450, 2012. doi:10.1051/ita/2012012.

16 Ugo de'Liguoro and Adolfo Piperno. Non deterministic extensions of untyped lambda-calculus. *Inf. Comput.*, 122(2):149–177, 1995. doi:10.1006/INCO.1995.1145.

17 Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. Probabilistic lambda-calculus and quantitative program analysis. *J. Log. Comput.*, 15(2):159–179, 2005. doi:10.1093/LOGCOM/EXI008.

18 Alejandro Díaz-Caro, Mauricio Guillermo, Alexandre Miquel, and Benoît Valiron. Realizability in the unitary sphere. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS'19*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785834.

19 Alejandro Díaz-Caro and Guido Martinez. Confluence in probabilistic rewriting. *Electr. Notes Theor. Comput. Sci.*, 338:115–131, 2018.

20 Thomas Ehrhard and Giulio Guerrieri. The bang calculus: an untyped lambda-calculus generalizing call-by-name and call-by-value. In *Proceedings of the 18th International Symposium on Principles and Practice of Declarative Programming (PPDP 2016)*, pages 174–187. ACM, 2016. doi:10.1145/2967973.2968608.

21 Thomas Ehrhard, Michele Pagani, and Christine Tasson. The computational meaning of probabilistic coherence spaces. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011, June 21-24, 2011, Toronto, Ontario, Canada*, pages 87–96. IEEE Computer Society, 2011. doi:10.1109/LICS.2011.29.

22 Claudia Faggian. Probabilistic rewriting: Normalization, termination, and unique normal forms. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPIcs*, pages 19:1–19:25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.FSCD.2019.19.

23 Claudia Faggian. Probabilistic rewriting and asymptotic behaviour: on termination and unique normal forms. *Log. Methods Comput. Sci.*, 18(2), 2022. doi:10.46298/LMCS-18(2:5)2022.

24 Claudia Faggian and Giulio Guerrieri. Factorization in call-by-name and call-by-value calculi via linear logic. In *Foundations of Software Science and Computation Structures - 24th International Conference, FOSSACS 2021*, volume 12650 of *Lecture Notes in Computer Science*, pages 205–225. Springer, 2021. doi:10.1007/978-3-030-71995-1_11.

25 Claudia Faggian, Gaetan Lopez, and Benoît Valiron. A rewriting theory for quantum λ-calculus. *CoRR*, abs/2411.14856, 2024. arXiv:2411.14856.

26 Claudia Faggian and Simona Ronchi Della Rocca. Lambda calculus and probabilistic computation. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785699.

27 Francesco Gavazzo and Claudia Faggian. A relational theory of monadic rewriting systems, part I. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–14. IEEE, 2021. doi:10.1109/LICS52264.2021.9470633.

28 Simon J. Gay. Quantum programming languages: survey and bibliography. *Mathematical Structures in Computer Science*, 16(4):581–600, 2006. doi:10.1017/S0960129506005378.

29 Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987. doi:10.1016/0304-3975(87)90045-4.

30 Alexander S. Green, Peter LeFanu Lumsdaine, Neil J. Ross, Peter Selinger, and Benoît Valiron. Quipper: A scalable quantum programming language. In Hans-Juergen Boehm and Cormac Flanagan, editors, *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI'13*, pages 333–342. ACM, 2013. doi:10.1145/2491956.2462177.

31 Giulio Guerrieri and Giulio Manzonetto. The bang calculus and the two Girard's translations. In *Proceedings Joint International Workshop on Linearity & Trends in Linear Logic and Applications (Linearity-TLLA 2018)*, volume 292 of *EPTCS*, pages 15–30, 2019. doi:10.4204/EPTCS.292.2.

32 Jan-Christoph Kassing, Florian Frohn, and Jürgen Giesl. From innermost to full almost-sure termination of probabilistic term rewriting. In Naoki Kobayashi and James Worrell, editors, *Foundations of Software Science and Computation Structures - 27th International Conference, FoSSaCS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II*, volume 14575 of *Lecture Notes in Computer Science*, pages 206–228. Springer, 2024. doi:10.1007/978-3-031-57231-9_10.

33 Maja H. Kirkeby and Henning Christiansen. Confluence and convergence in probabilistically terminating reduction systems. In *Logic-Based Program Synthesis and Transformation - 27th International Symposium, LOPSTR 2017*, pages 164–179, 2017. doi:10.1007/978-3-319-94460-9_10.

34 Emanuel H. Knill. Conventions for quantum pseudocode. Technical Report LAUR-96-2724, Los Alamos National Laboratory, Los Alamos, New Mexico, US., 1996.

35 Dongho Lee, Valentin Perrelle, Benoît Valiron, and Zhaowei Xu. Concrete categorical model of a quantum circuit description language with measurement. In Mikolaj Bojanczyk and Chandra Chekuri, editors, *Proceedings of the 41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2021*, volume 213 of *LIPIcs*, pages 51:1–51:20, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.51.

36 M.H.A. Newman. On theories with a combinatorial definition of equivalence. *Annals of Mathematics*, 43(2), 1942.

37 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2002.

38 Michele Pagani, Peter Selinger, and Benoît Valiron. Applying quantitative semantics to higher-order quantum computing. In Suresh Jagannathan and Peter Sewell, editors, *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'14)*, pages 647–658. ACM, 2014. doi:10.1145/2535838.2535879.

39 Luca Paolini, Mauro Piccolo, and Margherita Zorzi. qPCF: higher-order languages and quantum circuits. *Journal of Automated Reasoning*, 63(4):941–966, 2019. doi:10.1007/s10817-019-09518-y.

40 Jennifer Paykin, Robert Rand, and Steve Zdancewic. QWIRE: a core language for quantum circuits. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL'17*, pages 846–858. ACM, 2017. doi:10.1145/3009837.3009894.

41 Gordon D. Plotkin. Call-by-name, call-by-value and the lambda-calculus. *Theor. Comput. Sci.*, 1(2):125–159, 1975. doi:10.1016/0304-3975(75)90017-1.

42 Peter Selinger and Benoît Valiron. A lambda calculus for quantum computation with classical control. *Mathematical Structures in Computer Science*, 16:527–552, 2006. doi:10.1017/S0960129506005238.

43 Alex K. Simpson. Reduction in a linear lambda-calculus with applications to operational semantics. In *Term Rewriting and Applications, 16th International Conference (RTA 2005)*, volume 3467 of *Lecture Notes in Computer Science*, pages 219–234, 2005. doi:10.1007/978-3-540-32033-3_17.

A Convention for Garbage Collection

In the definition of programs, we use the convention that the size of the memory is exactly the number of registers manipulated in the term. The memory will grow when new qubits are allocated, and shrink when qubits are read (see Figure 1): the reduction perform garbage collection on the fly.

If this makes it easy to identify identical programs, it makes the proofs a bit cumbersome. We therefore rely for them on an equivalent representation, where a program can have spurious qubits, as long as they are not *entangled* with the rest of the memory – i.e. when measuring them would not change the state of the registers manipulated by the term. So for instance, in this model $[|0\rangle \otimes |\psi\rangle; r_1]$ is the same as $[\phi; r_0]$.

B Technical properties

In all proofs we freely use the following closure property, which is immediate by definition of context and surface context.

▶ **Fact B.1** (Closure).

$$\frac{[\mathtt{Q}; M] \to_c \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}}{[\mathtt{Q}; \boldsymbol{S}(\!|M|\!)] \to_q \{\!\{ p_i[\mathtt{Q}_i; \boldsymbol{S}(\!|M_i|\!)] \}\!\}}\ 1. \qquad \frac{[\mathtt{Q}; M] \to_\beta \{\!\{ [\mathtt{Q}; M'] \}\!\}}{[\mathtt{Q}; \boldsymbol{C}(\!|M|\!)] \to_\beta \{\!\{ [\mathtt{Q}; \boldsymbol{C}(\!|M'|\!)] \}\!\}}\ 2.$$

Surface closure (point 1.) also holds with $\to_\beta$ in place of $\to_q$.

We will also use the following lemma (analog to substitutivity in [7], p.54) The proof is straightforward.

▶ **Lemma B.2** (Substitutivity). *Assume $[\mathtt{Q}; P] \in \mathcal{P}$ and $[\mathtt{Q}; P] \to \{\!\{ p_i[\mathtt{Q}_i; P_i] \}\!\}$. Then for each term N : $[\mathtt{Q}; P\{N/x\}] \to \{\!\{ p_i[\mathtt{Q}_i; P_i\{N/x\}] \}\!\}$.*

The converse also holds, and it is simply Fact B.1, that can be reformulated as follows.

▶ **Fact B.3.** *Assume $[\mathtt{Q}; N] \in \mathcal{P}$, $[\mathtt{Q}; N] \to_q \{\!\{ p_i[\mathtt{Q}_i; N_i] \}\!\}$ and P a term such that x is* linear *in P. Then $[\mathtt{Q}; P\{N/x\}] \to_q \{\!\{ p_i[\mathtt{Q}_i; P\{N_i/x\}] \}\!\}$*

Surface Reduction. has a prominent role. We spell-out the definition.

Surface Reduction Step $\underset{s}{\to}$

$$\underset{s}{\to} := \underset{s}{\to}_\beta \cup \to_q$$

Surface Beta Step $\underset{s}{\to}_\beta$	**(Surface) q-Step $\to_q$**
$\dfrac{[\mathtt{Q}; M] \mapsto_\beta \{\!\{ [\mathtt{Q}; M'] \}\!\}}{[\mathtt{Q}; \mathbf{S}(\!|M|\!)] \underset{s}{\to}_\beta \{\!\{ [\mathtt{Q}; \mathbf{S}(\!|M'|\!)] \}\!\}}$	$\dfrac{[\mathtt{Q}; M] \mapsto_q \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}}{[\mathtt{Q}; \mathbf{S}(\!|M|\!)] \to_q \{\!\{ p_i[\mathtt{Q}_i; \mathbf{S}(\!|M_i|\!)] \}\!\}}$

Figure 5 Surface Reduction Steps.

C Surface Reduction has the Diamond Property

We obtain the diamond property (Proposition 6.3) from the pointed diamond, result using the following technique (from [26]) , which allows us to *work pointwise*.

▶ **Lemma** (pointwise Criterion (FaggianRonchi19)). *Let $\to_a, \to_b \subseteq \mathcal{P} \times \mathtt{MD}(\mathcal{P})$ and $\Rightarrow_a, \Rightarrow_b$ their lifting. To prove that $\Rightarrow_a, \Rightarrow_b$ diamond-commute,* i.e.

$$\textit{If } \mathbf{p} \Rightarrow_b \mathtt{m}_1 \textit{ and } \mathbf{p} \Rightarrow_a \mathtt{m}_2, \textit{ then } \exists \mathtt{r} \textit{ s.t. } \mathtt{n} \Rightarrow_a \mathtt{r} \textit{ and } \mathtt{s} \Rightarrow_b \mathtt{r}.$$

it suffices to prove the property (#) below (stated in terms of a single program $\mathbf{p}$)

$$\textit{(\#) If } \mathbf{p} \to_b \mathtt{m}_1 \textit{ and } \mathbf{p} \to_a \mathtt{m}_2, \textit{ then } \exists \mathtt{r} \textit{ s.t. } \mathtt{n} \Rightarrow_a \mathtt{r} \textit{ and } \mathtt{s} \Rightarrow_b \mathtt{r}.$$

The same result holds with $\rightrightarrows$ in place of $\Rightarrow$.

The criterion together with Lemma 6.2 (Point 1.) yields

▶ **Prop** (6.3). *Surface reduction $\underset{s}{\Rightarrow}$ has the diamond property. The same holds for $\underset{s}{\rightrightarrows}$.*

D Finitary Standardization

Shape Preservation. We recall a basic but key property of contextual closure. If a step $\to_\gamma$ is obtained by closure under *non-empty context* of a rule $\mapsto_\gamma$, then it preserves the shape of the term. We say that T and T' have *the same shape* if both terms belong to the same production (*i.e.*, both terms are an application, an abstraction, a variable, a register, a term of shape $!P$, $\mathtt{new}$, etc).

▶ **Fact D.1** (Shape preservation). *Assume* $[\mathtt{Q}; M] \to \{\!\{ p_i[\mathtt{Q}_i; M_i] \}\!\}$, $M = \boldsymbol{C}\langle\!| R |\!\rangle, M_i = \boldsymbol{C}\langle\!| R_i |\!\rangle$ *and that the context* $\boldsymbol{C}$ *is* non-empty. *Then (for each* i*),* M *and* M_i *have the same shape.*

An easy-to-verify consequence is the following, stating that *non-surface steps* ($\underset{\neg \mathsf{s}}{\Rightarrow}$)

- do not change the quantum memory
- do not change the shape of the terms

Notice that the qubit state is *unchanged* by $\underset{\neg \mathsf{s}}{\Rightarrow}$ steps, since it can only be a $\underset{\neg \mathsf{s}}{\Rightarrow}_\beta$ step

▶ **Lemma D.2** (Redexes and normal forms preservation). *Assume* $[\mathtt{Q}; M] \underset{\neg \mathsf{s}}{\Rightarrow}_\beta \{\!\{ [\mathtt{Q}; M'] \}\!\}$.

1. M *is a redex iff* M' *is a redex. In this case, either both are* β*-redexes, or both meas-redexes.*
2. M *is* s*-normal if and only if* M' *is* s*-normal.*

Proof of Lemma 6.6.

▶ **Lemma** (Lemma 6.6). $[\mathtt{Q}; M] \underset{\neg \mathsf{s}}{\Rightarrow}_\beta \{\!\{ [\mathtt{Q}; P] \}\!\}$ *and* $[\mathtt{Q}; P] \to_q \mathtt{n}$ *implies* $[\mathtt{Q}; M] \to_q \cdot \Rightarrow_\beta \mathtt{n}$.

Proof. By induction on the context $\mathbf{S}$ such that $P = \mathbf{S}\langle\!| R |\!\rangle$ and $[\mathtt{Q}; \mathbf{S}\langle\!| R |\!\rangle] \to_q \{\!\{ p_i[\mathtt{Q}_i; \mathbf{S}\langle\!| R_i |\!\rangle] \}\!\} = \mathtt{n}$. We exploit in an essential way the fact that M and P have the same shape. ◀

E Asymptotic normalization

Proof Sketch. The proof of Theorem 7.4 relies on an abstract result from the literature [24], which here we reformulate in our setting:

▶ **Lemma E.1** (Asymptotic completeness criterion [24]). *Assume*

i. s-factorisation*: if* $\mathtt{m} \Rightarrow^* \mathtt{n}$ *then* $\mathtt{m} \underset{\mathsf{s}}{\Rightarrow}^* \cdot \underset{\neg \mathsf{s}}{\Rightarrow}^* \mathtt{n}$*;*

ii. $\neg$s-neutrality *:* $\mathtt{m} \underset{\neg \mathsf{s}}{\Rightarrow} \mathtt{m}'$ *implies* $\mathbb{P}(\mathtt{m}) = \mathbb{P}(\mathtt{m}')$.

Then: $\mathtt{m} \Downarrow p$ *implies* $\mathtt{m} \Downarrow_s p$.

Proof of Theorem 7.4. We establishing the two items below, and then compose them.

1. $\mathtt{m} \Downarrow p$ implies $\mathtt{m} \Downarrow_s p$
2. $\mathtt{m} \Downarrow_s p$ implies $\mathtt{m} \downdownarrows_s p'$, with $p \leq p'$

Item (1.) holds because $\underset{\mathsf{s}}{\Rightarrow}$ satisfies both conditions in Lemma E.1: point (i.) holds by Theorem 6.7, point (ii.) by Lemma D.2. Item (2.) is immediate. ◀

Quantum and Classical Markovian Graphical Causal Models and Their Identification

Jonathan Barrett
University of Oxford, UK

Isaac Friend
University of Oxford, UK

Aleks Kissinger
University of Oxford, UK

Abstract

Markov categories allow formalization of probabilistic and causal reasoning in a general setting that applies uniformly to many different kinds of classical probabilistic processes. It has so far been challenging, however, to generalize these techniques to reasoning about quantum processes, as the quantum no-cloning theorem forbids "copy" maps of the sort that have been used to axiomatize conditional independence, and the related notions of complete common causes and Markovianity, in classical Bayesian networks. Here, we introduce a new categorical notion of Markovian causal model, according to which a distinguished subcategory of "common cause" maps plays a similar role to that of "copy" maps in the categorical formulation of Bayesian networks. Moreover, defining causal models as second-order processes yields a clean and flexible formulation of interventions. Our formalism is both rich enough to handle "complete common cause" assumptions and general enough to encompass not only standard classical causal identification scenarios, but also quantum causal scenarios and new kinds of classical causal identification based on imperfect observations. Furthermore, we show that one can reason uniformly across all of these cases using string-diagrammatic techniques.

2012 ACM Subject Classification Computing methodologies → Causal reasoning and diagnostics; Theory of computation → Categorical semantics; Theory of computation → Quantum computation theory

Keywords and phrases causal inference, Bayesian networks, quantum combs, process theories

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.48

Funding *Isaac Friend*: This research was supported in part by Perimeter Institute for Theoretical Physics. Research at Perimeter Institute is supported by the Government of Canada through the Department of Innovation, Science and Economic Development and by the Province of Ontario through the Ministry of Colleges and Universities.
Aleks Kissinger: This publication was made possible through the support of the ID# 62312 grant from the John Templeton Foundation, as part of the "The Quantum Information Structure of Spacetime" Project (QISS). The opinions expressed in this publication are those of the author(s) and do not necessarily reflect the views of the John Templeton Foundation.

Acknowledgements We thank Elie Wolfe and Rob Spekkens for insights regarding the classes of instruments that can be used for causal identification.

1 Introduction

Within the study of probabilistic reasoning, causal inference involves discerning from data the causal relationships responsible for generating them, and hence the effects of hypothetical interventions. Many researchers in quantum information and the foundations of quantum theory have tried to adapt concepts from causal inference to a setting in which, roughly, quantum systems replace random variables as causal relata, and quantum channels replace functions or stochastic maps as causal mechanisms [15, 17, 19, 1, 12]. A natural approach to

studying quantum generalizations of probabilistic reasoning is to start from the literature on categorical probability theory (e.g., [14, 11]), and simply replace a category of probabilistic processes with a category of quantum processes, which one hopes satisfies enough axioms to support analogous calculations to the classical case. But we quickly run into a major obstacle: an important axiom used in categorical probability to define *Markov categories*, and slight variations such as CD and CDU categories, is the existence of "copy" maps for all objects. Such maps are the basis of abstract formulations of conditional independence, and hence the Markov condition for Bayesian networks. When Bayesian networks are understood as causal models as in the work of Pearl [16], Markovianity sometimes involves a variable being "copied" and the copies distributed to other parts of the network that depend on the variable. The Markov condition for causal Bayesian networks is central to classical causal inference.

Such "copy" maps are forbidden for quantum processes, however, as famously shown by the quantum no-cloning [20] and no-broadcasting [2] theorems. While it is possible to define complete common causes, and hence Markovianity, in the quantum setting in several equivalent ways [1, 3], the absence of an explicit representation of copying as a well-defined quantum process in its own right limits the translation of standard causal inference techniques to the quantum setting. The present work solves this problem by introducing an abstract, categorical notion of Markovian causal model that can be instantiated in either a quantum or a classical setting[1]. As in prior work on categorical causal inference [13] (which depended on "copy" structure and did not include quantum models), a causal model involves two symmetric monoidal categories: a *syntactic category*, whose morphisms encode an abstract causal structure as a formal composition of "black boxes," and a *semantic category*, in which the abstract causal structure is functorially interpreted as a particular data-generating process, e.g., by filling in the black boxes with concrete stochastic matrices. Generalizing [13], we will provide a notion of abstract causal structure that can be interpreted in either a classical or a quantum semantic category. In particular, this framework subsumes ordinary classical Bayesian networks.

After defining the basic framework, we will describe how our formalism handles interventions, and pose the *causal identification* problem to which we will apply our mathematical technology. Causal identification is a type of causal inference problem for which the effects of counterfactual interventions are to be inferred from a combination of qualitative hypotheses (represented by a graph) and observational data. Our formulation of causal models lets us treat the statistics from a very restricted class of interventions as "observational data" available for inference. The precise class of interventions we choose is treated as a parameter in our framework, so different classes of interventions yield different kinds of causal identification problems. This flexibility is needed in the quantum case, where there is no standard notion of "passive observation," but is also useful in the classical case, where we can now study inference tasks whose input data have been obtained via imperfect procedures.

In classical causal inference, the assumption that an unknown causal model is a Markovian model based on a known graph, amounting to the assumption that there are no latent variables influencing multiple observed variables (i.e., no latent confounders), greatly expands the class of causal queries that can be answered with observational data. In particular, with this assumption, one can identify from the graph and the observational data the response of the model to arbitrary interventions. We demonstrate that a certain Markovianity

[1] The resulting notion of quantum Markovian causal model, specifically $(\mathbf{CPM}, \mathbf{Unitary}_\bullet)$-valued Markovian model, is closely related to proposals in [8] and [1, 3].

assumption formulated in our new framework is similarly powerful in the quantum setting, allowing the identification of the entire data-generating process from only very limited probing operations. We simultaneously demonstrate that for classical causal identification, noisy and disturbing observations can sometimes serve in place of ordinary perfect passive observations. The uniform handling of the classical and quantum cases is made possible by the string-diagrammatic calculus for (compact) symmetric monoidal categories.

2 Process theories

Throughout the paper, we will use *process theoretic* terminology, following, e.g., [5], to discuss morphisms in a symmetric monoidal category. Namely, we will refer to symmetric monoidal categories $(\mathcal{C}, \otimes, I)$ as *process theories* and the morphisms therein as *processes*. Because of their physical interpretation, we also introduce special terminology for morphisms into and out of the monoidal unit I. In a process theory, morphisms of the form $\rho : I \to A$ are called *states*, and morphisms of the form $\pi : A \to I$ are called *effects*. Morphisms of the form $\lambda : I \to I$ are called *numbers* or *scalars*.

We will focus on process theories equipped with distinguished families of *discarding* maps $d_A : A \to I$, one map for each object A, satisfying $d_{A\otimes B} = d_A \otimes d_B$ and $d_I = 1_I$. The main utility of discarding maps is allowing us to say when a process is *causal*, which in the classical and quantum settings imposes a normalization constraint.

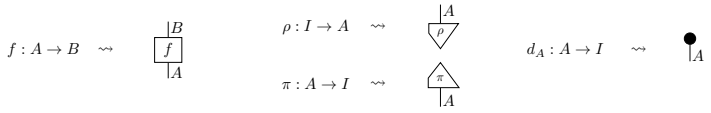
▶ **Definition 1.** *A process $f : A \to B$ is called causal if $d_B \circ f = d_A$.*

▶ **Example 2.** The process theory $\mathbf{Mat}[\mathbb{R}_+]$ has as objects natural numbers and as processes $M : m \to n$ the $n \times m$ matrices whose entries are non-negative real numbers $\{M^i_j \mid 1 \leq i \leq n, 1 \leq j \leq m\}$. The monoidal product is given by tensor product of matrices (a.k.a. Kronecker product), whose unit is the 1×1 matrix $[1] : 1 \to 1$. Discarding maps $d_n : n \to 1$ are the $1 \times n$ matrices (i.e., row vectors) consisting of all 1s. Composing with d_n corresponds to summing over an output index (i.e., marginalization). Consequently, causal states are column vectors of positive numbers whose entries sum to 1 (i.e., probability distributions), and causal processes are matrices whose columns each sum to 1 (i.e., stochastic maps, equivalent to conditional probability distributions with $P(i|j) := M^i_j$).

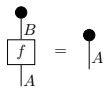
▶ **Example 3.** The process theory **CPM** has as objects finite-dimensional Hilbert spaces $\mathcal{H}, \mathcal{K}, \ldots$ and as morphisms completely positive maps $\Phi : L(\mathcal{H}) \to L(\mathcal{K})$, where $L(\mathcal{H})$ is the algebra of operators $\mathcal{H} \to \mathcal{H}$. The monoidal product is again given by tensor product, whose unit is the identity map on $L(\mathbb{C}) \cong \mathbb{C}$. Discarding maps are trace maps. A state $\rho : \mathbb{C} \to L(\mathcal{H})$ is fixed by a single positive operator $\rho(1) \in L(\mathcal{H})$, and causal states correspond to trace-1 positive operators. More generally, causal processes are the trace-preserving completely positive maps.

Since both matrices of positive numbers and completely positive maps are closed under sums, both $\mathbf{Mat}[\mathbb{R}_+]$ and **CPM** are additively enriched. We will first use this fact in Definition 4 to define *instruments*.

The presentation will use *string diagram* notation, with processes depicted as boxes and objects as wires in diagrams read from bottom to top. A process theory's monoidal unit object I and the identity process $I \to I$ are both depicted by empty space, and other identity processes are depicted as wires. Discarding, which will later serve as a counit for an internal comonoid structure, is depicted with a black dot.



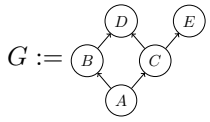
Diagrammatically, the causality condition for a process $f : A \to B$ from Definition 1 is



(1)

3 Causal Bayesian networks

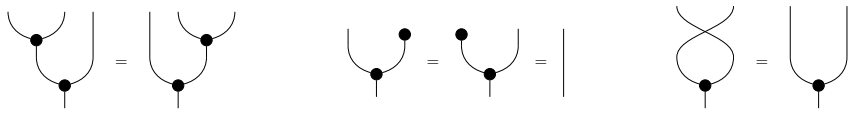
The usual notion of a joint probability distribution being Markov compatible with a directed acyclic graph (DAG) is that it factorizes in such a way that each variable (labeling a node of the graph) is independent when conditioned on its parents. For example, a joint distribution $P(ABCDE)$ is Markov compatible with DAG



(2)

precisely when $P(ABCDE) = P(A)P(B|A)P(C|A)P(D|BC)P(E|C)$.

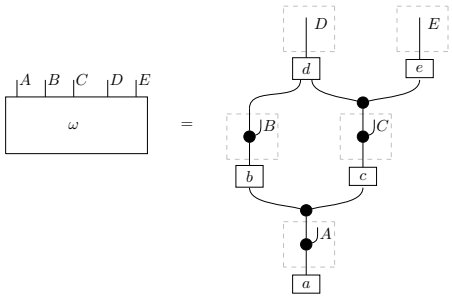
We now recall the string-diagrammatic formulation of Markov compatibility of a joint probability distribution with a DAG G, given, e.g., in [13]. First, we introduce for each object X in $\mathbf{Mat}[\mathbb{R}_+]$ a "copy" map $X \to X \otimes X$, whose composition with a point distribution $\psi : I \to X$ (i.e., a column vector with a single 1 entry and 0s elsewhere) is $\psi \otimes \psi$. With "copy" as comultiplication–depicted by a black dot with one input and two output wires–and discarding as counit, each object in $\mathbf{Mat}[\mathbb{R}_+]$ is given a cocommutative comonoid structure:



(3)

A symmetric monoidal category equipped with a compatible family of "copy" and discard maps for all objects is called a *CD category*. If in addition we impose the causality condition of Definition 1 on all maps, the category is called a *Markov category*. The copy and discard maps above endow $\mathbf{Mat}[\mathbb{R}_+]$ with the structure of a CD category; the subcategory $\mathbf{Stoch} \subseteq \mathbf{Mat}[\mathbb{R}_+]$ of stochastic (i.e., causal) maps is a Markov category.

We can form the string diagram associated with a DAG G by introducing a box $a : X_1 \otimes \ldots \otimes X_n \to A$ for every node A in G with parents $\{X_1, \ldots, X_n\}$. We compose these boxes by connecting each output A to the output of the overall diagram, as well as to the inputs of each of the children of A in G, introducing copy maps where necessary. A state being Markov with respect to G then means simply that it factorizes according to that diagram, for some choice of stochastic matrices $a, b, c, \ldots$ For example, a state ω is Markov with respect to the graph G from (2) when it can be decomposed as follows:



(4)

This diagrammatic condition corresponds precisely to the usual factorization of $P(ABCDE)$ given before, where $P(ABCDE)$ is the joint probability distribution given by the state ω.

The right-hand side of (4) may be said to represent ω as a *Bayesian network*. While Bayesian networks can in general just be seen as an efficient way to represent joint probability distributions, we can additionally provide them with a causal interpretation, using them to model how a certain scenario would respond to possible interventions. To model the effects of interventions using a Bayesian network, we interpret each of the boxes $a, b, c, \ldots$ as some actual (e.g., physical) mechanism that determines (stochastically) the value of its output, given any value of its input. One introduces a concept of local intervention whereby, for example, a change can be made at the input to box c while the rest of the network is left unchanged. In [13], local intervention is represented by endofunctors that "cut" diagrams like the one in Eq. 4. Such an intervention results in a different overall state from ω (the new state is sometimes called a "do-conditional" in the causal inference literature [16]). A Bayesian network representation of a state that is Markov compatible with DAG G, interpreted causally in this way, is called a *Markovian G-based causal model*.

In a causal inference problem, one is given a state like ω together with certain qualitative assumptions about how the state is generated, and the task is to determine further properties of the data-generating process and compute how ω would change under hypothetical interventions. One sort of qualitative assumption is that ω is generated by a Markovian causal model based on a certain DAG. Such an assumption turns out quite powerful for inference: with it, one can evaluate the results of essentially any hypothetical intervention. Discussions of "quantum causal modeling" naturally suggest the question of whether a "quantum Markovianity assumption" might provide similar inferential power in quantum causal scenarios. We therefore seek to formulate quantum Markovian causal models based on DAGs, and an associated inference problem.

Two obstacles arise. First, it is unclear what quantity constitutes the quantum analog of the state (ω in Eq. 4) that is an input in classical inference. That state carries "observational data," i.e., the probability distribution generated by the causal model when variables are merely observed rather than intervened on. In operational quantum theory, there is no standard notion of passive observation as distinct from more "active" intervention. Our solution makes no such distinction in principle, and instead simply allows any set of interventions to be declared the "accessible" ones whose outcome distributions will be available for inference. The second obstacle is the absence of copy maps in quantum theory. The assumption that observational data are generated by a process like the one on the right-hand side of Eq. 4 is useful for inference because copy maps guarantee, for example, that any randomness shared between the inputs to b and c is accounted for by variable A. (The Bayesian network representation in Eq. 4 also uses copy maps to produce an observed output for each variable while allowing the variable's value to be fed forward, undisturbed, to the rest of the network.) Our solution here allows any subtheory of maps in a process theory to take the role typically played by copy maps in distributing "information" from

the output of one box to the inputs of other boxes. With such a general framework, there remains the question of which choices of parameters in the quantum setting give a specific notion of "Markovian causal model" that is especially useful for inference. The answer, in Section 7, depends on a theorem showing how unitary quantum maps (more generally, what are called "autonomous" quantum channels) mimic the "function-maps" in $\mathbf{Mat}[\mathbb{R}_+]$.

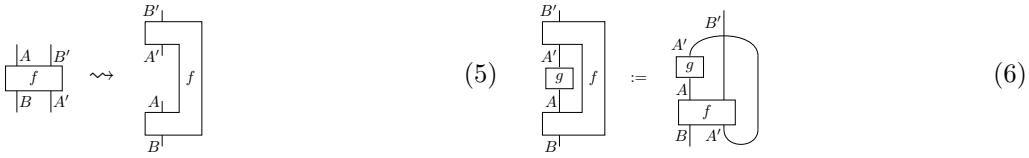
4 Generalized causal models

4.1 Combs and instruments

The interventional causal models studied in this paper will involve second-order processes, or *combs* [4], taking first-order processes as input and producing other first-order processes (usually numbers, i.e. processes $I \to I$). We will represent second-order processes as first-order ones by invoking the (self-dual) *compact structures* in the process theories $\mathbf{Mat}[\mathbb{R}_+]$ and $\mathbf{CPM}$: every object A is equipped with a pair of maps $\cup_A : I \to A \otimes A$ and $\cap_A : A \otimes A \to I$, called "cups" and "caps" respectively, satisfying the so-called *yanking equations*, which are depicted in string diagram notation as follows:

In $\mathbf{Mat}[\mathbb{R}_+]$, cups and caps are given by Kronecker delta matrices, with the two indices treated as either inputs or outputs: $\cup^{ij} = \cap_{ij} = \delta_{ij}$. In $\mathbf{CPM}$, $\cup_{\mathcal{H}}$ is given by the unnormalized maximally entangled state $\sum_{ij} |ii\rangle\langle jj|$ and $\cap_{\mathcal{H}}$ is its associated effect, seen as a completely positive map from $L(\mathcal{H}) \otimes L(\mathcal{H})$ to $\mathbb{C}$.

Using this structure, we can, for example, represent a process that takes processes of type $A \to A'$ and produces processes of type $B \to B'$ as a normal, first-order process $f : B \otimes A' \to A \otimes B'$. We then indicate its higher-order interpretation by drawing f as a box with a "hole" in it, often called a *comb*, and use cups and caps to define "plugging" another box into that hole:



As in [9, 10], a classical or quantum causal model will involve a comb in $\mathbf{Mat}[\mathbb{R}_+]$ or $\mathbf{CPM}$, respectively, encoding the stable mechanisms governing a repeated causal scenario. The "holes" in the comb will represent loci of intervention, where one can interact with the data-generating process in various ways, e.g., by implementing a causal map (a classical or quantum channel), or observing the value of a random variable and then feeding forward a certain state. An intervention procedure at a "hole" in a classical or quantum comb will be represented mathematically by an *instrument*.

▶ **Definition 4.** *An instrument of type $A \to A'$ valued in $\mathbf{Mat}[\mathbb{R}_+]$ or $\mathbf{CPM}$ is a finite set of maps $\{\phi_i : A \to A'\}_i$ whose sum $\sum_i \phi_i$ is a causal map. Each map ϕ_i is called a branch of the instrument.*

Branches correspond to possible outcomes of the intervention procedure. The probabilities of these outcomes are determined by the branches and by the process in which one is intervening.

▶ **Example 5.** The preparation of a causal state is represented by an instrument branch $\{\rho : I \to A\}$. A *demolition measurement* on a quantum system is represented by an instrument whose branches are effects $\{\phi_i : A \to I\}_i$. The only causal effect is the discarding map d_A, so the instrument condition says $\sum_i \phi_i = d_A$. For a causal state ρ, the probability of getting outcome i is $P(i|\rho) := \phi_i \circ \rho$. From the instrument condition, it follows that $\sum_i P(i|\rho) = \sum_i \phi_i \circ \rho = d_A \circ \rho = 1$.

If f in Eq. 5 is a causal process in $\mathbf{Mat}[\mathbb{R}_+]$ or $\mathbf{CPM}$, and one selects instruments $I \to B$, $A \to A'$, and $B' \to I$, then f will map each possible triple of branches to a probability, understood as the probability of realizing this triple when probing f with the selected instruments. Causal inference in general consists in using such probabilities–imagined to have been learned experimentally over many trials–to compute properties of f, whose value is initially unknown, and thereby predict how f would respond to other combinations of instruments.

This paper focuses on a kind of causal inference problem called *causal identification*, for which certain properties of the comb, namely its "shape," are assumed in advance, and those assumptions used together with the probabilities just described to compute the further properties of interest. The assumption we will formalize and use for inference is *Markovianity*. We will now give a process-theoretic definition of Markovian causal model that can be instantiated in either $\mathbf{Mat}[\mathbb{R}_+]$–where we recover a standard definition of Markovian causal model–or $\mathbf{CPM}$.

4.2 Abstract and concrete causal structures

We extend the recipe from [13] where a directed acyclic graph is used to generate a process theory whose morphisms are abstract causal structures encoding qualitative assumptions that will be used for inference. For a finite directed acyclic graph $G = (V_G, E_G)$ with vertex set V_G and edge set E_G, let $\mathbf{G}_0$ be a free symmetric monoidal category whose objects are generated by the set $V_G \uplus E_G$ and whose morphisms are generated by discarding maps for all objects and two additional kinds of maps:

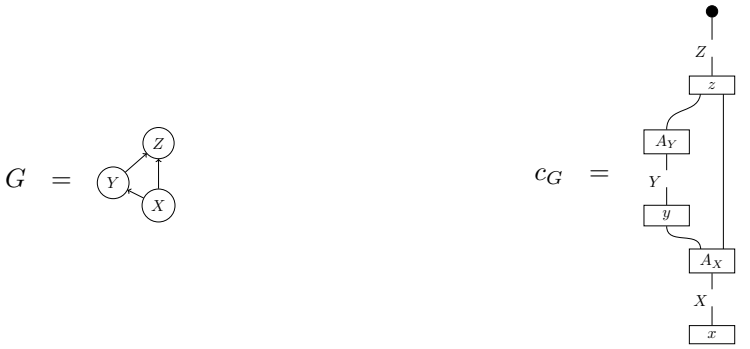
$$x : e_1 \otimes \ldots \otimes e_j \to X \qquad\qquad A_X : X \to e'_1 \otimes \ldots \otimes e'_k \tag{7}$$

for each $X \in V_G$, where $\{e_1, \ldots, e_j\}$ are the in-edges of X and $\{e'_1, \ldots, e'_k\}$ are the out-edges of X.

From the free category $\mathbf{G}_0$, we form the "syntactic" process theory $\mathbf{G}$ by additionally imposing the causality equation (1) for every generating map. In particular, for Z a vertex with no out-edges, $A_Z = d_Z$.

We then associate to the graph G a process $c_G : X_1 \otimes \ldots \otimes X_n \to X_1 \otimes \ldots \otimes X_n$ in $\mathbf{G}$, called the *abstract causal structure* associated with G, by taking each of the generators from (7) and plugging each input wire labeled by an edge to the unique associated output wire. The inputs and outputs of c_G are all labeled by vertices, each vertex labeling exactly one input and one output wire. Each input/output pair is depicted as a hole in a wire and labeled by the corresponding vertex in V_G.

▶ **Example 6.** The directed graph G indicates the abstract causal structure c_G:



The graph is one of the inputs for the causal inference problem we will be studying. Just as in [13] being given the graph in (2) would let an agent assume that the observed probability distribution is generated by a process conforming to (4), being given the three-node graph in this example will let an agent assume that the unknown causal scenario is represented by a comb conforming to c_G. A class of *interventional causal models* respecting this abstract causal structure is defined relative to a pair of process theories $(\mathcal{C}, \mathcal{C}_{cc})$, where $\mathcal{C}_{cc}$ is a subtheory of $\mathcal{C}$ called the "common-cause" subtheory. (The common-cause subtheory is a parameter in the framework; specifying the common-cause subtheory is part of defining a causal identification problem. We will study the consequences of various choices of common-cause subtheory.)

▶ **Definition 7.** *A G-based, $(\mathcal{C}, \mathcal{C}_{cc})$-valued Markovian interventional causal model consists of a discarding-preserving functor of process theories (i.e., a discarding-preserving symmetric monoidal functor) $F : \boldsymbol{G} \to \mathcal{C}$ such that $F(A_X)$ is in $\mathcal{C}_{cc}$ for every $X \in V_G$.*

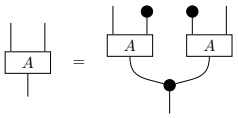
The process $F(c_G)$ is a concrete causal structure, i.e., it is a morphism in $\mathcal{C}$, such as a stochastic matrix (in the classical case) or a quantum channel, that assigns probabilities to outcomes of intervention procedures implemented at *intervention loci* (*loci* for short) represented by the input/output pairs that form the "holes" in the abstract and concrete causal structures. We will consider scenarios in which F is initially unknown, and we will try to compute elements in F's image from those probabilities.

In these scenarios, the process theories $\mathcal{C}$ and $\mathcal{C}_{cc}$, like the graph G, are given in advance. A process $F(A_X)$ in the common-cause subtheory $\mathcal{C}_{cc}$ distributes information from locus X toward the loci labeled by X's children in G. The common-cause subtheory determines what it means to assume (ultimately for the purpose of inference) that a locus is the *complete common cause* of the loci labeled by its children in G. There are no conditions on $\mathcal{C}_{cc}$ *a priori*, except that it should contain the family of discarding processes. In particular, we could have $\mathcal{C}_{cc} = \mathcal{C}$, in which case the notion of "complete common cause" is trivialized. However, for some classes of models below, $\mathcal{C}_{cc}$ will be a subtheory of processes that we think of as disallowing confounding between their outputs due to latent variables/systems. In this case, any observed correlations are thought of as arising entirely from the causal dependency of multiple output variables/systems on some input. We will formulate this concept for relevant subtheories of classical and quantum processes, where it will bestow significant inferential power. The basic idea is simple: if one wishes to infer the value of a process known to decompose according to a certain string diagram, then knowing that certain boxes are valued in a smaller subtheory will tend to make the task easier. We will show that certain classical and quantum subtheories are particularly useful in this regard, for mathematical reasons that are precisely analogous between the two settings. Nevertheless, it is important to understand that the term "Markovian" is used in Def. 7 in a new and abstract sense.

The causal models studied in this paper are valued in the process theories $\mathbf{Mat}[\mathbb{R}_+]$ and **CPM**. Denote by **Func** the subtheory of $\mathbf{Mat}[\mathbb{R}_+]$ consisting of function-maps, i.e., stochastic maps whose columns each contain precisely one 1. This theory is equivalent to the theory of finite sets and functions: associate with each matrix M in **Func** the unique function f with $M^i_j := 1$ if and only if $f(j) = i$. Relevant subtheories of **CPM** include the theories **Unitary** and **Isom** of unitary and isometric quantum channels, i.e. completely positive maps of the form $\mathcal{U}(\rho) := U \circ \rho \circ U^\dagger$ for U a unitary or an isometry. For a subtheory $\mathcal{D}$ of a process theory $\mathcal{C}$ with discarding, the theory $\mathcal{D}_\bullet$ is formed by adjoining discarding maps for all objects. Thus we have, e.g., $\mathbf{Func}_\bullet = \mathbf{Func}$, and $\mathbf{Unitary}_\bullet$ is the theory of what are called *autonomous quantum channels* [18].

4.3 Recovering classical causal Bayesian networks

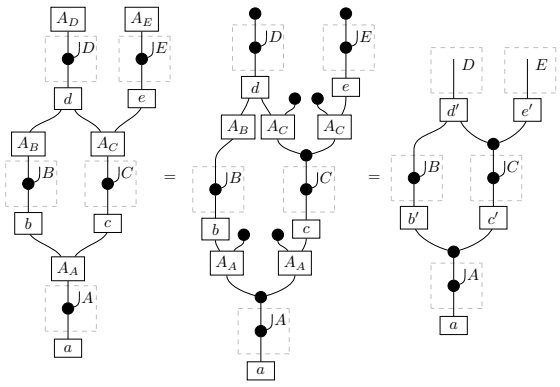
One reason for our use of the term "Markovian" in a manner specific to the new framework we are introducing is that this framework subsumes ordinary Markovian causal Bayesian networks. The key to establishing the relationship is the following property of function-maps in $\mathbf{Mat}[\mathbb{R}_+]$[2]:



(8)

▶ **Proposition 8.** *For any* $(\boldsymbol{Mat}[\mathbb{R}_+], \boldsymbol{Func}_\bullet)$*-valued model of directed acyclic graph* G*, the state in* $\boldsymbol{Mat}[\mathbb{R}_+]$ *derived by plugging a copy map into each locus (as in the left-hand side of Eq. 9) is Markov compatible with* G *in the standard sense used in probabilistic graphical modeling. Conversely, any Bayesian network based on DAG* G *is derivable from some* G*-based,* $(\boldsymbol{Mat}[\mathbb{R}_+], \boldsymbol{Func}_\bullet)$*-valued causal model by this prescription.*

Proof. A state like the one in Eq. 4 is a G-based, $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued model composed with copy maps at all loci. The common-cause functions following the loci happen also to be copy maps, which are indeed morphisms in $\mathbf{Func}_\bullet$. On the other hand, starting from a $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued model with generic common-cause functions also yields a state of this form, thanks to Eq. 8 for functions:



(9)

◀

[2] In synthetic probability based on Markov categories [11], Eq. 8 defines conditional independence of a map's outputs (conditioned on its input).

We have now shown using Eq. 8 how some of the copy maps in classical Bayesian networks emerge in our framework when $\mathbf{Func}_\bullet$ is the common-cause subtheory. In the next section, after posing the general identification problem, we will show how standard "observational data" can be extracted from our classical interventional models via what we call *perfect passive observation* instruments, which do not involve copy maps. We will then have recovered the standard notion of classical causal identification as just one case on the same footing as quantum and new classical problems.

5 Interventions and the identification problem

▶ **Definition 9.** *For a given abstract causal structure, a semantic process theory $\mathcal{C}$, and an object A in $\mathcal{C}$ for each locus in the abstract causal structure, a local intervention regime assigns to each A an instrument in $\mathcal{C}$ of type $A \to A$.*

For a fixed local intervention regime and a fixed classical or quantum model of the abstract causal structure, "implementing" the intervention regime for one iteration of the causal scenario results in the joint realization of a combination of maps at all the loci: at each locus, one branch of the instrument assigned to that locus is realized. The joint probability of this combination of local outcomes is the number resulting from plugging the maps into their loci. The problem of causal identification is to use such probabilities from a limited set of local intervention regimes, together with the shape of the abstract causal structure (equivalently, the graph), to infer probabilities of outcome combinations under other local intervention regimes.

The set of local intervention regimes whose outcome statistics are to be used for inference is constructed as follows: for each locus A, an *accessible set* $\mathcal{I}_A$ of instruments $A \to A$ is given. These accessible sets of instruments define a set of *accessible* local intervention regimes.

▶ **Definition 10.** *Given an accessible set $\mathcal{I}_A$ of instruments for each locus A in an abstract causal structure, an accessible local intervention regime assigns to each A an instrument from $\mathcal{I}_A$.*

The probabilities available for inference are the probabilities that can be "learned" from accessible local intervention regimes. That is, for each accessible local intervention regime, the joint probability of each combination of branches will be considered known.

Note that we will always assume every accessible set $\mathcal{I}_A$ contains the identity instrument of the appropriate type. An identity instrument has one branch, an identity process, depicted by a wire. Assuming the universal availability of identity instruments is a way of assuming that given any allowed local intervention regime, one can also choose to "do nothing" at one of the loci, keeping the same instruments at all other loci[3].

In causal identification, what one is trying to identify is the image of some map in the syntactic process theory $\mathbf{G}$ under F. Knowing such an image might allow one to determine how the model would respond to certain local intervention regimes. For instance, if one were confronted with an unknown model of the abstract causal structure in Example 6, inferring the value of $F(y \circ A_X)$ would allow one to predict the outcome probabilities for a

[3] The reason identity interventions are usually not discussed in classical causal inference literature is that they can be simulated from perfect passive observational data, by "marginalization." This is no longer the case, however, when we move beyond classical perfect passive observation. For example, performing a quantum measurement and then marginalizing over the outcome will *not* in general lead to the same statistics on the remaining loci as not doing the measurement at all.

local intervention regime consisting of an identity instrument at Z and arbitrary instruments at X and Y. However, one is initially given only limited access to the functor F, namely the probabilities associated with accessible local intervention regimes applied to the whole data-generating process $F(c_G)$. For simplicity, we will focus on the problem of inferring $F(c_G)$ itself, from which one can compute the outcome probabilities for arbitrary local intervention regimes. In general, however, we might only be interested in predicting the results of interventions at certain loci, in which case we might be able to focus on a simpler problem.

In full, the causal identification problem we will study is defined by the following in each instance: a directed acyclic graph G, specifying an abstract causal structure $c_G \in \mathbf{G}$; a semantic process theory $\mathcal{C}$ (either $\mathbf{Mat}[\mathbb{R}_+]$ or $\mathbf{CPM}$) and a subtheory $\mathcal{C}_{cc}$; a G-based, $(\mathcal{C}, \mathcal{C}_{cc})$-valued Markovian interventional causal model F; and an accessible set $\mathcal{I}_A$ of instruments for each locus A.

The inputs for the identification task are the (labeled) graph G, the pair of concrete process theories $(\mathcal{C}, \mathcal{C}_{cc})$, the accessible set of instruments for each locus A, and the data generated by $F(c_G)$ under each accessible local intervention regime (i.e., the joint probabilities of realizing combinations of branches). The task is to compute $F(c_G)$. If this task is possible, we will say G-based $(\mathcal{C}, \mathcal{C}_{cc})$-valued models are *identifiable* from the accessible sets of instruments.

For the kinds of classical and quantum causal scenarios we are studying, there always exist finite sets of local instruments that, if declared accessible, suffice for identification regardless of the common-cause subtheory[4]. In contrast, we will consider how accessible sets that do not suffice for identification of models with one common-cause subtheory become sufficient when the common-cause subtheory is further restricted.

A typical example in the case of $\mathcal{C} = \mathbf{Mat}[\mathbb{R}_+]$ is for the accessible set of instruments at each locus to consist of the "perfect passive observations." Note that when reasoning in both the classical and quantum process theories simultaneously, we draw generic states and effects as asymmetric triangles, reserving symmetric triangles for $\mathbf{Mat}[\mathbb{R}_+]$ alone.

▶ **Definition 11.** *For object A in $\boldsymbol{Mat}[\mathbb{R}_+]$, the perfect passive observation instrument of type $A \to A$ has branches*

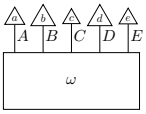
where the state labeled i is given by the column vector with 1 *in row i and all other entries* 0*, and the effect labeled i is the matrix transpose of that column vector.*

Knowing which branch of a perfect passive observation instrument has been implemented means being certain of the value a random variable has taken, and certain that the variable retains that value as it is input to subsequent causal mechanisms.

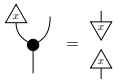
In the previous section, we saw that any $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued model can be translated diagrammatically into a Bayesian network by plugging "copy" maps into all loci. This procedure is equivalent to considering such a model with perfect passive observations at each locus. We can see this by noting that the probability of any particular joint outcome associated with a joint state ω such as the one in (4) can be obtained by plugging in the effect associated to that outcome, e.g.:

[4] See p. 107 of [9] for a way to construct such sets, and explanation of how the construction represents the idea of controlled experiments, from which we expect to be able to deduce any data-generating process.

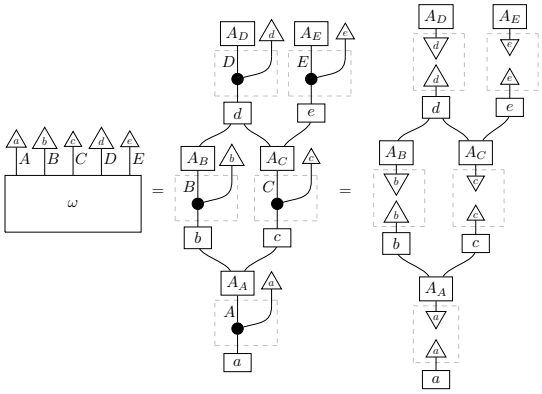
$P(A=a, B=b, C=c, D=d, E=e) \;=$



Then, we can apply the following equation satisfied by the copy and any effect associated with a unit vector:



to obtain a perfect passive observation at every locus; e.g.,



Hence, knowing the state ω is the same as knowing the probabilities associated with perfect passive observations.

We now study what kinds of classical models can be identified from perfect passive observations.

▶ **Proposition 12.** *Perfect passive observation instruments do not suffice for identifying* $(\boldsymbol{Mat}[\mathbb{R}_+], \boldsymbol{Mat}[\mathbb{R}_+])$*-valued Markovian models.*

The proposition means that for some graphs G, there are multiple G-based, $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Mat}[\mathbb{R}_+])$-valued Markovian models that behave identically under all local intervention regimes involving only perfect passive observation instruments, but differently under other local intervention regimes. It should not surprise readers familiar with causal inference; if the common-cause maps can be arbitrary stochastic matrices, they can essentially introduce confounding. The modifier "Markovian" would not ordinarily be applied to generic instances of what we are calling $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Mat}[\mathbb{R}_+])$-valued Markovian models. It would, however, describe what we call $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued Markovian models. For these models, where common-cause maps are restricted to functions, there are no hidden confounders and identification from perfect passive observation is always possible.

▶ **Proposition 13.** $(\boldsymbol{Mat}[\mathbb{R}_+], \boldsymbol{Func}_\bullet)$*-valued Markovian models are identifiable from perfect passive observation instruments.*

In the proof in Appendix A, the last rewriting step uses the fact that the classical "copy" map literally copies the state that leaves a locus after a perfect passive observation. In quantum inference, and in classical inference with generalized observation, this calculation will be unavailable, and a new technique will be introduced to take its place.

5.1 Quantum and generalized classical observation

In some classical causal scenarios, observational data are noisy[5]. After one learns the result of a test, one's credences about the possible values of the variable are given by a probability distribution. Furthermore, one's credences about the possible values being fed forward may be different–one may understand that the procedure whereby one learns about the variable's value tends to change the value. We now study causal identification in such situations and in quantum scenarios by process-theoretically generalizing perfect passive observation instruments to kinds of classical and quantum instruments that may be "noisy" rather than "perfect," and "disturbing" rather than "passive," but are consistent with the idea of observational instruments as those for which the state leaving a locus is determined by the effect that has been realized there, and not by the experimenter's further choice. Quantum causal identification with, e.g., projective measurement instruments turns out similar to classical causal identification with noisy and disturbing instruments.

The generalized observations that will constitute the accessible sets of instruments are such that when one learns an outcome, one models the observation as having implemented an effect followed by a state.

▶ **Definition 14.** *A process of type $A \to A'$ is called $\circ$-separable if it consists of an effect $A \to I$ followed by a state $I \to A'$.*

▶ **Definition 15.** *An instrument of type $A \to A'$ is $\circ$-separable if each of its branches is a $\circ$-separable process.*

▶ **Example 16.** A surgical intervention, composed of a discarding map followed by a causal state preparation, corresponds to a $\circ$-separable instrument with one branch.

▶ **Example 17.** A classical perfect passive observation instrument is a $\circ$-separable instrument.

▶ **Example 18.** Any orthonormal basis for a finite-dimensional Hilbert space induces a $\circ$-separable **CPM**-valued instrument called an ONB measurement (a.k.a. a non-degenerate von Neumann measurement).

The entities that will serve as well as perfect passive observation instruments for causal identification are "complete sets of $\circ$-separable instruments," whose definition invokes the concept of "informational completeness."

▶ **Definition 19.** *A set of effects $\{\pi_j : A \to I\}$ is called informationally complete for A if any state $\rho : I \to A$ is uniquely determined by the set of numbers $\pi_j \circ \rho$. Similarly, a set of states $\{\rho_j : I \to A\}$ is called informationally complete for A if any effect $\pi : A \to I$ is uniquely determined by the set of numbers $\pi \circ \rho_j$.*

In $\mathbf{Mat}[\mathbb{R}_+]$ and **CPM**, a set is informationally complete if and only if it spans the relevant vector space, where the vector space associated with an object A in **CPM** is the space $L(A)$ of linear operators on Hilbert space A.

▶ **Definition 20.** *A set of $\circ$-separable instruments of type $A \to A$ will be called complete if (i) the set of all states appearing in the branches of the instruments is informationally complete for A, and (ii) the set of all effects appearing in the branches of the instruments is informationally complete for A.*

[5] Hidden Markov models, the standard means of modeling this phenomenon, graphically represent variables quite differently from causal Bayesian networks, and moreover are not meant for studying interventions.

▶ **Example 21.** In $\mathbf{Mat}[\mathbb{R}_+]$, a perfect passive observation instrument by itself constitutes a complete set of $\circ$-separable instruments.

▶ **Example 22.** For any object in **CPM**, there is a finite set of ONB measurements forming a complete set of $\circ$-separable instruments. One example is the set of ONB measurements corresponding to the bases of eigenvectors for the d^2 generalized Pauli matrices on a Hilbert space of dimension d.

▶ **Example 23.** In $\mathbf{Mat}[\mathbb{R}_+]$, let

$$\phi = \begin{bmatrix} .8 & .9 \end{bmatrix} \qquad \phi' = \begin{bmatrix} .2 & .1 \end{bmatrix} \qquad \psi = \begin{bmatrix} .5 \\ .5 \end{bmatrix} \qquad \psi' = \begin{bmatrix} .9 \\ .1 \end{bmatrix}$$

The instrument with branches $\psi \circ \phi$ and $\psi' \circ \phi'$ constitutes a single-instrument marginally informationally complete set of $\circ$-separable instruments of type $2 \to 2$. When a locus representing a binary random variable (with values denoted 1 and 2) is probed with this instrument, if the variable's true value is 1, $\psi \circ \phi$ is realized with probability .8 and $\psi' \circ \phi'$ with probability .2. If the true value of the variable is 2, $\psi \circ \phi$ is realized with probability .9 and $\psi' \circ \phi'$ with probability .2. If branch $\psi \circ \phi$ is realized, the value of the variable fed forward after the probing is totally randomized. If branch $\psi' \circ \phi'$ is realized, the value fed forward is 1 with probability .9 and 2 with probability .1. Instruments that are "biased" toward the realization of certain branches can be thought of as modeling certain kinds of selection effects, which are an important topic of study both in statistics in general and in causal identification research [7].

We will show how classical and quantum Markovian causal models with appropriately restricted common-cause subtheories can be identified when the accessible set of instruments at each locus is a complete set of $\circ$-separable instruments.

6 Quantum common causes and convolution of maps

In this section, we will establish that autonomous quantum channels satisfy a quantum version of Eq. 8, where the quantum meaning of the "copy" dot will be given. Ultimately Eq. 8 will be exploited for identification of both classical and quantum Markovian causal models.

We pass from **CPM** to the larger process theory **FVect** of all linear maps to introduce super-operators that are not completely positive but will be used for diagrammatic quantum causal inference. First, we define super-operators

$$\mu : L(\mathcal{H}_A) \otimes L(\mathcal{H}_A) \to L(\mathcal{H}_A) \qquad \delta : L(\mathcal{H}_A) \to L(\mathcal{H}_A) \otimes L(\mathcal{H}_A)$$

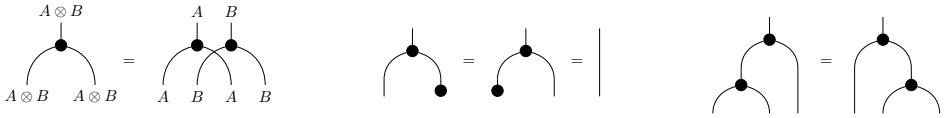
where μ corresponds to matrix multiplication, i.e., $\mu(\rho \otimes \sigma) := \rho\sigma$, and δ is its adjoint with respect to the Hilbert-Schmidt inner product. The latter is easiest to describe concretely by its action on basis elements written in Dirac's "bra-ket" notation (see Appendix B):

$$\delta(|i\rangle\langle j|) := \sum_k |i\rangle\langle k| \otimes |k\rangle\langle j|$$

Now we introduce diagrammatic notation like that used in the classical case. We'll write:

$\mu :=$  $\delta :=$  $I_{\mathcal{H}_A} :=$  $\mathrm{tr}_{\mathcal{H}_A} :=$  $\boldsymbol{d}_{A,B} =$ 

It is straightforward to show a few basic identities using these generators, e.g.,



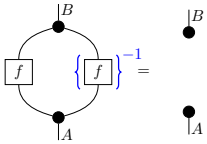
and their (vertical) mirror-images, which imply that δ and $\mathrm{tr}_{\mathcal{H}_A}$ give every object of the form $L(\mathcal{H}_A)$ in **FVect** a comonoid structure. Here, in contrast to the classical case, the comonoid structure is non-cocommutative, i.e., Eq. 3 does not hold. This structure has been described in [6].

The classical maps depicted by  and are the matrix transposes of those depicted by the already-defined vertical mirror images of the respective diagrams.

The following two definitions are the diagrammatic equivalents of those for the same terms in Appendix B.

▶ **Definition 24.** *The convolution* $\Phi_1 * \Phi_2$ *of two quantum or classical maps* $\Phi_1, \Phi_2 : A \to B$ *is* $\mu \circ (\Phi_1 \otimes \Phi_2) \circ \delta$.

▶ **Definition 25.** *For both* ***FVect*** *and* ***Mat***$[\mathbb{R}_+]$*, the convolution inverse of a map, indicated by that map's diagram inside* $\{-\}^{-1}$*, satisfies*



When this notation is used in calculations for Section 7, one or both of the objects A and B will be the unit object, i.e., f will be an effect or a number. This notation is consistent with the use of $\{-\}^{-1}$ for inverses of positive real numbers in the proof of Prop. 13.

With the quantum semantics for the black dot, the essential similarity between autonomous quantum channels and function-maps can be stated as follows:

▶ **Theorem 26.** *Any autonomous quantum channel A satisfies Eq. 8.*

The theorem follows from Propositions 31 and 36 in Appendix B. It implies that the common-cause maps in $(\mathbf{CPM}, \mathbf{Unitary}_\bullet)$-valued Markovian models can be rewritten just as the classical common-cause function-maps are, e.g., in the first equality of (9). This rewriting, together with convolution inverses, will yield an identification technique for $(\mathbf{CPM}, \mathbf{Unitary}_\bullet)$ models.

7 Identification

We will study the causal identification problem as described in Section 5, focusing on classical and quantum cases in which the accessible set of instruments at each locus is a complete set of $\circ$-separable instruments. Specifically, we will show, for the smallest graph in which two arrows leave a single vertex, that in both the classical and quantum settings, if the common-cause subtheory $\mathcal{C}_{cc}$ is appropriately restrictive, any complete sets of $\circ$-separable instruments at all loci suffice for identification; otherwise, generic such sets do not suffice.

▶ **Example 27.** Strictly positive Markovian models based on the graph G of Example 6 valued in $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Mat}[\mathbb{R}_+])$ or $(\mathbf{CPM}, \mathbf{Isom}_\bullet)$ are not identifiable from arbitrary complete sets of $\circ$-separable instruments.

For $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Mat}[\mathbb{R}_+])$, the statement follows from Proposition 12. For $(\mathbf{CPM}, \mathbf{Isom}_\bullet)$, an example of a pair of models that respond differently to some interventions but are indistinguishable under local projective measurement can be constructed from the example at the end of Section 3 of [9]. (The labels X and Z are swapped relative to the labeling in Example 6; the common-cause map $\boldsymbol{A_Z}$ is the parallel composite of an identity map and the state $\boldsymbol{u}$.) Here common-cause maps from $\mathbf{Isom}_\bullet$ can be thought of as potentially introducing unseen auxiliary systems that correlate outcomes at multiple loci.

If the common-cause subtheory is restricted to $\mathbf{Func}_\bullet$ in the classical case or $\mathbf{Unitary}_\bullet$ in the quantum case, complete sets of $\circ$-separable instruments at all loci become sufficient for identification.

▶ **Proposition 28.** *Markovian models based on the graph G of Example 6 valued in $(\boldsymbol{Mat}[\mathbb{R}_+], \boldsymbol{Func}_\bullet)$ or $(\boldsymbol{CPM}, \boldsymbol{Unitary}_\bullet)$ are identifiable whenever the accessible set of instruments at each locus is a complete set of $\circ$-separable instruments.*

The proof, in Appendix A, applies Theorem 26 and convolution inverses of quantum and classical maps.

8 Conclusion

This paper has addressed the problem of defining quantum Markovian graphical causal models by formulating causal models as combs and replacing the copy maps of categorical probability by "common cause" maps describing how information may be shared among intervention loci. The framework allows the formulation of causal inference problems parametrized by the process theory, common-cause subtheory, and available sets of instruments. When the quantum common-cause subtheory consists of autonomous quantum channels, Theorem 26 gives the quantum causal models a structure that can be used to identify the models even when one is given access to only highly restricted probing operations. Meanwhile, the formalism permits the study of new kinds of classical causal identification problems solvable when function-maps are taken as the classical common-cause subtheory.

References

1 John-Mark A. Allen, Jonathan Barrett, Dominic C. Horsman, Ciarán M. Lee, and Robert W. Spekkens. Quantum Common Causes and Quantum Causal Models. *Phys. Rev. X*, 7(3):031021, July 2017. doi:10.1103/PhysRevX.7.031021.

2 Howard Barnum, Carlton M. Caves, Christopher A. Fuchs, Richard Jozsa, and Benjamin Schumacher. Noncommuting Mixed States Cannot Be Broadcast. *Phys. Rev. Lett.*, 76(15):2818–2821, April 1996. Publisher: American Physical Society. doi:10.1103/PhysRevLett.76.2818.

3 Jonathan Barrett, Robin Lorenz, and Ognyan Oreshkov. Quantum Causal Models, 2019. doi:10.48550/arXiv.1906.10726.

4 G. Chiribella, G. M. D'Ariano, and P. Perinotti. Quantum Circuit Architecture. *Phys. Rev. Lett.*, 101(6):060401, August 2008. Publisher: American Physical Society. doi:10.1103/PhysRevLett.101.060401.

5 Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017. doi:10.1017/9781316219317.

6 Bob Coecke and Robert W. Spekkens. Picturing classical and quantum Bayesian inference. *Synthese*, 186(3):651–696, June 2012. doi:10.1007/s11229-011-9917-5.

7 Juan D. Correa, Jin Tian, and Elias Bareinboim. Identification of Causal Effects in the Presence of Selection Bias. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):2744–2751, July 2019. Section: AAAI Technical Track: Knowledge Representation and Reasoning. doi:10.1609/aaai.v33i01.33012744.

8 Fabio Costa and Sally Shrapnel. Quantum causal modelling. *New Journal of Physics*, 18(6):063032, June 2016. Publisher: IOP Publishing. doi:10.1088/1367-2630/18/6/063032.

9 Isaac Friend and Aleks Kissinger. Identification of Causal Influences in Quantum Processes. In Stefano Gogioso and Matty Hoban, editors, *Proceedings 19th International Conference on Quantum Physics and Logic, Wolfson College, Oxford, UK, 27 June - 1 July 2022*, volume 394 of *Electronic Proceedings in Theoretical Computer Science*, pages 101–115. Open Publishing Association, 2023. doi:10.4204/EPTCS.394.7.

10 Isaac Friend and Aleks Kissinger. Identification of causal influences in quantum processes. *Physical Review A*, 109(4):042214, April 2024. Publisher: American Physical Society. doi:10.1103/PhysRevA.109.042214.

11 Tobias Fritz. A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370:107239, August 2020. doi:10.1016/j.aim.2020.107239.

12 Christina Giarmatzi and Fabio Costa. A quantum causal discovery algorithm. *npj Quantum Information*, 4(1):17, March 2018. doi:10.1038/s41534-018-0062-6.

13 Bart Jacobs, Aleks Kissinger, and Fabio Zanasi. Causal Inference by String Diagram Surgery. In Mikołaj Bojańczyk and Alex Simpson, editors, *Foundations of Software Science and Computation Structures*, pages 313–329, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-17127-8_18.

14 Bart Jacobs and Fabio Zanasi. The Logical Essentials of Bayesian Reasoning. *CoRR*, abs/1804.01193, 2018. arXiv: 1804.01193. arXiv:1804.01193.

15 M. S. Leifer and Robert W. Spekkens. Towards a formulation of quantum theory as a causally neutral theory of Bayesian inference. *Phys. Rev. A*, 88(5):052130, November 2013. Publisher: American Physical Society. doi:10.1103/PhysRevA.88.052130.

16 Judea Pearl. *Causality*. Cambridge University Press, 2 edition, 2009. doi:10.1017/CBO9780511803161.

17 Katja Ried, Megan Agnew, Lydia Vermeyden, Dominik Janzing, Robert W. Spekkens, and Kevin J. Resch. A quantum advantage for inferring causal structure. *Nature Physics*, 11(5):414–420, May 2015. doi:10.1038/nphys3266.

18 Benjamin Schumacher and Michael D. Westmoreland. Locality and Information Transfer in Quantum Operations. *Quantum Information Processing*, 4(1):13–34, February 2005. doi:10.1007/s11128-004-3193-y.

19 Sally Shrapnel, Fabio Costa, and Gerard Milburn. Quantum Markovianity as a supervised learning task. *International Journal of Quantum Information*, 16(08):1840010, 2018. doi:10.1142/S0219749918400105.

20 W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, October 1982. doi:10.1038/299802a0.

A Identifiability proofs

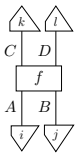
In our study of identifiability conditions, we always assume that all models, whether classical or quantum, are *strictly positive* in the following sense:

▶ **Definition 29.** *An interventional causal model based on graph G is called strictly positive if for each generating map of the form $x : e_1 \otimes \ldots \otimes e_j \to X$ in $\boldsymbol{G}$, in the case $\mathcal{C} = \boldsymbol{Mat}[\mathbb{R}_+]$ the stochastic matrix $F(x)$ has only strictly positive entries, or in the case $\mathcal{C} = \boldsymbol{CPM}$ the quantum channel $F(x)$ has full Choi rank.*

The Choi rank of a quantum channel is defined in terms of the channel's Choi matrix, discussed in Appendix B. When a strictly positive classical or quantum model is composed with any non-zero state and any non-zero effect, the result is a strictly positive real number. Our strict positivity assumption serves a similar purpose to that of standard requirements of strictly positive distributions in causal inference, which guarantee that relevant conditional probabilities are defined; here the guarantee is that scalars inverted in our identification protocols are in fact non-zero, and more generally that effects have "convolution inverses."

One common feature of $\mathbf{Mat}[\mathbb{R}_+]$ and $\mathbf{CPM}$ that will help with causal identification is *local process tomography.*

▶ **Proposition 30.** *The theories $\boldsymbol{Mat}[\mathbb{R}_+]$ and $\boldsymbol{CPM}$ have local process tomography: any process $f : A \otimes B \to C \otimes D$ is determined by numbers*



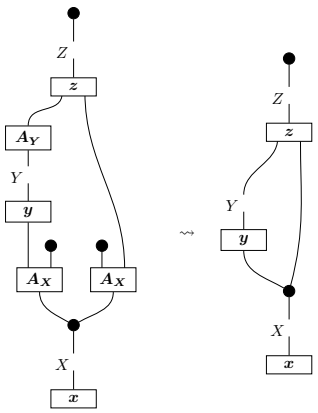
(10)

where i, j, k, and l index any informationally complete sets of states or effects for the appropriate objects.

We will often leave the interpretation functor F from the syntactic process theory $\mathbf{G}$ into the semantic process theory implicit and use boldface to distinguish abstract processes in $\mathbf{G}$ from their images under F, writing, e.g., $\boldsymbol{x} := F(x)$. Labels for objects/intervention loci will be identical between the syntactic and semantic process theories, since the distinction will already be clear from the labels for processes.

Proof of Proposition 13. The proposition can be proven with techniques from [13], via the equivalence we have discussed between our $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued Markovian models and the causal Bayesian networks formulated in that article. Here, however, we prove the proposition only for one graph, so that the procedure can be compared directly with the one in Section 7 for quantum and classical identification from generalized observation.

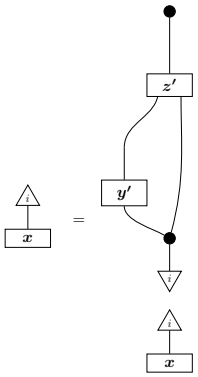
An unknown $(\mathbf{Mat}[\mathbb{R}_+], \mathbf{Func}_\bullet)$-valued Markovian model based on the graph G in Example 6 is a functorial interpretation of the abstract causal structure c_G in that example. Since the common-cause subtheory is $\mathbf{Func}_\bullet$, Eq. (8) applies to the common-cause maps, which can then be absorbed into larger processes $\boldsymbol{y'}$ and $\boldsymbol{z'}$, resulting in a new representation for the unknown model:



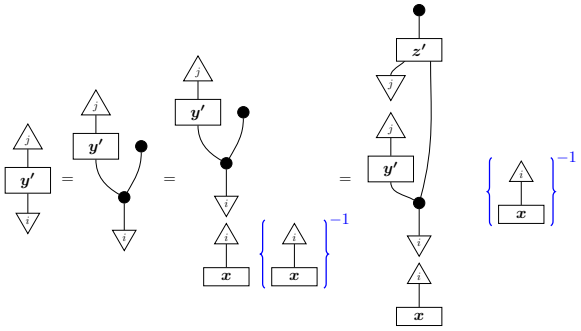
(11)

Inferring the values of the processes $\boldsymbol{x}$, $\boldsymbol{y'}$, and $\boldsymbol{z'}$ on the right-hand side–which one does by inferring all their matrix entries–is equivalent to inferring the process $F(c_G)$.

One first computes the processes $\boldsymbol{x}$ and $\boldsymbol{y'}$ by determining all their matrix entries. For $\boldsymbol{x}$, the numbers

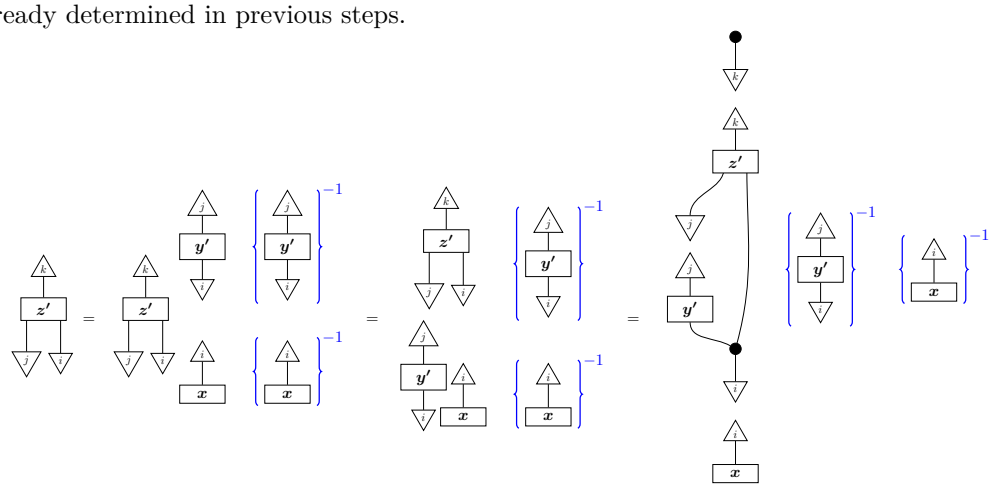


are obtained via a local intervention regime consisting of perfect passive observation at locus X and identity interventions at Y and Z. The process $\boldsymbol{y'}$ is tomographically determined as follows:



The desired quantity has been rewritten as the product of a probability obtained via perfect passive observation at X and Y (and identity intervention at Z) and a number obtainable from the known value of $\boldsymbol{x}$.

Finally, $\boldsymbol{z'}$ is computed from probabilities obtained via a local intervention regime consisting of perfect passive observation at all three loci, and from inverses of numbers already determined in previous steps.

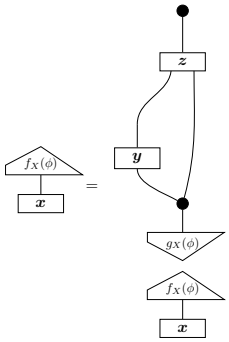


◀

Proof of Proposition 28. As in the demonstration of Prop. 13, but now for both the classical and quantum cases, Eq. 8 and defining new unknown processes $\boldsymbol{y'}$ and $\boldsymbol{z'}$ lead to a simplification of the unknown model as shown in expression (11). To identify the model, one proceeds as before to compute the processes $\boldsymbol{x}$ and $\boldsymbol{y'}$ by determining the probabilities given by their composition with appropriate informationally complete sets of states and effects.

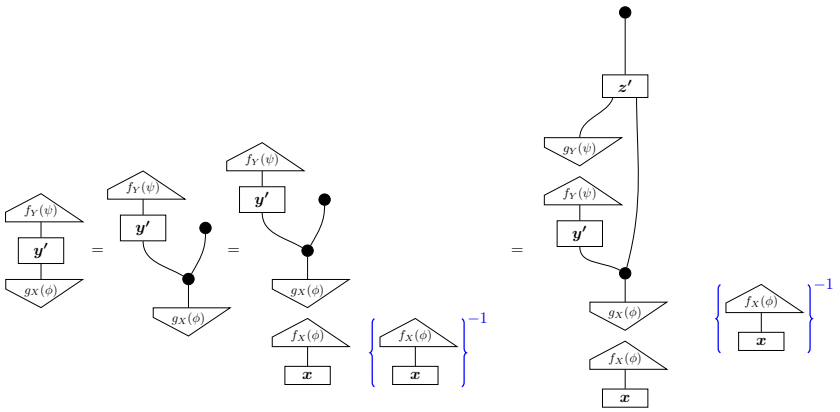
The union of the accessible set of instruments at a locus, say X, is a set of maps, indexed by, say, ϕ. Each map ϕ is composed of an effect $f_X(\phi)$ and a state $g_X(\phi)$, where f_X and g_X are functions associated with locus X. Marginal informational completeness of the set of instruments means that the set $\{f_X(\phi)\}_\phi$ of effects and the set $\{g_X(\phi)\}_\phi$ of states are each informationally complete for system-type X.

One determines $\boldsymbol{x}$ by learning for the informationally complete set of effects $f_X(\phi)$ the probability



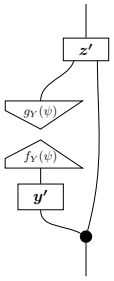
The right-hand diagram is a probability learned from probing with $\circ$-separable instruments at locus X and identity instruments elsewhere. In contrast to the case of classical perfect passive observation, inferring the value of $\boldsymbol{x}$ now might involve more than one local intervention regime, so that X can be probed with multiple instruments.

Next, one proceeds to determine $\boldsymbol{y'}$ tomographically as in the proof of Prop. 13, but in general collating data from multiple local intervention regimes:



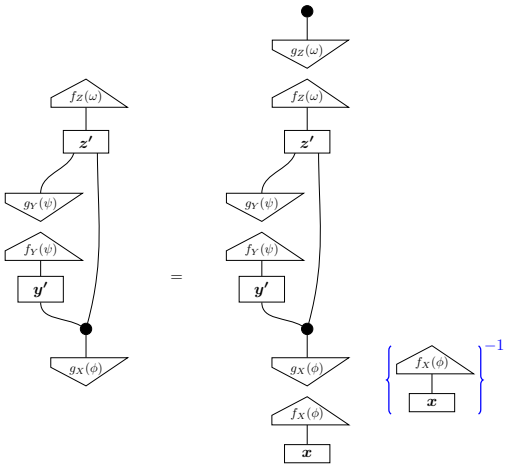
At this point, in the case of classical perfect passive observation, the computation of $\boldsymbol{z}$ in the proof of Prop. 13 uses the fact that the classical "copy" map literally copies the pure states that leave a locus after a perfect observation. For quantum measurements, even maximally informative projective measurements, and for generalized classical observation, the state leaving the first locus is not copiable, and hence this calculation isn't available.

In this general case, once one has computed the values of $\boldsymbol{x}$ and $\boldsymbol{y'}$, one can tomographically determine the value of the process



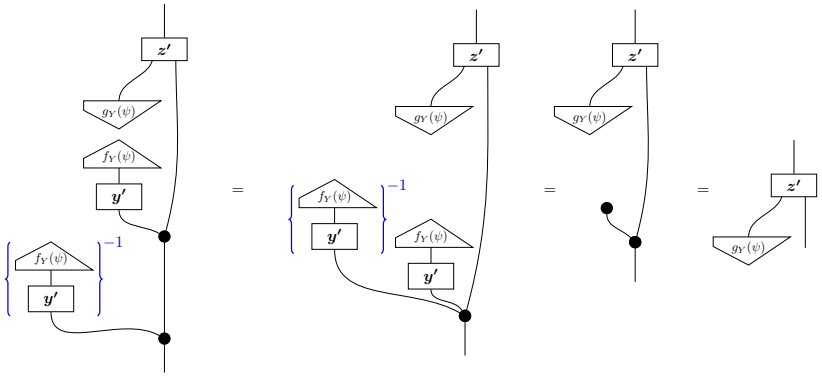
(12)

for each map ψ in a marginally informationally complete set, from the numbers



which one can learn for informationally complete sets of states $g_X(\phi)$ and effects $f_Z(\omega)$.

Knowing also the value of $\boldsymbol{y'}$, one can compute for each ψ the convolution inverse of $f_Y(\psi) \circ \boldsymbol{y'}$, and compose it with the process (12) as follows:



One now knows the latter process for an informationally complete set of states $g_X(\psi)$, and hence obtains the value of the process $\boldsymbol{z'}$. The known processes $\boldsymbol{x}$, $\boldsymbol{y'}$, and $\boldsymbol{z'}$ can now be composed to form the entire data-generating process in expression (11), completing the causal inference. ◀

B Choi-Jamiołkowski isomorphism and channel convolution

The Choi-Jamiołkowski isomorphism gives a bijective correspondence between linear super-operators $\mathcal{E} : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$ and linear maps $\rho^{\mathcal{E}} : \mathcal{H}_B \otimes \mathcal{H}_A^* \to \mathcal{H}_B \otimes \mathcal{H}_A^*$. The linear map $\rho^{\mathcal{E}}$, called the *Choi matrix* of $\mathcal{E}$, can be defined explicitly in terms of a basis $\{|i\rangle_A\} \subset \mathcal{H}_A$ and its dual basis $\{|i\rangle_{A^*}\} \subset \mathcal{H}_A^*$ as follows:

$$\rho^{\mathcal{E}} := \sum_{ij} \mathcal{E}(|i\rangle_A\langle j|) \otimes |i\rangle_{A^*}\langle j| \tag{13}$$

Here we have used Dirac's "bra-ket" notation to write operators/matrices as products of basis vectors ("kets" $|i\rangle_A$) and their associated dual vectors ("bras" ${}_A\langle i|$).

The Choi-Jamiołkowski isomorphism states that $\mathcal{E}$ is completely positive if and only if $\rho^{\mathcal{E}}$ is positive. Now, for a quantum channel $\Phi : L(\mathcal{H}_A) \to L(\mathcal{H}_B) \otimes L(\mathcal{H}_C)$ define the following three positive operators:

$$\rho_{BC|A} := \rho^{\Phi} \qquad \rho_{B|A} := I_{\mathcal{H}_C} \otimes \mathrm{tr}_{\mathcal{H}_C}(\rho^{\Phi}) \qquad \rho_{C|A} := I_{\mathcal{H}_B} \otimes \mathrm{tr}_{\mathcal{H}_B}(\rho^{\Phi}) \tag{14}$$

We can regard each of these as an operator on $\mathcal{H}_B \otimes \mathcal{H}_C \otimes \mathcal{H}_A^*$ (note that we have suppressed "swap" maps above).

Theorem 2 in [1] implies the following, which will be used to establish Eq. 8 for autonomous quantum channels[6].

▶ **Proposition 31.** *If Φ is an autonomous quantum channel, then it satisfies*

$$\rho_{BC|A} = \rho_{B|A}\rho_{C|A} \tag{15}$$

We therefore proceed to study channels satisfying Eq. 15.

B.1 Channel convolution

Satisfying Eq. 15 will be shown equivalent to decomposing in a certain way with respect to the following convolution operation for superoperators. For (not necessarily completely positive) linear maps $\Phi_1, \Phi_2 : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$, let $\Phi_1 * \Phi_2 : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$ be a new linear map defined on basis elements $|i\rangle\langle j| \in L(\mathcal{H}_A)$ as follows:

$$\Phi_1 * \Phi_2(|i\rangle\langle j|) := \sum_k \Phi_1(|i\rangle\langle k|)\Phi_2(|k\rangle\langle j|)$$

First, we show that the Choi-Jamiołkowski isomorphism carries this operation to matrix multiplication.

▶ **Lemma 32.** *Let $\rho^{\Phi_1}, \rho^{\Phi_2}$, and $\rho^{\Phi_1 * \Phi_2}$ be the Choi matrices of the super-operators Φ_1, Φ_2, and $\Phi_1 * \Phi_2$, respectively. Then $\rho^{\Phi_1}\rho^{\Phi_2} = \rho^{\Phi_1 * \Phi_2}$.*

Proof. Unroll (13) and simplify. ◀

Although the convolution of an arbitrary pair of completely positive maps need not be completely positive, the convolution of a pair of completely positive maps that commute under convolution is completely positive:

[6] Theorem 2 in [1] is used in that article to motivate a definition of quantum Markovianity based on the idea that directed edges in a graph should indicate signaling relations between input and output systems of a unitary quantum channel.

▶ **Corollary 33.** *For completely positive maps $\Phi_1, \Phi_2 : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$, the super-operator $\Phi_1 * \Phi_2$ is completely positive if and only if $\Phi_1 * \Phi_2 = \Phi_2 * \Phi_1$.*

Proof. Suppose $\Phi_1 * \Phi_2 = \Phi_2 * \Phi_1$. Then, using Theorem 32, we have: $\rho^{\Phi_1}\rho^{\Phi_2} = \rho^{\Phi_1 * \Phi_2} = \rho^{\Phi_2 * \Phi_1} = \rho^{\Phi_2}\rho^{\Phi_1}$. So $\rho^{\Phi_1 * \Phi_2}$ is the product of commuting positive operators ρ^{Φ_1} and ρ^{Φ_2}, and hence positive itself. Therefore $\Phi_1 * \Phi_2$ is completely positive. Conversely, if $\Phi_1 * \Phi_2$ is completely positive then $\rho^{\Phi_1 * \Phi_2}$ is positive. Hence, by Theorem 32, $\rho^{\Phi_1}\rho^{\Phi_2}$ is also positive, which is only possible if the positive operators ρ^{Φ_1} and ρ^{Φ_2} commute. This in turn implies that $\rho^{\Phi_1 * \Phi_2} = \rho^{\Phi_2 * \Phi_1}$. By an inverse application of the Choi-Jamiołkowski isomorphism, we conclude that $\Phi_1 * \Phi_2 = \Phi_2 * \Phi_1$. ◀

Now, we can get a fully channel-based version of Eq. 15. For Hilbert spaces $\mathcal{H}_A, \mathcal{H}_B$, we define the (un-normalized) depolarizing channel as follows for all states $\rho \in L(\mathcal{H}_A)$:

$$\boldsymbol{d}_{A,B}(\rho) = \mathrm{tr}(\rho) I_{\mathcal{H}_B}$$

This channel is equivalent to the identity operator under the Choi-Jamiołkowski isomorphism, so it behaves as a unit for channel convolution.

For a channel $\Phi_{BC|A} := \Phi$, we define the reduced channels $\Phi_{B|A}$ and $\Phi_{C|A}$ simply by applying $\boldsymbol{d}$ to the appropriate output:

$$\Phi_{B|A} := (1_{\mathcal{H}_B} \otimes \boldsymbol{d}_{\mathcal{H}_C}) \circ \Phi \qquad \Phi_{C|A} = (\boldsymbol{d}_{\mathcal{H}_B} \otimes 1_{\mathcal{H}_C}) \circ \Phi$$

One can straightforwardly check that the Choi matrices of these channels are the reduced states in (14). From that fact and Lemma 32, we can immediately conclude:

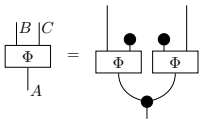
▶ **Lemma 34.** *A channel $\Phi : L(\mathcal{H}_A) \to L(\mathcal{H}_B) \otimes L(H_C)$ has Choi matrices satisfying Eq. 15 if and only if $\Phi_{BC|A} = \Phi_{B|A} * \Phi_{C|A}$.*

▶ **Definition 35.** *The convolution inverse of a completely positive map Φ is a linear map $\Phi^{(-1)}$ satisfying $\Phi * \Phi^{(-1)} = \Phi^{(-1)} * \Phi = \boldsymbol{d}_{A,B}$.*

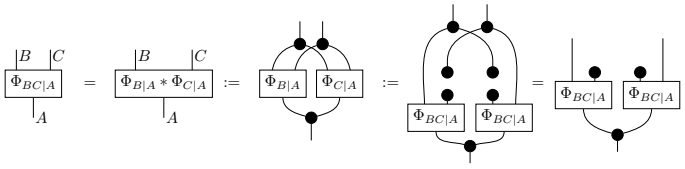
A completely positive map has a convolution inverse if and only if its associated Choi matrix ρ_Φ is invertible; in that case, the convolution inverse is the linear map defined by the usual matrix inverse ρ_Φ^{-1} under the Choi-Jamiołkowski isomorphism. For classical positive matrices, we can define the convolution inverse similarly, and it is given concretely by the positive matrix whose elements are $(M^{(-1)})_{i,j} := 1/M_{i,j}$.

From the graphical rules for convolution in Section 6, we can derive the condition for a quantum map to satisfy Eq. 8:

▶ **Proposition 36.** *The Choi matrix of channel $\Phi : L(\mathcal{H}_A) \to L(\mathcal{H}_B) \otimes L(\mathcal{H}_C)$ factors according to Eq. 15 if and only if*



Proof. Writing Φ as $\Phi_{BC|A}$, we apply Lemma 34, then simplify:



◀

Minimality in Finite-Dimensional ZW-Calculi

Marc de Visme
Université Paris-Saclay, Inria, CNRS, ENS Paris-Saclay,
Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France

Renaud Vilmart
Université Paris-Saclay, Inria, CNRS, ENS Paris-Saclay,
Laboratoire Méthodes Formelles, 91190, Gif-sur-Yvette, France

Abstract

The ZW-calculus is a graphical language capable of representing 2-dimensional quantum systems (qubit) through its diagrams, and manipulating them through its equational theory. We extend the formalism to accommodate finite dimensional Hilbert spaces beyond qubit systems.

First we define a qu**d**it version of the language, where all systems have the same arbitrary finite dimension d, and show that the provided equational theory is both complete – i.e. semantical equivalence is entirely captured by the equations – and minimal – i.e. none of the equations are consequences of the others. We then extend the graphical language further to allow for mixed-dimensional systems. We again show the completeness and minimality of the provided equational theory.

2012 ACM Subject Classification Theory of computation → Quantum computation theory; Theory of computation → Equational logic and rewriting; Theory of computation → Semantics and reasoning

Keywords and phrases Quantum Computing, Categorical Quantum Mechanics, ZW-calculus, Qudits, Finite Dimensional Hilbert Spaces, Completeness, Minimality

Digital Object Identifier 10.4230/LIPIcs.CSL.2025.49

Related Version *Full version with proofs*: `https://arxiv.org/abs/2401.16225`

Funding This work is supported by the PEPR integrated project EPiQ ANR-22-PETQ-0007 part of Plan France 2030, by the projects ANR-22-PNCQ-0002 and ANR-22-CE47-0012.

Acknowledgements The authors would like to thank Antoine Guilmin-Crépon for discussions about the minimality of the present equational theories. The diagrams of the present paper were drawn using the TikZit tool [33].

1 Introduction

Graphical languages for quantum computations are a product of the categorical quantum mechanics program [1,15] devoted to studying the foundations of quantum mechanics through the prism of category theory. These graphical languages come in different flavours, depending on which generators are used to build the diagrams (graphical representations of the quantum operators), and critically, displaying different kinds of interactions between said generators. The ZX-calculus describes the interaction between two complementary bases [13], the ZW-calculus, the interaction between the two "spiders" derived from the "GHZ" and "W" states, the only two fully entangled tripartite states up to SLOCC-equivalence [14], and the ZH-calculus the interaction between the same GHZ-state inferred spider and a spider obtained by generalising the Hadamard gate [3].

The equations that describe these interactions form "equational theories" that define syntactic equivalence classes of diagrams, that are also semantically equivalent. When the syntactic equivalence matches perfectly the semantic one (i.e. when two diagrams represent the same quantum operator iff they can be turned into one another), we say that the equational

theory is *complete*. Complete equational theories have been found for the aforementioned graphical languages, betimes for restrictions of them [2, 4, 10, 11, 24, 26, 29, 31, 47].

As is customary for the computer science part of the quantum computation community, the focus was largely set onto *qubit* systems, i.e. systems where the base quantum system is 2-dimensional, yet this is enough to get applications in optimisation [5, 34], quantum error correction [18, 28, 46], verification [22, 27], simulation [35–37]... However, physics allows for qudit systems (where the base quantum system is d-dimensional with $d > 2$) and even infinite dimensional systems. Several attempts have hence been made to go beyond the qubit case [7, 25, 48], but it was only recently that a complete equational theory was found for d-dimensional (i.e. qudit) systems [42] and later for finite dimensional systems (so-called "qufinite", i.e. for the category **FdHilb**) [50]. The results were obtained by generalising both the ZX and the ZW calculi and mixing them together. The W-node in particular allows for a neat intuitive (and unique) normal form for the diagrams. Satisfying the necessary conditions for every diagram to be normalisable then yields a complete equational theory. However, we believe that the ones obtained in [42, 50] are far from being *minimal*, due in particular to the presence of the generators from both the ZX and the ZW calculi.

From a foundational perspective, it can be enlightening to know if an equation is a defining property of quantum systems (and hence necessary), or on the contrary if it is derivable from more fundamental properties (see e.g. [16, 21]). The redundancy in the equational theory may also cause issues when trying to explore the space of equivalent diagrams, or to transport the completeness result to other diagrammatic languages for qudit systems (every equation has to be proven in the new language, hence the fewer the better); or when trying to generalise further, e.g. to the **FdHilb** setting.

We argue here that the ZW-calculus is enough to get a natural normal form (akin to that of [42]) even in the qudit version, and provide an elegant equational theory that we show to be *complete*, resorting to the normal form instead of transporting the completeness result from [42], for the reason described above. We also show that the equational theory is minimal, meaning that none of the equations can be derived from the others, hence avoiding the aforementioned redundancy in the presentation.

We then adapt diagrams and the equational theory of the graphical language to accommodate all finite dimensional Hilbert spaces (**FdHilb**), in a way that requires no additional generator and only one new equation. Here again we prove the completeness and the minimality of the equational theory, by leveraging that of the qudit setting.

The paper is split into two parts, Section 2 and Section 3, devoted respectively to the $\mathbf{Qudit}_d$ version, and to the **FdHilb** version. In the $\mathbf{Qudit}_d$ version, diagrams and their interpretation are introduced in Section 2.1 and the equational theory is introduced and discussed in Section 2.2. We then show its minimality in Section 2.3 and its completeness in Section 2.4. In the **FdHilb** version, diagrams and their interpretation are introduced in Section 3.1, and the equational theory is introduced and shown to be complete in Section 3.2. All missing proofs are provided in the full version.

The Dirac Notation

All the upcoming diagrams can be given an interpretation as a linear map, in the appropriate category. In quantum information, it is usual to express such linear maps using the so-called Dirac notation. The current section hence serves as a gentle introduction to this notation.

Let $d \geq 2$. In the d-dimensional Hilbert space $\mathbb{C}^d$, the canonical basis

$$\left\{ \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^\intercal, \begin{pmatrix} 0 & 1 & \cdots & 0 \end{pmatrix}^\intercal, \ldots, \begin{pmatrix} 0 & 0 & \cdots & 1 \end{pmatrix}^\intercal \right\}$$

is usually denoted $\{|0\rangle, |1\rangle, ..., |d-1\rangle\}$ (with $(.)^\intercal$ being the transpose). All 1-qudit systems have states that live in $\mathbb{C}^d$ and that can hence be represented by linear combinations of the elements of this basis: $|\psi\rangle = a_0 |0\rangle + a_1 |1\rangle + ... + a_{d-1} |d-1\rangle$ (notice that the "ket" notation $|.\rangle$ is used for states in general, not only basis elements).

To combine systems, we use the tensor product (Kronecker product): $(. \otimes .)$ which is a fairly standard operation on linear maps. In particular, the overall state obtained by composing two 1-qudit systems in respective states $|\psi\rangle$ and $|\varphi\rangle$ is simply $|\psi\rangle \otimes |\varphi\rangle$. Notice that $\{|i\rangle \otimes |j\rangle\}_{0\leq i<d, 0\leq j<d'}$ forms a basis of $\mathbb{C}^d \otimes \mathbb{C}^{d'} \simeq \mathbb{C}^{d\times d'}$. It is customary to write $|\psi, \varphi\rangle$ to abbreviate $|\psi\rangle \otimes |\varphi\rangle$.

The "bra" notation $\langle .|$ is used to represent the dagger (the conjugate transpose) of a state, i.e. $\langle\psi| = |\psi\rangle^\dagger = \overline{|\psi\rangle}^\intercal$. The choice of the "bra-ket" notation is such that composing a bra with ket forms the bracket, the usual inner product in $\mathbb{C}^d$: $\langle\psi| \circ |\varphi\rangle = \langle\psi|\varphi\rangle$. Linear combinations of "ket-bras" $|i\rangle\langle j|$ of the canonical basis can be used to represent any linear map of the correct dimensions, e.g. the 1-qudit identity: $id = \sum_{k=0}^{d-1} |k\rangle\langle k|$.

2 ZW-Calculus for Qudit Systems

In this section, we introduce a graphical language for quantum systems that all have the same fixed dimension d ($d \geq 2$): qudit systems.

2.1 Diagrams of ZW_d and their Interpretation

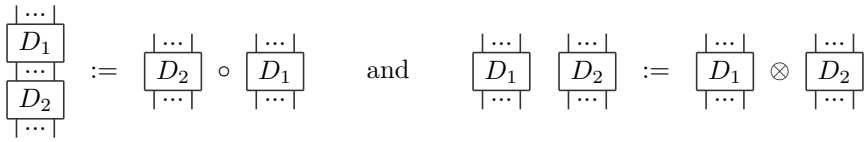
First, we need to introduce the mathematical objects at the heart of the graphical language – the diagrams – and what they represent.

As is traditional for graphical languages for finite-dimensional quantum systems, we work with a †-compact prop [38,45,51]. Categorically speaking, this is a symmetric, compact closed monoidal category generated by a single object, endowed with a contravariant endofunctor that behaves well with the symmetry and the compact structure. The following explains some of these concepts in more detail.

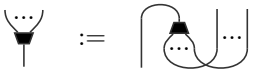
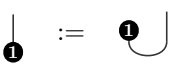
Let us denote $\mathbf{ZW}_d$ the †-compact prop generated by:

- the Z-spiders  $: n \to m$, for $r \in \mathbb{C}$ and $n, m \geq 0$
- the W-nodes  $: 1 \to n$, for $n \geq 0$
- the state $|1\rangle$  $: 0 \to 1$
- the global scalars $r : 0 \to 0$, for $r \in \mathbb{C}$
- the swap $: 2 \to 2$ representing the symmetry of the prop
- the cup  $: 2 \to 0$ and cap  $: 0 \to 2$ representing the compact structure
- and the identity  $: 1 \to 1$.

All these generators can be composed sequentially and in parallel, as follows:

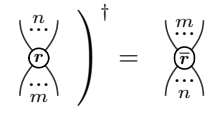


The symmetry and the compact structure satisfy the following identities:

This compact structure in particular allows us to define the "upside-down" version of the generators, for instance:  and 

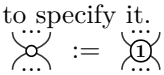
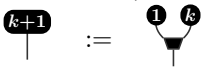
The † functor is defined inductively as:

$$(D_2 \circ D_1)^\dagger = D_1^\dagger \circ D_2^\dagger \qquad (D_1 \otimes D_2)^\dagger = D_1^\dagger \otimes D_2^\dagger \qquad (r)^\dagger = \bar{r}$$



with the other generators being mapped to their upside-down version.

Notice that thank to the identities satisfied by the †-compact prop, the †-functor is involutive.

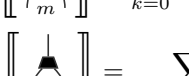
As will be made clearer in what follows, in $\mathbf{ZW}_d$, $d \geq 2$ represents the dimension of the "base" quantum system, called qudit. As this d will be fixed in the following, we may forget to specify it. For convenience, we define an empty white node as a parameter-1 Z-spider:  and give the $0 \to 1$ W-node a special symbol, akin to that of $|1\rangle$ (as its interpretation, as we will see later, is merely $|0\rangle$):  . We generalise the ket symbol inductively as follows (for $2 \leq k < d$):  . These last symbols can be given an upside-down definition using the compact structure as was done for $|1\rangle$ and the W-node.

2.1.1 The Interpretation

The point of the diagrams of the $\mathbf{ZW}_d$ is to represent quantum operators on multipartite d-dimensional systems. The way those are usually specified is thanks to the category $\mathbf{Qudit}_d$. This forms again a symmetric †-compact prop, where the base object is $1 := \mathbb{C}^d$, and morphisms $n \to m$ are linear maps $\mathbb{C}^{d^n} \to \mathbb{C}^{d^m}$. The symmetry and the compact structure correspond to their counterparts in $\mathbf{ZW}_d$, they will be stated out in the following, as part of the interpretation of $\mathbf{ZW}_d$ diagrams. The † functor is the usual † of linear maps over $\mathbb{C}$.

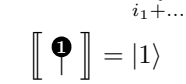
We may hence interpret diagrams of the $\mathbf{ZW}_d$-calculus thanks to the functor $[\![.]\!] : \mathbf{ZW}_d \to \mathbf{Qudit}_d$ inductively defined as follows:

$$[\![D_2 \circ D_1]\!] = [\![D_2]\!] \circ [\![D_1]\!] \qquad [\![D_1 \otimes D_2]\!] = [\![D_1]\!] \otimes [\![D_2]\!]$$

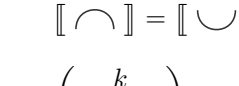


$$= \sum_{k=0}^{d-1} r^k \sqrt{k!}^{n+m-2} \, |k^m\rangle\langle k^n|$$

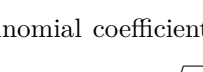
$$[\![\, | \,]\!] = \sum_k |k\rangle\langle k|$$



$$= \sum_{\substack{k \in \{0,\dots,d-1\} \\ i_1+\dots+i_n=k}} \sqrt{\binom{k}{i_1, \dots, i_n}} \, |i_1, \dots, i_n\rangle\langle k|$$

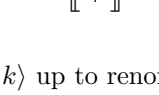


$$= \sum_{k,\ell} |\ell, k\rangle\langle k, \ell|$$



$$= |1\rangle$$

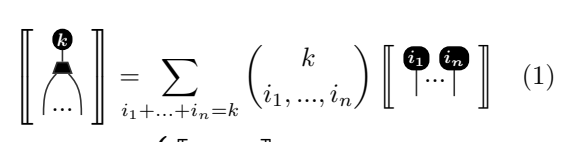
$$[\![\cap]\!] = [\![\cup]\!]^\dagger = \sum_k |k, k\rangle \qquad [\![r]\!] = r$$

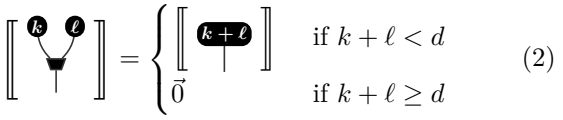
where $\binom{k}{i_1, \dots, i_n} = \frac{k!}{i_1! \dots i_n!}$ is a multinomial coefficient. Notice that the interpretation of the $0 \to 1$ W-node is simply:  $= \sum_{\substack{k \in \{0,\dots,d-1\} \\ 0=k}} \sqrt{\binom{k}{0}} \, |k\rangle = |0\rangle$, and that of the black node symbol k for $k < d$ is $|k\rangle$ up to renormalisation:  $= \sqrt{\binom{k}{1, \dots, 1}} \, |k\rangle = \sqrt{k!} \, |k\rangle$. The

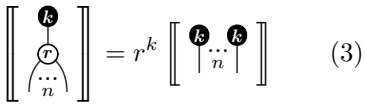
presence of $\sqrt{\cdots}$ on the coefficients is not particularly relevant, and is simply an artefact of us maintaining some symmetry between generators and their dagger. Indeed, we want  $= k!$, and for that the coefficient $k!$ needs to be split between both nodes, resulting in either an asymmetric presentation or a square root (see Section A for an equivalent semantics, without any $\sqrt{\cdots}$ but asymmetric instead).

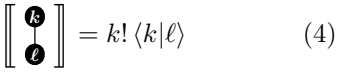
Notice also that the interpretation of the Z-spider differs from more usual generalisations of its qubit counterpart, because of the $\sqrt{k!}^{n+m-2}$ which depends on the degree of the spider. While it makes the interpretation of the diagrams slightly more complicated, it allows us – as will be stated later – to quite conveniently generalise equations from the qubit ZW-calculus, and hence have a simpler equational theory. It will be shown in the following (Corollary 10), that the above set of generators makes for a universal calculus, i.e. any linear map of $\mathbf{Qudit}_d$ can be represented by a $\mathbf{ZW}_d$-diagram.

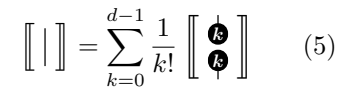
To gain intuition about the upcoming equations between diagrams, it can be useful to *semantically* decompose a diagram into sums of simpler ones[1]. To do so, it can be convenient to understand $|k\rangle$ as a bunch of k indistinguishable particles:











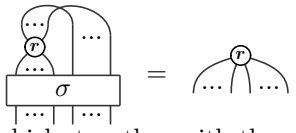
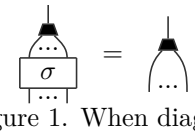
Equation (1) explains how the W-node spreads the k "particles" that enter it following a multinomial distribution. Equation (2) shows that the $2 \to 1$ W-node takes two bunches of particles k and ℓ and regroups them into one, and yields the null state if $k + \ell$ exceeds the "capacity" (i.e. the dimension) of a single wire. This will be proven graphically (Lemma 30) from the upcoming equational theory (Figure 1). When $k + \ell < d$, the fact that there is no additional scalar is due to the rescaling of the k-dots to represent $\sqrt{k!}\,|k\rangle$. This rescaling also makes the "copy" more natural: The Z-spider $1 \to n$ copies any bunch of k particles entering it, yielding global scalar r^k in the process, as is shown by Equation (3). This will again be proven graphically (Lemma 27) from the equational theory. The rescaling, however, forces Equation (4). Finally, it can be useful to decompose the identity as a linear combination of products of kets and bras as is done in Equation (5).

2.2 Equational Theory

With the above interpretation of the $\mathbf{ZW}_d$, different diagrams may yield the same linear map. All axioms of symmetric †-compact props in particular preserve the interpretation. More generally, we may want to relate together *all* diagrams that have the same semantics. This is done through an equational theory, i.e. a set of equations that can be applied locally in a diagram without changing the semantics of the whole.

[1] Notice that here, such decompositions are merely semantical. The upcoming completeness is only interested in equivalence between single diagrams.

2.2.1 Equations of the ZW_d-Calculus

On top of the axioms of symmetric †-compact props, we assume some conventional equations about the topology of the generators, which should align with the symmetries of the symbols used to depict them. The Z-spider does not distinguish between any of its connections: it is "flexsymmetric" [9], meaning that we can interchange any of its legs without changing the semantics. Graphically, for any permutation of wires σ:  . On the other hand, the binary W-node is only co-commutative, which, together with the upcoming Equation (a), means that all the outputs of the n-ary W-node can be exchanged, i.e. for any permutation of wires σ:  . With all this in place, we can give the core of the equational theory, in Figure 1. When diagram D_1 can be turned into diagram D_2 using the rules of ZW_d, we write $\mathrm{ZW}_d \vdash D_1 = D_2$.

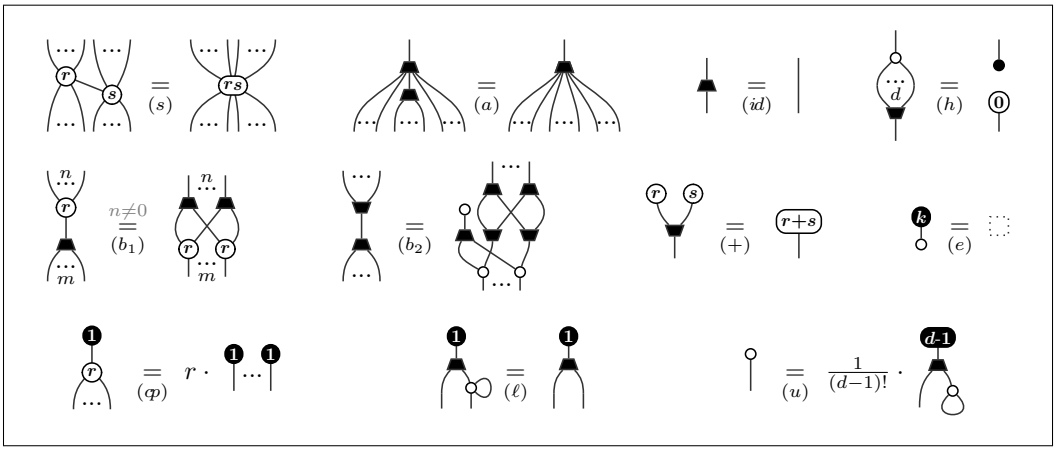
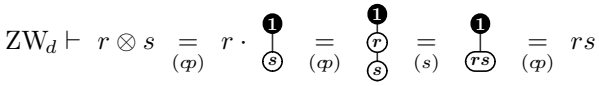


Figure 1 Equational theory ZW_d for the qudit **ZW**-calculus.

▶ Remark 1. In this framework, we can tensor global scalars together $r \otimes s$, which graphically could be confused with their product rs. This is actually unambiguous in the equational theory, as, using Equation $(\mathcal{P})$:

$\mathrm{ZW}_d \vdash$ 

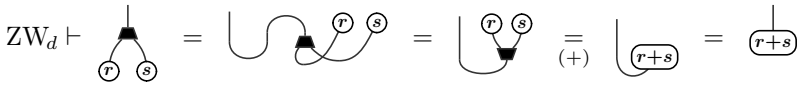
Moreover, using Equation (e), one can easily show that global scalar 1 is the empty diagram (Lemma 21). Scalar multiplication is assumed to be automatically applied, and scalar 1 is assumed to be automatically removed in the following after Lemma 21 is proven.

All rules up to (e) are fairly standard generalisations of rules of the qubit **ZW**-calculus (with (b_2) being inspired from [42] to avoid using a *fermionic swap*[2]). The non-conventional

[2] The fermionic swap, introduced in [24], is a generator that in many situations behave like the actual swap. The qubit version of ZW uses the fermionic swap, but this generator loses many properties (for instance its involution, or the fact that it maps $|k, \ell\rangle$ to $|\ell, k\rangle$ up to some coefficient) when going in larger dimensions. By not using it here, we avoid having to axiomatise it.

$\sqrt{k}^{n+m-2}$ coefficients in the interpretation of the Z-spider seem to be necessary for Equations (s), (b_1) and $(+)$ to all work. Notice that this makes the Z-spider *non-special*, meaning that:  . Equation (ℓ) however gives a context in which that inequality becomes an equality. Finally, Equation (u) shows how a $0 \to 1$ Z-spider can be obtained by distributing $d-1$ "particles" over two paths, and erasing (or post-selecting) adequately one of the two paths.

▶ Remark 2. Thanks to the compact structure and its interaction with the generators of the language, all upside-down version of the equations of Figure 1 are derivable, by simply deforming the diagrams to get the actual axiom. For instance, the upside-down version of $(+)$ can be derived as follows:



▶ **Proposition 3.** *All equations in* ZW_d *are sound, i.e.:*

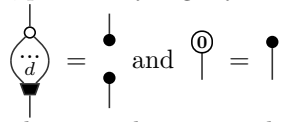
$$\mathrm{ZW}_d \vdash D_1 = D_2 \implies [\![D_1]\!] = [\![D_2]\!]$$

Proof. This is a straightforward verification for most of the equations. They can all be proven using the aforementioned identities (Equations 1 to 5) in the semantics of the diagrams, especially the decomposition of the identity (Equation 5). Equations (b_2) and $(+)$ require respectively the Vandermonde identity and the binomial formula:

$$\sum_{k_1+\ldots+k_p=m} \binom{n_1}{k_1} \ldots \binom{n_p}{k_p} = \binom{n_1+\ldots+n_p}{m} \quad \text{and} \quad (r+s)^n = \sum_{k=0}^{n} \binom{n}{k} r^k s^{n-k}. \qquad \blacktriangleleft$$

2.3 Minimality

Minimality of an equational theory states that every single equation is necessary: none can be derived from the others. Said otherwise, as soon as we remove one of the equations, some equalities (that were previously provable) become unprovable. Minimality is fundamental, as it allows us to pinpoint properties that are necessary to our model, and a contrario those that are consequences of the necessary ones. Notice however that there is usually not a single minimal equational theory, for two reasons: 1) it often happens that one equation can be replaced by an equivalent one, and 2) one could start with a totally different set of equations. Most of the equations we chose are generalisations or adapatations of equations that were already used by other graphical languages, and are usually motivated by either categorical or physical considerations. We only "filled in the blanks" with equations (ℓ) and (u).

In trying to prove minimality, it often happens that two equations fail to be proven necessary individually, but that the pair (i.e. at least one of the two) can be proven necessary. Such cases underline some sort of proximity between the two equations, and the obstacle it poses to minimality can sometimes be circumvented (somewhat artificially) by merging them into a single, potentially slightly less intuitive, equation. This happened once here: we merged equations  , that we initially had as axioms, into Equation (h). This finally provides us with a minimal equational theory for qudit ZW-calculus.

To prove that an equation is necessary, we define a non-standard interpretation which is preserved by all the equations (including the axioms of †-compact props), except the

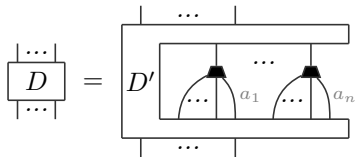
equation of interest. When such an interpretation is exhibited, we can safely conclude that the equation is necessary, since if it were a consequence of the others, it would also preserve this interpretation. Interpretations like these sometimes simply take the form of a quantity that turns out to be invariant for all equations except the one that is considered. In the realm of quantum graphical theories, such arguments were used for single equations in [23, 30, 32, 39, 44], partial minimality results were obtained for Clifford ZX-calculus [6], unrestricted ZX-calculus [47], and quite recently, full minimality (with completeness) was obtained for quantum circuits [12].

In the following, we show that the equational theory ZW_d from Figure 1 is minimal, i.e. that none of the equations can be derived from the others. It is to be noted that most of the equations in Figure 1 are schemas, that is they are parametrised, and the equation is assumed for all possible values of the parameters. Our minimality result is "weak" in the sense that for each equation schema, we show that *at least* one of the occurrences cannot be derived from the other equations, but we do not pinpoint for which parameters the equation is necessary or not. Nevertheless:

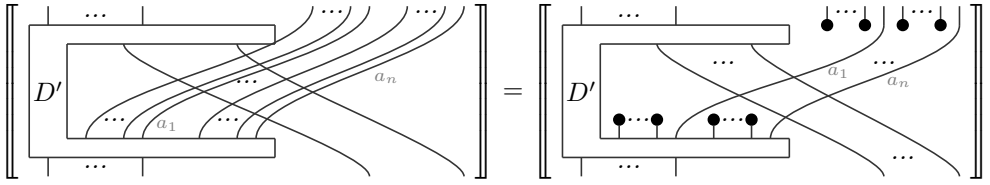
▶ **Theorem 4.** *All equations in* ZW_d *are necessary, hence* ZW_d *is minimal.*

Several arguments in the proof are graphical, meaning that they rely on the understanding that a diagram can also be seen as a special kind of graph with vertices (inputs, outputs, Z-spiders, W-nodes, states, global scalars) and edges (wires, cup, cap, with swaps representing crossings). As such, when we talk about "a W-node in a diagram D", we actually refer to a vertex (of the Z type) of said diagram. One argument in the proof require the notion of "effective Z-path", that is a path that goes only through Z-spiders and trivial W-nodes – which W-nodes that could be replaced by identity wires without it making any difference – and that can be used in non-trivial computations.

▶ **Definition 5** (Sole effective output of a W-node)**.** *Let D be a diagram with a collections of n distinguished W-node. Let's call "a_1",…,"a_n" one output of each, as shown in Equation* (6) *below. We say that wires "a_1",…,"a_n" are* jointly *the* sole effective outputs *of the W-nodes if Equation* (8) *is satisfied:*

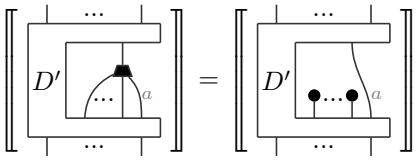


(6)



(7)

In particular, if we only consider one W-node that has a sole effective output – we call such a node a *trivial W-node* – it follows that:



(8)

▶ **Definition 6** (Effective Z-path). *An effective Z-path of a diagram D is a path going from a boundary to another (inputs and/or outputs), such that:*

- *For each W-node it goes through, it cannot use two outputs of the W-node (so it must use its input). Considering all those W-nodes at once, those outputs must be jointly the sole effective outputs of those W-nodes.*
- *There exists a state $|\phi\rangle$ that is not of the form $\lambda|0\rangle$ or $\lambda|1\rangle$ for some $\lambda \in \mathbb{C}$, that* inhabits *the path. That is, if we feed to $[\![D]\!]$ the state $|\phi\rangle$ and/or the effect $\langle\phi|$ to the two inputs and/or outputs of the path, then the result is not $\vec{0}$.*

We can now prove Theorem 4:

Proof. We consider each of these equations individually, and give more details in the full version:

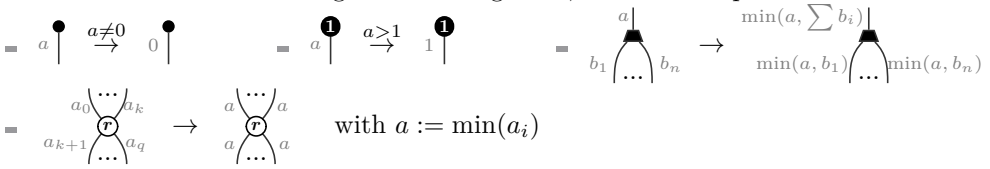
(s) When applying the transformation that turns all Z-spider parameters and global scalars to their real part ($r \mapsto \mathrm{Re}(r)$), Equation (s) is the only one that is not preserved.

(a) It is the only equation permitting to create non-trivial W-nodes with arity $> d$ from a diagram that only has non-trivial W-nodes with arity $\leq d$.

(id) It is the only equation that can create nodes connected to boundaries from a node-free diagram.

(h) To each wire in a diagram D, we associate a number $0 \leq k < d$ (or more graphically we annotate each wire by some number k)[3]. The procedure to do so is as follows:

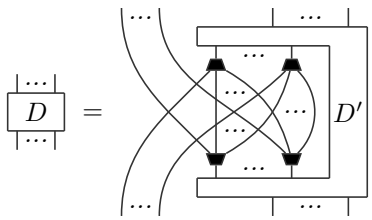
1. annotate all wires with $d-1$
2. rewrite the annotations using the following rules, until a fixed point is reached:



This simple procedure obviously terminates, as a step is only applied if at least one of the annotations is decreased. By considering inputs and outputs of D (which are the only wires that can be guaranteed to remain during rewrites with ZW_d), we can check that Equation (h) is the only one that can modify the outcome of the procedure.

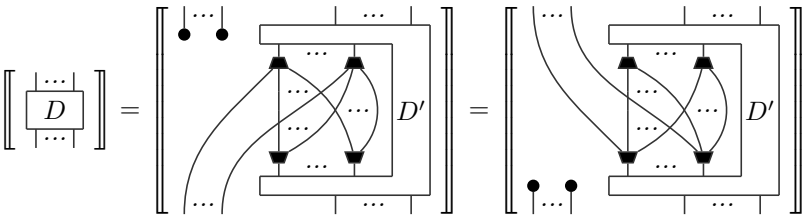
(b_1) Consider diagrams as graphs, and use the above definition of an "effective Z-path". All equations except (b_1) preserve the existence of effective Z-paths, although proving it for (b_2) is quite involved and relies on the following lemma:

▶ **Lemma 7.**
Let D be a diagram of the form shown in Equation (9)*, such that we have Equation* (10)*.*



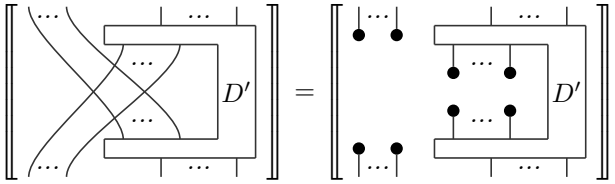
(9)

[3] The annotations produced here are upper bounds on the value of the kets that can go through the wires. As such, they are very closely related to the *capacities* from Section 3.



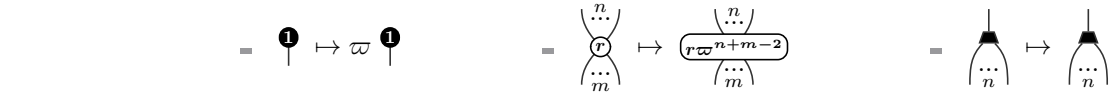
(10)

Then Equation (11) *is satisfied:*



(11)

- (**b_2**) Consider diagrams as graphs, and define a "W-path" in the diagram as a path 1) that goes from a boundary to another boundary, 2) which cannot use two outputs of a W-node (if it goes through a W-node, it has to use the input edge) and 3) that does not go through a Z-spider. All equations, except (b_2), preserve the existence of a W-path. (b_2) is the only equation that can bring the number of W-paths from non-zero to zero (which is done by adding a Z-spider on the path).
- (**$+$**) Take the interpretation that maps all the Z-spider parameters (and the global scalars) to their absolute value ($r \mapsto |r|$). This interpretation preserves all equations except ($+$).
- (**e**) It is the only equation that can create generators out of empty diagrams.
- (**φ**) It is the only equation that can create a non-real scalar from a diagram with only real scalars.
- (**ℓ**) Let $\varpi := e^{i\frac{\pi}{d-1}}$ be the first $(2d-2)$-th root of unity. Consider the †-compact monoidal functor (i.e. functor that preserves compositions, symmetry and compact structure) that maps the generators as follows:

  When $d > 2$, all equations but (ℓ) are preserved by this functor. We can make the argument work when $d = 2$ by choosing any ϖ such that $\varpi^2 \neq 1$ and by working up to a scalar factor.
- (**u**) Take the interpretation that maps  (and subsequently all) to  . All rules hold (up to a the scalar factor) except (u). ◀

2.4 Completeness

Completeness of an equational theory with respect to a semantics is the fundamental property that ensures that semantical equivalence of diagrams is entirely captured by the equational theory. Minimality is worthless without some form of completeness, as it is extremely simple to design minimal, but not complete, equational theories. For instance, the empty equational theory (that contains no axiom) is minimal but clearly not complete for the qudit ZW-diagrams. We hence show in this section that we indeed have completeness.

2.4.1 Normal Form and Universality

The usual way to prove completeness is to show that any diagram can be put in a normal form, and that this normal form is unique and similar for all equivalent diagrams. As is customary in a category that is compact-closed, we can focus on states, as there is an isomorphism between operators and states [1]:

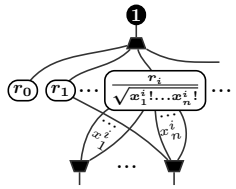
Proving completeness requires a fair amount of diagrammatic derivations, especially when starting from a minimal equational theory, to get enough material to define a normalisation strategy.

▶ **Definition 8.** *We define* $\mathcal{N} : \mathbf{Qudit}_d \to \mathbf{ZW}_d$ *as the functor that maps any* n*-qudit state*

$$|\psi\rangle = r_0 \,|0...0\rangle + r_1 \,|0...01\rangle + ... + r_i \left|x_1^i ... x_n^i\right\rangle + ...$$

to the diagram below. We say of any diagram in the image of $\mathcal{N}$ *that it is in normal form.*

$\mathcal{N}(|\psi\rangle) =$



This construction is a direct generalisation of the normal form of the qubit ZW-diagrams [24], which is also considered in [25] in the context of q-arithmetic. It creates a diagram whose interpretation is the starting state:

▶ **Proposition 9.** $\forall\, |\psi\rangle \in \mathbf{Qudit}_d[0, n],\ [\![\mathcal{N}(|\psi\rangle)]\!] = |\psi\rangle$.

Where for any category $\mathcal{C}$, we write $\mathcal{C}[n, m]$ for the set all the morphisms from n to m. As a simple consequence of this proposition, any qudit operator can be represented by a diagram of $\mathbf{ZW}_d$:

▶ **Corollary 10** (Universality). $\forall f \in \mathbf{Qudit}_d[n, m],\ \exists D_f \in \mathbf{ZW}_d[n, m],\ [\![D_f]\!] = f$.

Since we defined the normal form as the image of a map from the semantics, any diagram can only be associated to a unique normal form.

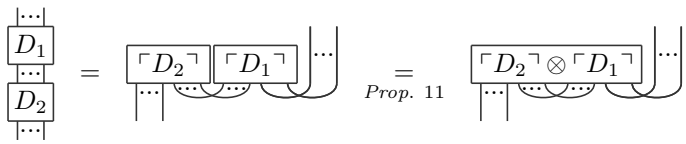
2.4.2 Completeness

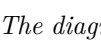
Our goal now is to show that any diagram can be put in normal form. To do so, we show that all generators can be put in normal form, and that all compositions of diagrams in normal form can be put in normal form.

We start by showing the latter for the tensor product:

▶ **Proposition 11.** *The spatial composition of diagrams in normal form can be put in normal form, i.e.* $\mathrm{ZW}_d \vdash \mathcal{N}(v_1) \otimes \mathcal{N}(v_2) = \mathcal{N}(v_1 \otimes v_2)$.

When turning arbitrary operators into states, the sequential composition turns into the application of cups ◡ onto pairs of outputs of the state, as:



▶ **Proposition 12.** *The diagram obtained by applying a cup*  *to two outputs of a diagram in normal form can be put in normal form.*

Then we move on to showing that all the generators can be put in normal form. To do so, the following lemma will prove useful:

▶ **Lemma 13.** *The diagram obtained by applying*  *to two outputs of a normal form can be put in normal form.*

▶ **Proposition 14.** *All generators of the* $\mathbf{ZW}_d$*-calculus can be put in normal form.*

Putting all the latter results together, we can show the completeness of the language:

▶ **Theorem 15** (Completeness for Qudit Systems)**.** *The language is complete: for any two diagrams* D_1 *and* D_2 *of the* $\mathbf{ZW}_d$*-calculus:*

$$[\![D_1]\!] = [\![D_2]\!] \iff \mathrm{ZW}_d \vdash D_1 = D_2$$

Proof. By Proposition 14, any generator of the language can be put in normal form. Thanks to Propositions 11 and 12, compositions of diagrams in normal form can be put in normal form. As a consequence, any diagram can be put in normal form. By uniqueness of this normal form, if two diagrams share the same semantics, they can be rewritten into the same diagram. This proves completeness. ◀

3 Finite Dimensional Hilbert Spaces

In the previous setting, all systems are required to be d-dimensional for some fixed d. Here we relax that constraint, which allows us to go "mixed-dimensional" and to represent morphisms of **FdHilb**[4].

FdHilb is the strict symmetric monoidal †-compact category of finite dimensional Hilbert spaces [1]. Its objects are tensor products of finite dimensional Hilbert spaces $\mathbb{C}^d (d \in \mathbb{N} \setminus \{0\})$, and its morphisms are linear maps between them. The symmetry and the compact structure are naturally extended from that of $\mathbf{Qudit}_d$.

In this new setting, we will be able to represent *all* morphisms of **FdHilb**, at the cost of annotating the wires of the diagrams to keep track of their dimensions. Instead of the dimension itself, we rather annotate the wire with its dimension -1, i.e. with the largest k such that $|k\rangle$ is allowed on the wire. We call such k the *capacity* of the wire. This makes the bookkeeping a little bit less tedious.

3.1 Diagrams and Interpretation

We also require the following constraints for the capacities around each generator:

- All capacities around a Z-spider are the same
- The input capacity of the W-node must be larger than (or equal to) each of its outputs

The first constraint follows from the fact that Z-spiders in ZW can be seen as a generalisation of graph edges – more precisely they can be seen as hyperedges. Hence the whole hyperedge should have a single capacity. The second constraint simply comes from the fact that a larger capacity on the outputs of a W-node will never be used, so we might as well prevent

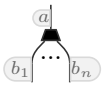
[4] Technically, the skeleton of **FdHilb**, i.e. where all d-dimensional Hilbert spaces are identified with the canonical representative $\mathbb{C}^d$. We take the liberty in this paper to name **FdHilb** this skeleton.

it. When considering $1 \to 1$ W-nodes, which represent projections, this restriction allows us to see at a glance which side has the largest dimension.

The first restriction further allows us to put the capacity annotation on the Z-spider rather than on all its legs, making annotating diagrams less cumbersome.

We now work with a †-compact symmetric monoidal category, which is not a prop anymore. Our base objects are $\mathbb{C}^d$ for $d \in \mathbb{N} \setminus \{0\}$. Every pair of objects can be composed with $\otimes$ to form a third object, with $\otimes$ being associative, and with the tensor unit I being $I := \mathbb{C}^1$. We work with a strict monoidal category, so we consider $I \otimes \mathbb{C}^d = \mathbb{C}^d = \mathbb{C}^d \otimes I$. To simplify notations, we represent objects $\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes ... \otimes \mathbb{C}^{d_n}$ by a list of capacities $(\!|d_1 - 1, d_2 - 1, ..., d_n - 1|\!)$ (the tensor product simply becomes the concatenation of lists). The tensor unit is represented by $(\!|\,|\!)$. Since $(\!|0|\!) = (\!|\,|\!)$, we forbid 0 capacities on the wires.

In this new setting, the generators are generalised as follows:

- Z-spiders  $: (\!|\overbrace{a, ..., a}^{n}|\!) \to (\!|\overbrace{a, ..., a}^{m}|\!)$ with $r \in \mathbb{C}$ and $a \geq 1$
- W-nodes  $: (\!|a|\!) \to (\!|b_1, ..., b_n|\!)$ with $a \geq \max_{1 \leq i \leq n} (b_i)$ and $b_i \geq 1$
- state $|1\rangle$  $: (\!|\,|\!) \to (\!|a|\!)$ with $a \geq 1$
- global scalars $r : (\!|\,|\!) \to (\!|\,|\!)$ with $r \in \mathbb{C}$

with the symmetry and the compact structure being generalised to
-  $: (\!|a, b|\!) \to (\!|b, a|\!)$
-  $: (\!|\,|\!) \to (\!|a, a|\!)$ and  $: (\!|a, a|\!) \to (\!|\,|\!)$,

(again with $a, b \geq 1$) and the identity to $: (\!|a|\!) \to (\!|a|\!)$ $(a \geq 1)$.

Diagrams can still be composed together both sequentially and in parallel. The sequential composition prevents us from composing diagrams with unmatched objects (e.g. two Z-spiders with different capacities in sequence). Diagrams with capacities are called $\mathbf{ZW_f}$-diagrams, and are graphical representations of the morphisms of the †-compact symmetric monoidal category **FdHilb** (the dagger functor can be given in $\mathbf{ZW_f}$ in a similar way as $\mathbf{ZW}_d$).

The compact structure still allows us to define upside-down versions of the W-node and the kets. Again, we give the $(\!|\,|\!) \to (\!|a|\!)$ W-node a special symbol (for $a \geq 1$):  :=  , and we generalise the ket symbol inductively as follows (for $k \geq 1$ and $a > k$):  :=  .

The interpretation of these diagrams is now a monoidal functor $[\![.]\!] : \mathbf{ZW_f} \to \mathbf{FdHilb}$ inductively defined as:

$$[\![D_2 \circ D_1]\!] = [\![D_2]\!] \circ [\![D_1]\!] \qquad \left[\!\!\left[\begin{array}{c} a \\ \diagup\!\diagdown \\ b \end{array} \right]\!\!\right] = \sum_{k=0}^{a} \sum_{\ell=0}^{b} |\ell, k\rangle\langle k, \ell|$$

$$[\![D_1 \otimes D_2]\!] = [\![D_1]\!] \otimes [\![D_2]\!]$$

$$\left[\!\!\left[\begin{array}{c} a \\ | \end{array} \right]\!\!\right] = \sum_{k=0}^{a} |k\rangle\langle k| \qquad \left[\!\!\left[\begin{array}{c} n \\ \vdots \\ a\ (r) \\ \vdots \\ m \end{array} \right]\!\!\right] = \sum_{k=0}^{a} r^k \sqrt{k!}^{n+m-2} |k^m\rangle\langle k^n|$$

$$\left[\!\!\left[\begin{array}{c} a \\ \cap \end{array} \right]\!\!\right] = \left[\!\!\left[\begin{array}{c} \cup \\ a \end{array} \right]\!\!\right]^{\dagger} = \sum_{k=0}^{a} |k, k\rangle \qquad \left[\!\!\left[\begin{array}{c} \bullet 1 \\ a \end{array} \right]\!\!\right] = |1\rangle$$

$$[\![r]\!] = r$$

$$\left[\!\!\left[\begin{array}{c} a \\ \triangle \\ b_1 \cdots b_n \end{array} \right]\!\!\right] = \sum_{\substack{0 \leq k_i \leq b_i \\ k_1 + \ldots + k_n \leq a}} \sqrt{\binom{k_1 + \ldots + k_n}{k_1, \ldots, k_n}} |k_1, \ldots, k_n\rangle\langle k_1 + \ldots + k_n|$$

By composition, one can check that, for $k > 1$ and $a \geq k$: $\left[\!\!\left[\begin{array}{c} \bullet k \\ a \end{array} \right]\!\!\right] = \sqrt{k!}\,|k\rangle$.

Notice that we use the same notation for the interpretation of $\mathbf{ZW}_d$-diagrams, and for the interpretation of the $\mathbf{ZW_f}$-diagrams. Which interpretation we are referring to should be clear from the context.

3.2 Complete Equational Theory

We once again equip the language with an equational theory ZW_f, defined in Figure 2. This equational theory only slightly differs from the one for qudit systems in Figure 1. It is interesting to notice that 1) the associativity of the W-node is broken down into two equations (a) and (o), whose choice depends on the capacities involved, 2) the W-bialgebra equation (b_2) does not need a context anymore, but instead side conditions on the capacities, 3) we managed to remove Equation (e), 4) we now need an equation (i) that states that a $|1\rangle$, when "injected" into a larger dimensional Hilbert space, is still a $|1\rangle$.

We also notice the existence of an interesting equation, that we did not include in Figure 2 as it turns out to be derivable; and which states that a $0 \to 1$ Z-state can be "copied" by the W-node, as follows: $\left[\!\!\left[\begin{array}{c} a+b \\ \triangle \\ a \quad b \end{array} \right]\!\!\right] = \left[\!\!\left[\begin{array}{c} a \quad b \\ \curlyvee \end{array} \right]\!\!\right]$.

The category $\mathbf{Qudit}_d$ is a full subcategory of $\mathbf{FdHilb}$, and as such there is an obvious inclusion functor $\mathbf{Qudit}_d \overset{i_d}{\hookrightarrow} \mathbf{FdHilb}$. This inclusion transports to the ZW-calculi: we can turn any $\mathbf{ZW}_d$-diagram into a $\mathbf{ZW_f}$-diagram through ι_d in such a way that the following diagram commutes:

$$\begin{array}{ccc} \mathbf{ZW}_d & \xrightarrow{\quad\iota_d\quad} & \mathbf{ZW_f} \\ \Big\downarrow{\scriptstyle [\![.]\!]} & & \Big\downarrow{\scriptstyle [\![.]\!]} \\ \mathbf{Qudit}_d & \xrightarrow{\quad i_d\quad} & \mathbf{FdHilb} \end{array}$$

The functor ι_d simply takes a $\mathbf{ZW}_d$-diagram and annotates all its wires with $d-1$.

We show that the present equational theory is complete. To do so, we need to adapt the notion of normal form from qudit systems (and we again use the map/state duality to focus on states rather than arbitrary morphisms):

▶ **Definition 16.** *We define* $\mathcal{N} : \mathbf{FdHilb} \to \mathbf{ZW_f}$ *as the functor that maps any n-ary state* $|\psi\rangle \in \mathbf{FdHilb}[(\!|), (\!|a_1, ..., a_n|\!)]$*:*

$$|\psi\rangle = r_0 \,|0...0\rangle + r_1 \,|0...01\rangle + ... + r_i \left|x_1^i ... x_n^i\right\rangle + ...$$

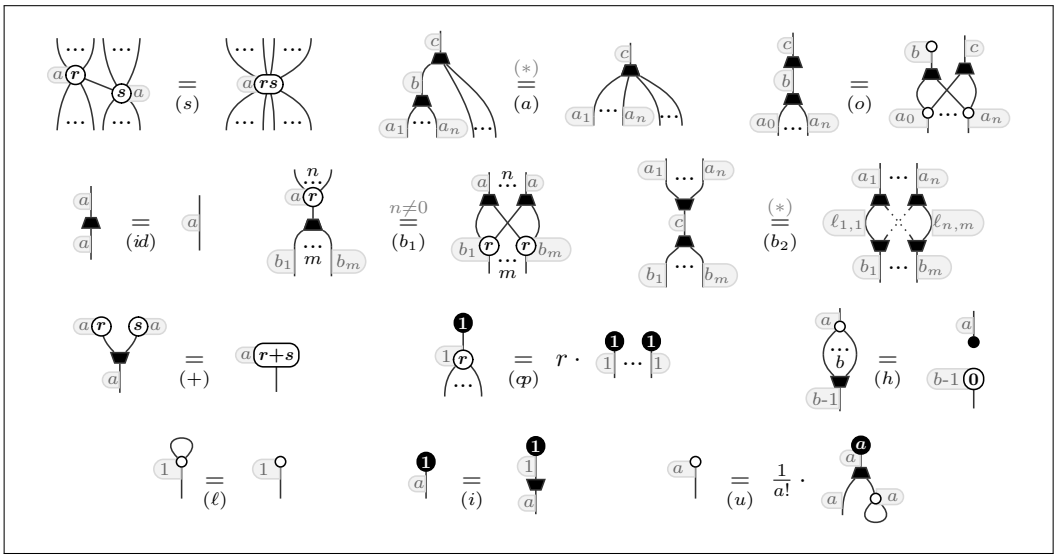
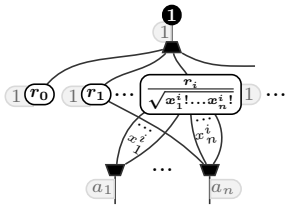


Figure 2 Equational theory ZW_f for the finite-dimensional **ZW**-calculus. In (a), we require that $b = c$ or $b \geq \sum_i a_i$; and in (b_2) that $c \geq \min(\sum a_i, \sum b_i)$ on the lhs, and that $\ell_{ij} = \min(a_i, b_j)$ on the rhs.

to the diagram below. We say of any diagram in the image of $\mathcal{N}$ that it is in normal form.

$\mathcal{N}(|\psi\rangle) =$ 

We can once again show that $\mathcal{N}$ builds a diagram that represents $|\psi\rangle$:

▶ **Lemma 17.** $\forall\, |\psi\rangle \in \mathbf{FdHilb}[(\!|\mathbb{1}|\!), (\!|a_1, ..., a_n|\!)],\ [\![\mathcal{N}(|\psi\rangle)]\!] = |\psi\rangle$

We hence get universality of the language as a direct consequence:

▶ **Corollary 18** (Universality of $\mathbf{ZW_f}$).
$\forall f \in \mathbf{FdHilb}[(\!|a_1, ..., a_n|\!), (\!|b_1, ..., b_m|\!)],\ \exists D_f \in \mathbf{ZW_f}[(\!|a_1, ..., a_n|\!), (\!|b_1, ..., b_m|\!)],\ [\![D_f]\!] = f$

Most of the arguments given for the minimality of ZW_d can be adapted to arguments for the necessity of the equations of ZW_f, and the few remaining equations can be given a new argument. We hence have:

▶ **Theorem 19.** *The equational theory ZW_f is minimal.*

Proof. The argument given for the necessity of Equation (b_2) now works for the necessity of Equation (o), and that of Equation (e) now works for Equation (b_2). Namely, Equation (b_2) is the only that can create a non-empty diagram from an empty diagram. A less artificial argument for Equation (b_2) is that it is the only equation that can create a capacity $> k$ from a diagram whose capacities are all $\leq k$ for $k \geq 1$. Moreover:

Equation (i) is the only equation that can create a $|1\rangle$ with capacity $a \neq 1$, from a diagram whose $|1\rangle$s are all on capacity 1.

We can reformulate the argument of Equation (a) as: It is the only equation permitting to create non-trivial W-nodes with arity ≥ 3 from a diagram where all non-trivial W-nodes have arity ≤ 2.

In the argument of Equation (ℓ), one can take for ϖ any complex number such that $\varpi^2 \neq 1$, and by working up to a scalar factor, as is done initially for $d = 2$.

In the argument of Equation (h), we can instantiate the protocol by annotating the wires with their capacities, then continue with the protocol as explained in the initial argument.

All the other arguments work right off the bat for their mixed-dimensional counterpart, hence the result of minimality. ◀

Using the normal form, we can then leverage the completeness from the qudit ZW-calculus to get the similar result in the current setting:

▶ **Theorem 20** (Completeness for Finite Dimensional Systems). *The language is complete: for any two diagrams D_1 and D_2 of the* $\mathbf{ZW_f}$*-calculus:*

$$[\![D_1]\!] = [\![D_2]\!] \iff \mathrm{ZW}_f \vdash D_1 = D_2$$

Proof. The right-to-left implication (soundness) is again a straightforward verification. The other is proven in its entirety in the full version, and uses the completeness of ZW_d for qudit systems. The idea is to show that i) we can turn both D_1 and D_2 into a diagram with a high enough capacity d everywhere (except boundaries), and that ii) all the equations of ZW_d can be proven in ZW_f (through ι_d). ◀

4 Related Work

4.1 Qudit Framework

The first and only result to date of a complete equational theory for a graphical language describing qudit systems comes from the "ZXW-calculus" [42]. There, the authors start from a qudit version of the ZX-calculus and most probably end up requiring a W-node in the definition of a normal form, and hence in the equational theory leading to completeness. We argue here that we can get a complete equational theory purely inside the ZW-calculus. By keeping the number of generators as low as possible, we also end up with few, intuitive equations in the equational theory.

The W-node we used is a different generalisation of the qubit W-node than the one used in [42]. The version we used offers two advantages with respect to the aims of the paper. First, it allows to use a single parameter in the Z-spiders (which aligns with the spirit of keeping things as minimal as possible) and to sum such parameters together, while the other version requires to have $(d-1)$-sized lists of coefficients as parameters in order to get a $(+)$-like rule to sum coefficients together. Second, it allows us to define $|k\rangle$ (up to a scalar) as a composition using only $|1\rangle$ and the W-node. Again, this lowers the number of generators, as all the $|k\rangle$ (for $k > 1$) become syntactic sugar.

Focussing on ZW-calculus is not a new idea. The first ever completeness proof for qubit graphical languages was in the (qubit) ZW-calculus, introduced in [14] and tweaked and made complete in [24, 26]. The ZW-calculus noticeably has very nice combinatorial properties different from those of its counterparts, which in particular allows for a very natural notion of normal form. It is hence not suprising that some attempts were made to get a complete equational theory of qudit systems purely in ZW. There have then been tentative generalisations for qudit systems, in particular in [25] where q-arithmetic is used, and in [49]

where the W-node is generalised in a different way (and that we encounter in [42]). It is to be noted that our two main generators are essentially the same as in [25], except with usual arithmetic instead of q-arithmetic. While some equations are sound with respect to the q-arithmetic semantics, others are truly specific to the standard arithmetic. Adapting the results of the present paper to q-arithmetic semantics hence seems non-trivial. Other presentations for qudit systems have also been proposed (without proof of completeness) in [17, 43]. Finally, complete presentations for fragments of qudit quantum mechanics can be found e.g. in [8, 40].

Another system we are close to is **QPath** [19]. Our W-node is merely the "triangle" node of **QPath** that we truncated to a finite dimension[5], and we generalised their "line weight" to an n-ary Z-spider. The degree-2 Z-spider furthermore has exactly the same interpretation as the line weight. While in **QPath** the triangle nodes satisfy a bialgebra, this is not the case when truncating to finite dimension. Here we could either resort to define a "fermionic swap" that would replace the usual swap in the bialgebra (as in [25] and [49]), or give a context in which the bialgebra works (as is done in [42]). While such a "fermionic swap" exists in our setting, it does not have all the nice properties of the qubit fermionic swap, that in particular allow us to see it as a quasi-proper swap. Instead we went with the latter solution, which as it turns out works in our setting, despite the W-node having a different interpretation from that of [42], and we end up with Equation (b_2).

4.2 Finite Dimensional Framework

Another complete presentation of a graphical language for **FdHilb** was announced recently before the first version of the current paper [50]. This one builds upon the aforementioned ZXW-calculus, and introduces a new generator that takes two systems, of dimensions a and b, and builds a system of dimension $a \times b$. Our approach builds upon $\mathbf{ZW}_d$, the qudit version of the ZW-calculus from Section 2 and hence starts with fewer generators and equations. As a consequence, the graphical language for **FdHilb** we end up with has fewer equations as well. Moreover, we did not require a new generator, and simply promoted the qudit W-node to work with any mix of dimensions in a natural manner, which was enough to provide us with universality.

A version of the ZX-calculus for **FdHilb** was recently provided and shown to be complete [41]. The proof of completeness for their graphical language was obtained by transporting the property from the $\mathbf{ZW}_\mathbf{f}$-calculus of the first version of the current paper to the ZX-calculus, through a system of translations between the two languages.

5 Conclusion

In this paper, we explored the potential for a minimal yet complete diagrammatic language for quantum mechanics beyond qubit systems. This starts with a well-chosen generalisation of the generators of the ZW-calculus, allowing us to have few and intuitive equations. For both qudit systems and finite dimensional systems, we showed that the diagrams are universal, and that the equational theories are both minimal and complete for their respective interpretation.

[5] The idea of truncating this tensor has also been considered in [20] during a translation between graphical languages.

References

1 S. Abramsky and B. Coecke. A categorical semantics of quantum protocols. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004.*, pages 415–425, July 2004. doi:10.1109/LICS.2004.1319636.

2 Miriam Backens. The ZX-calculus is complete for stabilizer quantum mechanics. In *New Journal of Physics*, volume 16, page 093021. IOP Publishing, September 2014. doi:10.1088/1367-2630/16/9/093021.

3 Miriam Backens and Aleks Kissinger. ZH: A complete graphical calculus for quantum computations involving classical non-linearity. In Peter Selinger and Giulio Chiribella, editors, *Proceedings of the 15th International Conference on Quantum Physics and Logic, Halifax, Canada, 3-7th June 2018*, volume 287 of *Electronic Proceedings in Theoretical Computer Science*, pages 23–42, 2019. doi:10.4204/EPTCS.287.2.

4 Miriam Backens, Aleks Kissinger, Hector Miller-Bakewell, John van de Wetering, and Sal Wolffs. Completeness of the ZH-calculus. *Compositionality*, 5, July 2023. doi:10.32408/compositionality-5-5.

5 Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski, and John van de Wetering. There and back again: A circuit extraction tale. *arXiv: Quantum Physics*, 2020. arXiv:2003.01664.

6 Miriam Backens, Simon Perdrix, and Quanlong Wang. Towards a Minimal Stabilizer ZX-calculus. *Logical Methods in Computer Science*, Volume 16, Issue 4, December 2020. doi:10.23638/LMCS-16(4:19)2020.

7 Robert I. Booth and Titouan Carette. Complete ZX-Calculi for the Stabiliser Fragment in Odd Prime Dimensions. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:15, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2022.24.

8 Robert I. Booth, Titouan Carette, and Cole Comfort. Graphical symplectic algebra, 2024. doi:10.48550/arXiv.2401.07914.

9 Titouan Carette. *Wielding the ZX-calculus, Flexsymmetry, Mixed States, and Scalable Notations. (Manier le ZX-calcul, flexsymétrie, systèmes ouverts et limandes)*. PhD thesis, University of Lorraine, Nancy, France, 2021. URL: https://tel.archives-ouvertes.fr/tel-03468027.

10 Titouan Carette, Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. Completeness of Graphical Languages for Mixed States Quantum Mechanics. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 108:1–108:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2019.108.

11 Titouan Carette, Etienne Moutot, Thomas Perez, and Renaud Vilmart. Compositionality of Planar Perfect Matchings: A Universal and Complete Fragment of ZW-Calculus. 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 120:1–120:17, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.120.

12 Alexandre Clément, Noé Delorme, and Simon Perdrix. Minimal equational theories for quantum circuits, 2023. doi:10.48550/arXiv.2311.07476.

13 Bob Coecke and Ross Duncan. Interacting quantum observables: Categorical algebra and diagrammatics. *New Journal of Physics*, 13(4):043016, April 2011. doi:10.1088/1367-2630/13/4/043016.

14 Bob Coecke and Aleks Kissinger. The compositional structure of multipartite quantum entanglement. In *Automata, Languages and Programming*, pages 297–308. Springer Berlin Heidelberg, 2010. doi:10.1007/978-3-642-14162-1_25.

15 Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, Cambridge, 2017. doi:10.1017/9781316219317.

16 Joseph Collins and Ross Duncan. Hopf-frobenius algebras and a simpler drinfeld double. *Electronic Proceedings in Theoretical Computer Science*, 318:150–180, May 2020. doi:10.4204/eptcs.318.10.

17 Niel de Beaudrap and Richard D. P. East. Simple zx and zh calculi for arbitrary finite dimensions, via discrete integrals, 2023. arXiv:2304.03310.

18 Niel de Beaudrap and Dominic Horsman. The ZX calculus is a language for surface code lattice surgery. *Quantum*, 4:218, January 2020. doi:10.22331/q-2020-01-09-218.

19 Giovanni de Felice and Bob Coecke. Quantum linear optics via string diagrams. In Stefano Gogioso and Matty Hoban, editors, *Proceedings 19th International Conference on Quantum Physics and Logic, QPL 2022, Wolfson College, Oxford, UK, 27 June - 1 July 2022*, volume 394 of *EPTCS*, pages 83–100, 2022. doi:10.4204/EPTCS.394.6.

20 Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang, and Bob Coecke. Light-matter interaction in the zxw calculus. *Electronic Proceedings in Theoretical Computer Science*, 384:20–46, August 2023. doi:10.4204/eptcs.384.2.

21 Ross Duncan and Kevin Dunne. Interacting frobenius algebras are hopf. In *2016 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, 2016.

22 Ross Duncan and Maxime Lucas. Verifying the Steane code with Quantomatic. In Bob Coecke and Matty Hoban, editors, *Proceedings of the 10th International Workshop on Quantum Physics and Logic, Castelldefels (Barcelona), Spain, 17th to 19th July 2013*, volume 171 of *Electronic Proceedings in Theoretical Computer Science*, pages 33–49. Open Publishing Association, 2014. doi:10.4204/EPTCS.171.4.

23 Ross Duncan and Simon Perdrix. Graphs states and the necessity of Euler decomposition. *Mathematical Theory and Computational Practice*, 5635:167–177, 2009. doi:10.1007/978-3-642-03073-4.

24 Amar Hadzihasanovic. A diagrammatic axiomatisation for qubit entanglement. In *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 573–584, July 2015. doi:10.1109/LICS.2015.59.

25 Amar Hadzihasanovic. *The Algebra of Entanglement and the Geometry of Composition*. PhD thesis, University of Oxford, 2017. arXiv:1709.08086.

26 Amar Hadzihasanovic, Kang Feng Ng, and Quanlong Wang. Two complete axiomatisations of pure-state qubit quantum computing. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 502–511, New York, NY, USA, 2018. ACM. doi:10.1145/3209108.3209128.

27 Anne Hillebrand. Quantum Protocols involving Multiparticle Entanglement and their Representations. Master's thesis, University of Oxford, 2011. URL: https://www.cs.ox.ac.uk/people/bob.coecke/Anne.pdf.

28 Jiaxin Huang, Sarah Meng Li, Lia Yeh, Aleks Kissinger, Michele Mosca, and Michael Vasmer. Graphical css code transformation using zx calculus. *Electronic Proceedings in Theoretical Computer Science*, 384:1–19, August 2023. doi:10.4204/eptcs.384.1.

29 Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. A complete axiomatisation of the ZX-calculus for Clifford+T quantum mechanics. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 559–568, New York, NY, USA, 2018. ACM. doi:10.1145/3209108.3209131.

30 Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. Y-calculus: A language for real matrices derived from the zx-calculus. *Electronic Proceedings in Theoretical Computer Science*, 266:23–57, February 2018. doi:10.4204/eptcs.266.2.

31 Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. Completeness of the ZX-Calculus. *Logical Methods in Computer Science*, Volume 16, Issue 2, June 2020. doi:10.23638/LMCS-16(2:11)2020.

32 Emmanuel Jeandel, Simon Perdrix, Renaud Vilmart, and Quanlong Wang. ZX-calculus: Cyclotomic supplementarity and incompleteness for Clifford+T quantum mechanics. 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 11:1–11:13, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2017.11.

33 Aleks Kissinger. Tikzit, 2019. URL: https://tikzit.github.io/index.html.

34 Aleks Kissinger and John van de Wetering. Reducing the number of non-Clifford gates in quantum circuits. *Phys. Rev. A*, 102:022406, August 2020. doi:10.1103/PhysRevA.102.022406.

35 Aleks Kissinger and John van de Wetering. Simulating quantum circuits with zx-calculus reduced stabiliser decompositions. *Quantum Science and Technology*, 7(4):044001, July 2022. doi:10.1088/2058-9565/ac5d20.

36 Aleks Kissinger, John van de Wetering, and Renaud Vilmart. Classical Simulation of Quantum Circuits with Partial and Graphical Stabiliser Decompositions. In François Le Gall and Tomoyuki Morimae, editors, *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*, volume 232 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:13, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.TQC.2022.5.

37 Mark Koch, Richie Yeung, and Quanlong Wang. Speedy contraction of zx diagrams with triangles via stabiliser decompositions, 2023. arXiv:2307.01803.

38 Stephen Lack. Composing PROPs. In *Theory and Applications of Categories*, volume 13, pages 147–163, 2004. URL: http://www.tac.mta.ca/tac/volumes/13/9/13-09abs.html.

39 Simon Perdrix and Quanlong Wang. Supplementarity is necessary for quantum diagram reasoning. In *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*, volume 58 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 76:1–76:14, Krakow, Poland, August 2016. doi:10.4230/LIPIcs.MFCS.2016.76.

40 Boldizsár Poór, Robert I. Booth, Titouan Carette, John van de Wetering, and Lia Yeh. The qupit stabiliser zx-travaganza: Simplified axioms, normal forms and graph-theoretic simplification. *Electronic Proceedings in Theoretical Computer Science*, 384:220–264, August 2023. doi:10.4204/eptcs.384.13.

41 Boldizsár Poór, Razin A. Shaikh, and Quanlong Wang. Zx-calculus is complete for finite-dimensional hilbert spaces, 2024. arXiv:2405.10896.

42 Boldizsár Poór, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung, and Bob Coecke. Completeness for arbitrary finite dimensions of zxw-calculus, a unifying calculus. In *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–14, 2023. doi:10.1109/LICS56636.2023.10175672.

43 Patrick Roy, John van de Wetering, and Lia Yeh. The qudit zh-calculus: Generalised toffoli+hadamard and universality. *Electronic Proceedings in Theoretical Computer Science*, 384:142–170, August 2023. doi:10.4204/eptcs.384.9.

44 Christian Schröder de Witt and Vladimir Zamdzhiev. The zx-calculus is incomplete for quantum mechanics. *Electronic Proceedings in Theoretical Computer Science*, 172:285–292, December 2014. doi:10.4204/eptcs.172.20.

45 Peter Selinger. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pages 289–355. Springer, 2010.

46 Alex Townsend-Teague, Julio Magdalena de la Fuente, and Markus Kesselring. Floquetifying the colour code. *Electronic Proceedings in Theoretical Computer Science*, 384:265–303, August 2023. doi:10.4204/eptcs.384.14.

47 Renaud Vilmart. A near-minimal axiomatisation of zx-calculus for pure qubit quantum mechanics. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, June 2019. doi:10.1109/LICS.2019.8785765.

48 Quanlong Wang. Qutrit zx-calculus is complete for stabilizer quantum mechanics. In Bob Coecke and Aleks Kissinger, editors, *Proceedings 14th International Conference on Quantum Physics and Logic, Nijmegen, The Netherlands, 3-7 July 2017*, volume 266 of *Electronic Proceedings in Theoretical Computer Science*, pages 58–70, 2018. doi:10.4204/EPTCS.266.3.

49 Quanlong Wang. A non-anyonic qudit zw-calculus, 2021. arXiv:2109.11285.

50 Quanlong Wang and Boldizsár Poór. Completeness of qufinite zxw calculus, a graphical language for mixed-dimensional quantum computing, 2023. arXiv:2309.13014.

51 Fabio Zanasi. *Interacting Hopf Algebras – the theory of linear systems*. PhD thesis, Université de Lyon, 2015. URL: http://www.zanasi.com/fabio/#/publications.html.

A Asymmetric Presentation for Qudit Systems

In this section, we give an alternative semantics for the ZW_d-diagrams, which breaks the up/down symmetry of the generators. On the one hand, the dagger-functor becomes less natural; on the other hand, the combinatorics associated to the diagrams becomes simpler (except for the cap):

$$\llbracket D_2 \circ D_1 \rrbracket_{\wr} = \llbracket D_2 \rrbracket_{\wr} \circ \llbracket D_1 \rrbracket_{\wr}$$

$$\llbracket D_1 \otimes D_2 \rrbracket_{\wr} = \llbracket D_1 \rrbracket_{\wr} \otimes \llbracket D_2 \rrbracket_{\wr}$$

$$\left\llbracket \; | \; \right\rrbracket_{\wr} = \sum_k |k\rangle\langle k|$$

$$\left\llbracket \; \times \; \right\rrbracket_{\wr} = \sum_{k,\ell} |\ell, k\rangle\langle k, \ell|$$

$$\left\llbracket \; \cup \; \right\rrbracket_{\wr} = \sum_k k! \, \langle k, k|$$

$$\left\llbracket \; \cap \; \right\rrbracket_{\wr} = \sum_k \frac{1}{k!} \, |k, k\rangle$$

$$\left\llbracket \; \overset{n}{\underset{m}{\overset{\cdots}{\underset{\cdots}{(r)}}}} \; \right\rrbracket_{\wr} = \sum_{k=0}^{d-1} r^k k!^{n-1} \, |k^m\rangle\langle k^n|$$

$$\left\llbracket \; \overset{}{\underset{n}{\overset{\cdots}{\underset{\cdots}{\blacktriangle}}}} \; \right\rrbracket_{\wr} = \sum_{\substack{k \in \{0,\ldots,d-1\} \\ i_1+\ldots+i_n=k}} \binom{k}{i_1, \ldots, i_n} |i_1, \ldots, i_n\rangle\langle k|$$

$$\left\llbracket \; \overset{\cdots}{\blacktriangledown} \; \right\rrbracket_{\wr} = \sum_{\substack{k \in \{0,\ldots,d-1\} \\ i_1+\ldots+i_n=k}} |k\rangle\langle i_1, \ldots, i_n|$$

$$\left\llbracket \; \overset{\mathbf{1}}{|} \; \right\rrbracket_{\wr} = |1\rangle \qquad \text{and} \qquad \left\llbracket \; \underset{\mathbf{1}}{|} \; \right\rrbracket_{\wr} = \langle 1|$$

$$\llbracket r \rrbracket_{\wr} = r$$

As a direct consequence, we have:

$$\left\llbracket \; \overset{k}{|} \; \right\rrbracket_{\wr} = |k\rangle \qquad \text{and} \qquad \left\llbracket \; \underset{k}{|} \; \right\rrbracket_{\wr} = k! \, \langle k|$$

This semantics being equivalent to $\llbracket - \rrbracket_{\wr}$, the equational theory of ZW_d-diagrams remains universal, sound and complete in this setting.

B Lemmas for Completeness of Qudit

The full version on Arxiv details the proof of completeness of $\mathbf{ZW}_d$'s equational theory, including the proof of the following derivable equations:

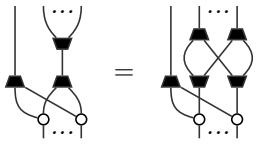
▶ **Lemma 21.**

$\mathrm{ZW}_d \vdash 1 = \; \square$

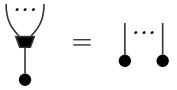
▶ **Lemma 22.**

$\mathrm{ZW}_d \vdash \; \circ \; = \; |$

▶ **Lemma 23.**

$\mathrm{ZW}_d \vdash$  $=$

▶ **Lemma 24.**

$\mathrm{ZW}_d \vdash$  $=$

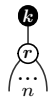
▶ **Lemma 25.**

$\mathrm{ZW}_d \vdash$  $=$ 

▶ **Lemma 26.**

$\mathrm{ZW}_d \vdash$  $=$ 

▶ **Lemma 27.**

$\mathrm{ZW}_d \vdash$  $= r^k \cdot$ 

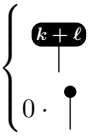
▶ **Lemma 28.**

$\mathrm{ZW}_d \vdash$  $=$ 

▶ **Lemma 29.**

$\mathrm{ZW}_d \vdash$  $=$ 

▶ **Lemma 30.**

$\mathrm{ZW}_d \vdash$  $=$ 

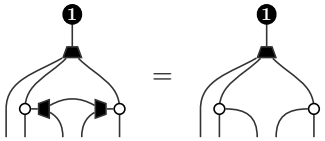
$k+\ell$ *if* $k+\ell < d$

$0 \cdot$ *if* $k+\ell \geq d$

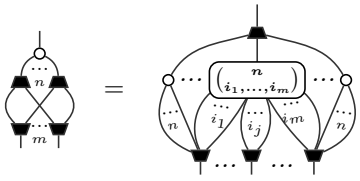
▶ **Lemma 31.**

$\mathrm{ZW}_d \vdash$  $=$ 

▶ **Lemma 32.**

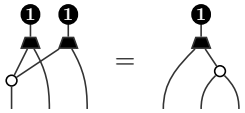
$\mathrm{ZW}_d \vdash$ 

▶ **Lemma 33.**

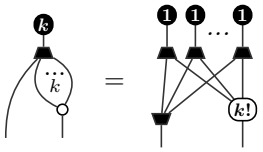
$\mathrm{ZW}_d \vdash$ 

where on the right-hand-side, there are as many Z-spiders as there are ways to decompose n as a sum of m natural numbers: $n = i_1 + ... + i_m$.

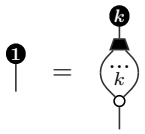
▶ **Lemma 34.**

$\mathrm{ZW}_d \vdash$ 

▶ **Lemma 35.** *With $0 < k < d$:*

$\mathrm{ZW}_d \vdash$ 

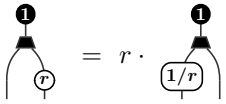
▶ **Lemma 36.** *If $0 < k < d$:*

$\mathrm{ZW}_d \vdash k!$ 

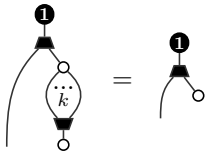
▶ **Lemma 37.** *If $k \neq \ell$:*

$\mathrm{ZW}_d \vdash$  $= k!$ *and* $\mathrm{ZW}_d \vdash$  $= 0$

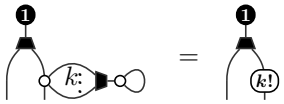
▶ **Lemma 38.**

$\mathrm{ZW}_d \vdash$ 

▶ **Lemma 39.** *If $0 \leq k < d$:*

$\mathrm{ZW}_d \vdash$ 

▶ **Lemma 40.** *If $0 \leq k < d$, we have:*

$\mathrm{ZW}_d \vdash$ 

▶ **Lemma 41.** *The bialgebra between W-nodes can be used in the following context, if $0 \leq k_i < d$ for all i:*

$\mathrm{ZW}_d \vdash$